Drone Station Location and Routing Optimization for Infrastructure Inspection

Peisong Liu, Xiangpei Hu, Chun Cheng*
School of Economics and Management, Dalian University of Technology, Dalian, 116024, China
*Corresponding author. chun.cheng@polymtl.ca

Abstract: Recent advancements in drone technology have expanded drone applications in logistics, humanitarian aid, and infrastructure surveillance. Motivated by the demand for drone-based infrastructure inspection and the emerging design of drone battery swapping stations, this paper introduces the location routing problem with heterogeneous stations and drones (LRPHSD). In the problem, two types of stations are considered: one that supports a single drone, and another that accommodates two heterogeneous drones with different battery capacities. The objective is to determine the locations and types of stations, along with the multi-trip routes of drones, to minimize the total cost. We formulate the LRPHSD as a mixed-integer linear programming model and develop a two-stage adaptive large neighborhood search algorithm (TSALNS) for resolution. Numerical results show that for large-sized LRPHSD instances, TSALNS can efficiently find solutions comparable to or better than those obtained by Gurobi within a 3600-second time limit. For classical location routing problem instances, TSALNS provides solutions with optimality gaps of around 3% for instances with over 150 nodes in under two minutes. The results based on simulated data and a real-world case study collectively demonstrate that a heterogeneous station-drone configuration can achieve a better balance between cost and task completion time compared to various homogeneous configurations.

Key words: Location, Routing, Drone inspection, Heterogeneity, Adaptive large neighborhood search algorithm

1. Introduction

The reliable operation of grid infrastructure is essential to the functioning of modern society, as disruptions or failures in such systems can lead to significant economic and societal consequences (León 2025). Thus, it is important to conduct systematic inspections and implement proactive maintenance to ensure the resilience of critical infrastructure. Existing inspection methods mainly include manual checks, satellite imaging, and ground- or helicopter-based observations. Currently, the inspection of high-altitude grid infrastructure still heavily relies on manual climbing and ground assessments. However, power lines often span remote and complex terrains, many of which are mountainous or forested regions, bringing operational difficulties and safety risks to inspection personnel (Santos et al. 2019).

Fortunately, drones equipped with high-definition imaging systems offer a safer alternative to these traditional methods. Vidović et al. (2024) indicate that using drones for grid infrastructure inspections

can reduce personnel injuries and fatalities by more than 70%. In China, the total distance covered by drone inspections exceeded 5 million kilometers in 2024, with drones replacing 92% of high-risk manual operations (AOPA 2025). Similarly, in Europe, Electricité de France has employed drones for inspections at the Blyth offshore wind farms (Richard 2022).

Besides grid infrastructure inspections, drones have demonstrated broad applicability across various domains. For instance, Hovering Solutions in Spain develops autonomous drones for inspecting underground infrastructure, including water collectors, mining chambers, and hydropower penstocks (Collins 2023). The EU-funded PROACTIF project, led by Nokia, is advancing drone systems designed to safeguard critical infrastructure such as railways and ports (Nin 2025). Other drone-enabled inspection applications include monitoring emissions in controlled areas (Xia et al. 2019), evaluating forest fire risks (De la Fuente et al. 2024), and managing urban traffic (Meng et al. 2024b).

Despite the great potential of drone inspections, current practice still faces challenges. Specifically, human intervention is often necessary for drone operations, such as takeoff/landing support and battery replacement (Dukkanci et al. 2024). Although trucks have been introduced as mobile platforms for drone deployment, enabling on-site launch, recovery, and servicing, the truck–drone model is often constrained by geographical barriers. In China, for instance, approximately 45% of transmission towers are located in mountainous or marshy areas, where poor road conditions restrict truck access.

To address these challenges, researchers and practitioners have proposed alternative solutions, such as drone stations (Cicek et al. 2021, Lejeune & Ma 2025). However, conventional stations typically accommodate only a single drone and rely on battery charging, leading to idle time between tasks and reduced operational efficiency. Thus, recently, some companies have begun to develop new drone stations, e.g., the automatic battery swapping stations, as shown in Figure 1. This type of station utilizes robotic arms to quickly replace drone batteries. Moreover, they can recharge batteries within minutes, and when equipped with multiple batteries, they enable continuous drone operations with negligible idle time. In addition, some stations feature a dual-drone configuration, as shown in Figure 1(b), allowing for the simultaneous dispatch of two heterogeneous drones.

Motivated by the infrastructure inspection scenarios and the novel design of drone stations, this paper introduces a new drone inspection problem involving two station types: the single-drone automatic battery swapping station (SABS) and the dual-drone automatic battery swapping station (DABS). An SABS deploys a small drone (SD), while a DABS accommodates both an SD and a large drone (LD) with greater battery capacity. The proposed problem involves determining station locations and types, assigning inspection targets to stations, and planning multi-trip drone routes. We define this integrated decision problem as the location routing problem with heterogeneous stations and drones (LRPHSD). The main contributions of this work are summarized as follows:

(a) Station with a single small drone (b) Station with a small drone and a large drone

Figure 1 Two different automatic battery swapping stations for drone operation

1. Motivated by practical inspection scenarios and the innovative design of drone stations, we propose a new variant of the location routing problem (LRP), termed LRPHSD, which simultaneously considers the heterogeneity of stations and drones. This problem also incorporates multiple real-world constraints, such as station service radius, inspection time window, and drone endurance limit, providing a more practical decision framework for infrastructure inspection.
2. We formulate the LRPHSD as a mixed-integer linear programming (MILP) model, which can be directly solved by off-the-shelf solvers to obtain optimal solutions for small-sized instances. To efficiently address large-sized instances, we develop a two-stage adaptive large neighborhood search (TSALNS) algorithm that integrates the strengths of the adaptive large neighborhood search (ALNS) for routing problems with the superiority of the two-stage algorithm for location problems. TSALNS incorporates destroy-repair operators and selection strategies tailored to heterogeneous station and drone configurations, thereby enhancing its scalability.
3. Numerical tests based on simulated data for LRPHSD and classical LRP instances demonstrate that the proposed TSALNS can produce high-quality solutions within a short computational time. Specifically, for large-sized LRPHSD instances, TSALNS can provide solutions that are comparable to or better than those obtained by Gurobi within a 3600-second time limit. For classical LRP instances, when compared with the best-known exact methods, TSALNS produces solutions with optimality gaps of around 3% for instances with more than 150 nodes, with a runtime of less than 2 minutes. In addition to simulation tests, a real-world case study is also presented. Their results jointly demonstrate the effectiveness of the heterogeneous station-drone configuration in balancing cost and task completion time.

The rest of this paper is organized as follows. Section 2 reviews related works. Section 3 defines and formulates the problem. Section 4 presents the solution method. Section 5 reports the computational results. Section 6 concludes the paper and suggests future research opportunities.

2. Literature review

This section reviews related literature on the drone routing problem, the drone inspection problem, and the LRP with drones.

2.1. Drone routing problem

We classify the existing literature on drone routing problems into two categories: truck-drone problems and drone-only problems. In truck-drone problems, trucks and drones may operate either in tandem or in parallel. Tandem systems require synchronization between trucks and drones at specific locations to launch and retrieve drones. In contrast, parallel systems allow trucks and drones to operate independently, serving customers without coordination. In drone-only problems, the service system relies solely on drones to fulfill all requests. Note that drone routing problems have been extensively studied in the literature, and the following only covers a subset of relevant works. For a more comprehensive review, interested readers can see the review paper by Chung et al. (2020).

Murray & Chu (2015) introduce two truck-drone routing problems. The first is the flying sidekick traveling salesman problem (FSTSP), where a truck performs deliveries while also serving as a mobile depot for launching and retrieving a drone. In this setting, the truck dispatches the drone from a customer site to perform a delivery, then continues its route and later reunites with the drone at another customer site. The second is the parallel drone scheduling traveling salesman problem (PDSTSP), where a truck and multiple drones work independently to serve customers. The objective in both problems is to minimize the latest return time to the depot for the truck and the drone. Building on this pioneering work, subsequent studies explore various variant problems. For example, Sacramento et al. (2019) and Wang & Sheu (2019) extend the FSTSP to a multi-vehicle case, i.e., the vehicle routing problem with drones (VRPD), where multiple trucks and drones collaborate to serve customers. Meng et al. (2024a) further extend the VRPD by allowing each drone to visit multiple customers per trip and to perform both pickup and delivery operations. Similarly, Nguyen et al. (2022) extend the PDSTSP to a multi-vehicle setting, and the objective is to minimize total transportation costs.

Studies on drone-only systems typically assume the deployment of multiple drones, each capable of visiting multiple customers in a single trip. Drone operations in such systems are often constrained by limited battery capacity; thus, some authors focus on the exact computation of energy consumption. For example, Cheng et al. (2020) address a multi-trip drone routing problem, where drone energy consumption is a nonlinear function of payload and travel distance. To accurately track energy usage, they propose two types of cuts within a branch-and-cut framework. Some authors consider employing a heterogeneous fleet of drones. For example, Chowdhury et al. (2021) study a heterogeneous drone routing problem to minimize the total inspection cost in a disaster-affected area. Cheng et al. (2024)

explore a two-period drone delivery problem with uncertainty, where varying weather conditions lead to uncertain travel times. To mitigate risks of delayed delivery, the authors apply a distributionally robust optimization method to plan drone routes.

2.2. Drone inspection problem

Some researchers focus on the applications of drones in inspection and surveillance tasks, where drones operate without payloads and can therefore visit multiple targets within a single trip. Unlike logistics operations—where it is commonly assumed that service time at each customer is negligible—inspection scenarios require drones to spend time hovering over targets to collect information. In such cases, longer service times enable more accurate data acquisition and greater operational benefits. In this section, we review drone routing problems in this area.

Baik & Valenzuela (2021) propose a coordinated truck-drone system for inspecting offshore wind power facilities. Using a cluster-then-route approach, they first group facilities into clusters, then optimize drone routes within each cluster, followed by truck routes between clusters. Kyriakakis et al. (2022) employ drones for humanitarian search and rescue operations. They utilize an approximate cellular decomposition method to partition the search region into a grid structure, providing more comprehensive spatial coverage. De la Fuente et al. (2024) propose an optimization framework for designing forest fire monitoring systems that integrate surveillance towers, aerial monitoring balloons, and drones. The model maximizes area coverage by jointly optimizing the placement of towers and balloons, as well as drone routes. Fang et al. (2023) investigate a team orienteering problem that involves planning routes for multiple drones to monitor dispersed landslides, with a particular focus on prioritizing visits to areas deemed relatively unstable.

In addition to fixed infrastructure, drones can also be deployed to monitor mobile targets, such as vehicles, vessels, and other dynamic entities. For instance, Xia et al. (2019) apply drones to monitor ships operating within emission control areas. Given the continuously changing positions of the vessels, drone travel times between vessels are also variable. To address this challenge, the authors construct a time-expanded network and develop a Lagrangian relaxation-based solution method. Zandieh et al. (2024) investigate using drones for aerial monitoring of cash-in-transit operations. In their problem, drones dynamically track the trajectories of cash transport vehicles based on the evolving security conditions of the surrounding area.

2.3. Location routing problem with drones

Our work is also closely related to the classical LRP, which involves two interdependent decisions: selecting facility locations and determining vehicle routes accordingly. In this section, we review LRP variants with drones. For a comprehensive overview of various LRP variants, readers are referred to the survey by Mara et al. (2021).

Yakıcı (2016) introduce a prize-collecting LRP in a naval intelligence context, using a homogeneous drone fleet. The drones are allocated to candidate bases to maximize the total profits of covered points of interest. Liu et al. (2019) study an extended LRP formulation that concurrently addresses depot capacity and drone flight endurance constraints. Bruni et al. (2023) consider uncertain drone travel times and propose a robust drone latency LRP aimed at minimizing customers' waiting times. Unlike models based on fixed candidate locations, Yakıcı et al. (2024) propose a dynamic coordinated deployment framework that simultaneously optimizes the positioning of mobile mothership platforms and drone routing. Their model also incorporates the selection of drone launch and recovery nodes, as well as the assignment of targets. Among existing studies, the work by Cai et al. (2025) is most closely related to ours. They propose a drone inspection scheme based on automatic battery swapping stations, incorporating both coverage and operational time constraints. However, unlike our work, they assume that all stations and drones are homogeneous, with each station supporting only a single drone. In contrast, we focus on a more flexible scheme where the stations and drones can be heterogeneous, accommodating a wider range of application scenarios.

To summarize, the aforementioned studies often assume homogeneous site or platform configurations, overlooking site heterogeneity, which is, however, an essential aspect in facility location problems (Dukkanci et al. 2023). Thus, our work aims to consider the heterogeneity of drone stations and drones to further explore the potential of drone service systems.

3. Problem definition and formulation

This section describes the problem and constructs the mathematical model.

3.1. Problem definition

The problem is defined on a directed graph $G = (V, A)$, where $V = J \cup I$ is the set of nodes. Set $J = \{1, \ldots, m\}$ denotes inspection targets, and set $I = \{m+1, \ldots, m+n\}$ represents candidate locations for placing either an SABS or a DABS. The arc set is defined as $A = \{(i, j) : i, j \in V, i \neq j\}$. Set $D = \{d_1, d_2\}$ denotes drones, where d_1 and d_2 represent the small and large drones, respectively. The fixed construction costs for an SABS and a DABS are c_s and c_l, respectively. Hell et al. (2017) indicate that, to ensure stable and reliable data transmission, the distance between each station and its drones cannot exceed a specified limit. Additionally, as the stations are battery-powered, both SABS and DABS have limited operational durations. Thus, we define (R_s, T_s) for SABS and (R_l, T_l) for DABS as their coverage radii and work endurance, respectively. The fixed costs of small and large drones are c_1 and c_2, respectively, and their unit travel cost is c_t. Note that small and large drones differ only in the energy density of their batteries, i.e., the latter has a larger energy capacity. We denote their energy capacities as Q_s and Q_l, respectively. All drones operate at a constant speed v.

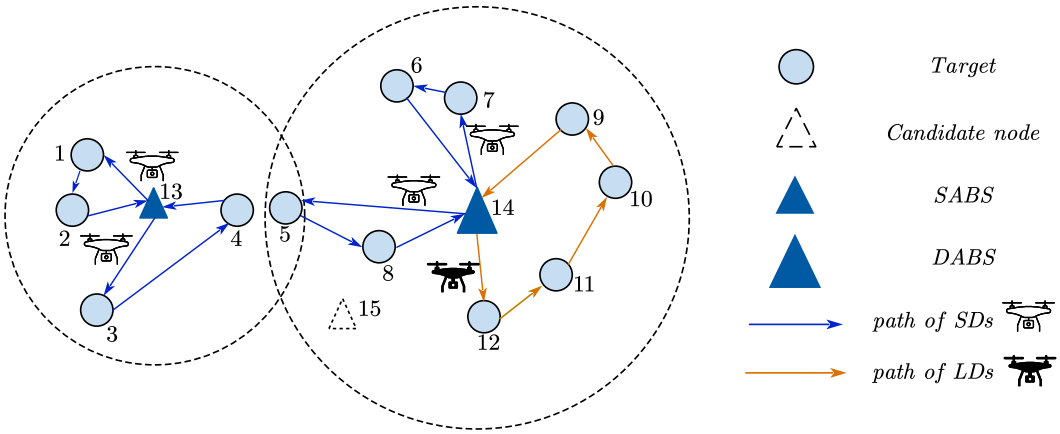

Figure 2 **An example of LRPHSD**

Each arc $(i,j) \in A$ is associated with a travel time t_{ij} and energy consumption rate p_1. Each target requires a fixed inspection time t_f, during which the drone consumes energy at a constant rate p_2.

Each drone can perform multiple trips, visiting several targets per trip. However, all visited targets must lie within the coverage radius of the drone's base station. After completing a trip, the drone must return to its launching station to swap its battery before starting the next trip. Each station is assumed to have sufficient batteries, allowing drones to swap batteries upon return. Thus, the time required for battery replacement is considered negligible.

The problem involves determining the locations and types of stations, assigning inspection targets to stations, and planning drone routes. The objective is to minimize the total cost, which includes the fixed costs of stations and drones, and the travel costs of drones. Figure 2 illustrates a feasible solution to the LRPHSD with three candidate station sites: 13, 14, and 15. An SABS is installed at node 13, deploying a single SD that performs two trips: 13–1–2–13 and 13–3–4–13. A DABS is placed at node 14, equipped with both an SD and an LD. The SD performs two trips: 14–6–7–14 and 14–5–8–14, while the LD assumes one trip: 14–12–11–10–9–14.

3.2. An MILP model

To formulate the problem, we define the following decision variables: z_i^s (z_i^l) is a binary variable equaling 1 if an SABS (a DABS) is located at site i, and 0 otherwise. y_{ij}^d is a binary variable equaling 1 if target j is served by drone d from station i, and 0 otherwise. x_{jk}^{id} is a binary variable equaling 1 if arc $(j,k) \in A$ is traversed by drone d from station i, and 0 otherwise. w_{jk}^{id} is a binary variable equaling 1 if drone d from station i visits the last target j in a trip then visits the first target k in

a subsequent trip, and 0 otherwise. e_j^{id} is a continuous variable representing the remaining battery energy of drone d from station i after visiting target j. The MILP model is constructed as follows.

$$\min \quad \sum_{i\in I}\left\{z_i^l c_l + z_i^s c_s + z_i^l(c_1+c_2) + z_i^s c_1\right\} + \sum_{i\in I}\sum_{d\in D}\sum_{(j,k)\in A} c_t x_{jk}^{id}(t_{jk}+t_f) \tag{1}$$

$$\text{s.t.} \quad \sum_{i\in I}\sum_{d\in D}\sum_{j\in V:j\neq k} x_{jk}^{id} = 1 \qquad \forall k\in J, \tag{2}$$

$$\sum_{j\in J} x_{ij}^{id} = \sum_{k\in J} x_{ki}^{id} \qquad \forall i\in I, d\in D, \tag{3}$$

$$\sum_{j\in V:j\neq k} x_{jk}^{id} = \sum_{j\in V:j\neq k} x_{kj}^{id} \qquad \forall i\in I, k\in J, d\in D, \tag{4}$$

$$\sum_{d\in D}\sum_{i\in I} y_{ij}^{d} = 1 \qquad \forall j\in J, \tag{5}$$

$$z_i^s + z_i^l \leq 1 \qquad \forall i\in I, \tag{6}$$

$$vt_{ij} y_{ij}^d \leq z_i^s R_s + z_i^l R_l \qquad \forall i\in I, j\in J, d\in D, \tag{7}$$

$$\sum_{(j,k)\in A} x_{jk}^{id}(t_{jk}+t_f) \leq T_l z_i^l + T_s z_i^s \qquad \forall i\in I, d\in D, \tag{8}$$

$$e_k^{id} + p_2 t_f + p_1 t_{jk} \leq e_j^{id} + (1-x_{jk}^{id})M_{jk} \qquad \forall i\in I, j,k\in J, d\in D, \tag{9}$$

$$p_1 t_{ji} \leq e_j^{id} + (1-x_{ji}^{id})M \qquad \forall i\in I, j\in J, d\in D, \tag{10}$$

$$2x_{jk}^{id} \leq y_{ij}^d + y_{ik}^d \qquad \forall i\in I, j,k\in V, d\in D, \tag{11}$$

$$\sum_{j\in J} y_{ij}^{d_2} \leq (1-z_i^s)\,|J| \qquad \forall i\in I, \tag{12}$$

$$\sum_{j\in J:j\neq k} w_{jk}^{id} \leq x_{ik}^{id} \qquad \forall i\in I, k\in J, d\in D, \tag{13}$$

$$\sum_{k\in J:j\neq k} w_{jk}^{id} \leq x_{ji}^{id} \qquad \forall i\in I, j\in J, d\in D, \tag{14}$$

$$\sum_{d\in D}\sum_{j\in J} x_{ij}^{id} - \sum_{d\in D}\sum_{j\in J}\sum_{k\in J} w_{jk}^{id} \leq k_l z_i^l + k_s z_i^s \qquad \forall i\in I, \tag{15}$$

$$x_{ij}^{di} = 0 \qquad \forall i,j\in I, d\in D, \tag{16}$$

$$e_i^{id_1} = Q_s \qquad \forall i\in I, \tag{17}$$

$$e_i^{id_2} = Q_l \qquad \forall i\in I, \tag{18}$$

$$x_{jk}^{id}, w_{jk}^{id} \in \{0,1\} \qquad \forall i\in I, j,k\in V, d\in D, \tag{19}$$

$$y_{ij}^d \in \{0,1\} \qquad \forall i\in I, j\in J, d\in D, \tag{20}$$

$$z_i^s, z_i^l \in \{0,1\} \qquad \forall i\in I, \tag{21}$$

$$e_j^{id_1}, e_j^{id_2} \geq 0 \qquad \forall i\in I, j\in V. \tag{22}$$

The objective function (1) minimizes the total cost. Constraints (2) guarantee that each target is visited exactly once. Constraints (3) enforce that each drone returns to its base station from which it is dispatched. Constraints (4) ensure the flow conservation at each target. Constraints (5) guarantee that each target is assigned to exactly one drone. Constraints (6) stipulate that at most

one station can be located at each candidate site. Constraints (7) limit the distance between a drone and its base station to remain within the station's service radius. Constraints (8) ensure that all inspection tasks assigned to a station are completed within the station's work duration. Constraints (9) track the battery consumption in a trip and eliminate sub-tours. We set the large constant $M_{jk} = Q_l + p_2 t_f + p_1 t_{jk}$. Constraints (10) ensure that the remaining energy after visiting the last target in a trip is sufficient for the drone to return to its base station. Constraints (11) require that a drone can only serve targets assigned to it. The large constant is set as $M = Q_l$. Constraints (12) enforce that an SABS is not equipped with a large drone, and thus no task is assigned to such an unequipped drone. Constraints (13)–(14) link decision variables $\boldsymbol{x}$ and $\boldsymbol{w}$. Constraints (15) limit the number of drones that can be deployed at each station. Constraints (16) eliminate trips between candidate sites. Constraints (17)–(18) ensure that each time a drone departs from a station to begin a new trip, it is equipped with a fully charged battery. Constraints (19)–(22) define the domains of decision variables.

4. Solution method

Although the MILP model above can be directly solved using off-the-shelf solvers such as CPLEX and Gurobi, these solvers often fail to produce satisfactory solutions for medium- and large-sized instances within a reasonable computation time due to the NP-hardness of the LRPHSD. Thus, this section introduces a TSALNS algorithm tailored to solve LRPHSD instances more effectively.

The ALNS has been widely used for solving VRPs and LRPs (Ropke & Pisinger 2006, Schiffer & Walther 2018). Unlike the VRP, the LRP involves two interdependent layers of decision-making: facility location in the first layer and routing in the second. Since location decisions inherently influence routing outcomes, two-stage algorithms are commonly adopted for solving LRPs (Tuzun & Burke 1999, Escobar et al. 2013). Accordingly, we integrate the strengths of ALNS in solving routing problems with a two-stage algorithmic framework, proposing a TSALNS algorithm.

The overall framework of TSALNS is presented in Algorithm 1. The process starts with the construction of an initial solution, as detailed in Section 4.1, followed by a two-stage optimization process. In the first stage, local search explores alternative facility location decisions, and a greedy routing heuristic generates the corresponding drone routes. In the second stage, these routes are further optimized using a combination of local search and ALNS.

4.1. Initial solution construction

Given the hierarchical structure of the LRPHSD, we adopt a composite representation for the solution: a one-dimensional list encodes the facility location decisions, while a high-dimensional list captures the associated routing decisions. Figure 3 presents an illustrative example of this representation for the solution shown in Figure 2. In this solution representation, the left column is a

Algorithm 1 Framework of TSALNS algorithm

Input: Algorithm parameters; problem parameters.
Output: Optimized solution and cost.
Generate an initial solution ▷ Section 4.1
Set current solution and the global optimal solution
for $K_{\max}$ times **do**
  Stage 1: Location optimization
  Apply local search on initial solution to obtain a new location decision ▷ Section 4.2.1
  Generate corresponding routes using *GreedyRouting* operation in Algorithm 2
  Stage 2: Routing optimization
  Local search phase:
  for $K'_{\max}$ times **do**
    Apply local search on routes ▷ Section 4.2.2
    Acceptance criteria: Accept when improved
  ALNS phase:
  for $K''_{\max}$ times **do**
    Apply ALNS on routes ▷ Section 4.3
    Acceptance criteria: Simulated annealing strategy
return Optimized solution and cost.

one-dimensional list for the facility location decisions: a value of 0 indicates that no station is established at the corresponding site, while values of 1 and 2 denote the establishment of an SABS and a DABS, respectively. The right side of the figure presents a three-dimensional list—corresponding to the three candidate facility sites—that encodes the routing decisions. Within each station, dummy node 0 is used to separate multiple trips conducted by the same drone, whereas dummy node -1 separates trips executed by different drones.

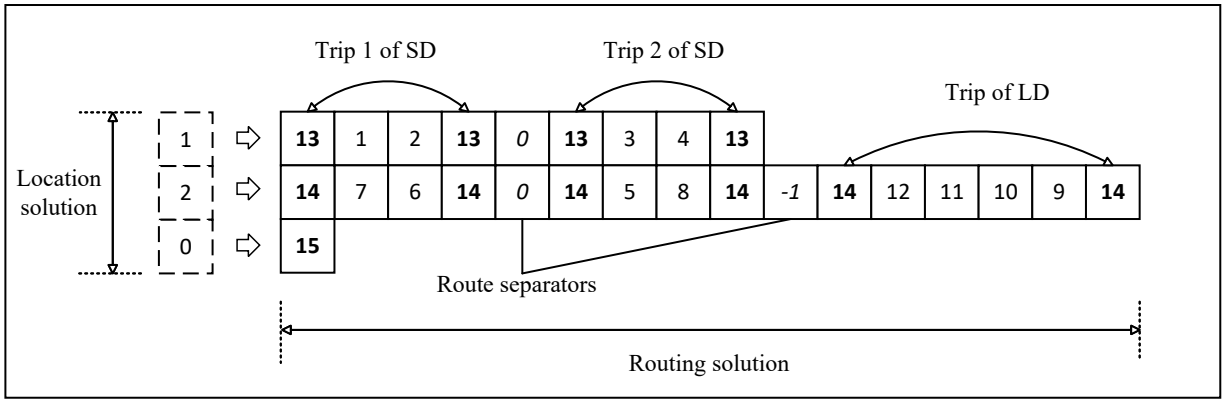

Figure 3 **Solution representation for the LRPHSD example in Figure 2**

As the quality of the initial solution influences the algorithm's overall effectiveness, we design a three-stage greedy algorithm to construct the initial solution, following the natural decision-making sequence of the problem: site selection, target assignment, and drone routing. The outline of the algorithm is presented in Algorithm 2. We begin by initializing all stations as unopened and all targets as uncovered. In each iteration, we calculate the number of uncovered targets that each unopened station can serve when configured as either an SABS or a DABS. The station with the highest coverage is selected—giving priority to an SABS if the coverage counts are equal—and its covered targets are marked as covered. This process repeats until all targets are covered. Once all targets are covered, each target is assigned to its nearest open station. Drones are then placed at each station based on the station type.

Algorithm 2 Initial solution construction

Input: Target set J, station set I, radii R_s and R_l, station durations T_s and T_l, energy capacities Q_s and Q_l.
Output: Station locations and types K, assignments A, routes P.
Initialize $K \leftarrow \{0\}^{|I|}$ ▷ All stations initially unopened
$U \leftarrow J$ ▷ Uncovered targets
while $U \neq \emptyset$ **do**
  $(i^*, t^*) \leftarrow \arg \max_{\substack{i \in I \\ t \in \{1,2\}}} |\{u \in U : d(i,u) \leq R_t\}|$ ▷ Search for station type t^*
  if no valid (i^*, t^*) exists **then**
    return $\emptyset$
  $I \leftarrow I \setminus \{i^*\}$ ▷ Remove selected station
  $K[i^*] \leftarrow t^*$; $U \leftarrow U \setminus \{u \in J : d(i^*, u) \leq R_{t^*}\}$ ▷ Set station type t^*, remove covered targets
for each $j \in J$ **do**
  $i_j \leftarrow \arg \min_{i \in I : d(i,j) \leq R_{K[i]}} d(i,j)$; $A[i] \leftarrow A[i] \cup \{j\}$ ▷ Find the closest station
for each $i \in I$ **do**
  if $K[i] = 1$ **then**
    $P_i^s \leftarrow \text{GreedyRouting}(i, A[i], Q_s, T_s)$ ▷ Construct routes by greedy algorithm
  else if $K[i] = 2$ **then**
    $P_i^l \leftarrow \text{GreedyRouting}(i, A[i], Q_l, T_l)$; $P_i^s \leftarrow \text{GreedyRouting}(i, A[i], Q_s, T_s)$
return K, A, P.

For each station, drone routes are constructed using the cheapest insertion heuristic (Solomon 1987), which sequentially inserts targets to minimize the marginal increase in total cost, while satisfying both time and energy constraints. This process continues until no additional targets can be feasibly inserted into the current trip. When a drone can no longer serve any targets within the current trip, it returns to the station to replace its battery and starts a new trip. To ensure workload balance between the two drones in a DABS, the greedy routing procedure is applied alternately to the two drones, starting with the LD.

4.2. Operators used in local search

Two local search procedures are employed in our algorithm: one in the first stage to refine the current facility configuration, and another in the second stage to improve the routing decision. The following sections detail the operators used in each process.

4.2.1. Location operators. Unlike the standard LRP, the LRPHSD requires consideration of both the location and type of stations. Thus, we design several customized location operators:

- **Upgrade operator**: Randomly select a station to upgrade (i.e., $0 \rightarrow 1$ or $1 \rightarrow 2$). Direct upgrades from 0 to 2 are prohibited.
- **Degrade operator**: Randomly select a station to downgrade (i.e., $2 \rightarrow 1$ or $1 \rightarrow 0$). Direct downgrades from 2 to 0 are not allowed.
- **Swap operator (location phase)**: Randomly select two stations and swap their types.
- **Close operator**: Randomly close a station, provided it is closable (i.e., its tasks can be reassigned to other open stations).
- **Open operator**: Randomly open a currently closed candidate station.

4.2.2. Routing operators. In each iteration, we randomly apply one of the following routing operators:

- **Swap operator 1**: Randomly swap two targets within the same trip of the same drone.
- **Swap operator 2**: Randomly swap two targets between different trips of the same drone.
- **Swap operator 3**: Randomly swap two targets between different drones at the same station.
- **Swap operator 4**: Randomly swap two targets between drones from different stations.
- **2-opt operator**: Reverse the sequence of targets between two positions within a trip.

4.3. ALNS algorithm

The ALNS has been widely applied to address routing optimization problems due to its flexible operator selection mechanism and strong search capabilities. The main idea of ALNS is to iteratively improve solutions by applying a set of destroy and repair operators, with their selection probabilities dynamically adjusted based on historical performance. In our problem, drones are subject to endurance and coverage constraints, with routing involving multiple trips serving a small number of targets each. To accommodate these features, we refine the operator selection mechanism and develop a tailored ALNS algorithm, as presented in Algorithm 3. The following sections describe the algorithm's key components.

At the start of the algorithm, all operators are allocated equal weights and scores. Selection probabilities are computed accordingly, and one destroy and one repair operator are applied. The score

Algorithm 3 Framework of ALNS algorithm

Input: Solution $\mathcal{S}'$ from local search stage.
Output: Optimized solution $\mathcal{S}^*$, best cost C^*.
$\mathcal{S} \leftarrow \mathcal{S}'$, $\mathcal{S}^* \leftarrow \mathcal{S}'$, $T \leftarrow T_0$ ▷ Initialize the current solution, the global optimal solution
for $k'' \leftarrow 1$ **to** $K''_{\max}$ **do**
  Select destroy/repair operators according to $P(w_i^d)$ and $P(w_j^r)$ ▷ Selection probability $P(w_i) = \frac{w_i}{\sum_i w_i}$
  $\mathcal{S}_{\text{destroyed}}, \mathit{RemovedNodes} \leftarrow \text{ApplyDestroyOperator}(\mathcal{S}')$ ▷ Apply destory operators in Section 4.3.1
  $\mathcal{S}'' \leftarrow \text{ApplyRepairOperator}(\mathcal{S}_{\text{destroyed}}, \mathit{RemovedNodes})$ ▷ Apply repair operators in Section 4.3.2
  Update operator scores based on performance
  Update operator weights based on their scores
  Calculate cost difference: $\Delta C \leftarrow \Phi(\mathcal{S}'') - \Phi(\mathcal{S})$
  Compute acceptance probability: ▷ Simulated annealing strategy
    $P_{\text{acc}} = 1$ if $\Delta C < 0$, and $\exp(-\Delta C/T)$ otherwise.
  if random value $\in (0,1) < P_{\text{acc}}$ **then**
    Update current solution: $\mathcal{S} \leftarrow \mathcal{S}''$
    if $\Phi(\mathcal{S}') < C^*$ **then**
      Update best solution: $\mathcal{S}^* \leftarrow \mathcal{S}''$
Update temperature: $T \leftarrow \alpha T$ ▷ Cooling down
return $\mathcal{S}^*$ and C^*.

of the selected operator is subsequently updated according to the quality of the new solution, using the update rule:

$$s^{(t+1)} = \gamma s^{(t)} + \eta,$$

where $s^{(t)}$ and $s^{(t+1)}$ are the scores of an operator at iteration t and iteration $t+1$, respectively. Parameter γ is the decay factor, which balances the recent and historical performance of the operator. A larger value of γ means that the algorithm places greater emphasis on past performance. The term η denotes the reward from performance improvement and is updated as follows:

$$\eta = \begin{cases} \xi + \tau \dfrac{\Delta C}{1 + \Delta C} & \text{if improved,} \\ -\phi & \text{otherwise,} \end{cases}$$

where ξ is the base reward for accepting a new solution. If the new solution is better than the current one, an additional reward is granted, scaled by the cost reduction ΔC and adjusted by the enhancement factor $\tau > 0$. This expression ensures that larger improvements are associated with greater rewards. If the solution is not accepted, ϕ points are deducted. Under this scoring scheme, the scores of operators gradually update as the algorithm continues. To integrate this evaluation rule into the selection process, we introduce the corresponding weight update rule:

$$w^{(t+1)} = \beta w^{(t)} + (1 - \beta)\mathbb{E}[s]$$

In iteration $t+1$, the weight of an operator, $w^{(t+1)}$, is determined by its weight in iteration t and the expected score $\mathbb{E}[s]$ over a predefined evaluation window. Parameter β balances the influence of recent and historical performance. The expected score $\mathbb{E}[s]$ is computed as the ratio of the operator's accumulated score s_{total} to the number of times h it was selected during the evaluation window, i.e., $\mathbb{E}[s] = s_{\text{total}}/h$. To guarantee that no operator is dominant or prematurely excluded from selection, we refine the weight update rule as

$$w^{t+1} = \max\left(w^{lb},\ \min(w^{ub},\ \beta w^t + (1-\beta)\frac{s_{total}}{h})\right),$$

where w^{ub} and w^{lb} are hyperparameters, determined through parameter tuning in experiments. This update rule ensures that the weight of each destroy/repair operator remains within a predefined range, such that the maximum weight does not exceed the upper bound w_{ub}, and the minimum weight is not lower than the lower bound w_{lb}. This constraint guarantees the algorithm's ability to maintain search diversity. Subsequently, the selection probability of operation i is calculated as $P(w_i) = w_i/\sum_i w_i$.

The solution acceptance criterion follows a simulated annealing (SA) strategy, which probabilistically accepts inferior solutions to escape from local optima. In the SA module, the initial temperature is denoted by T_0, and the temperature decreases according to a cooling factor α. Let T be the current temperature. Then the acceptance probability of a solution, P_{acc}, is computed as

$$P_{acc} = \begin{cases} 1 & \text{if } \Delta C < 0, \\ \exp(-\Delta C/T) & \text{otherwise.} \end{cases}$$

4.3.1. Destroy operators. In this section, we illustrate the destroy operators, which serve to disrupt the current solution to expand the search scope.

- **randomRemove operator:** Randomly remove a target from its current route.
- **greedyRemove operator:** Remove the target that yields the greatest reduction in total cost when removed from its current route.
- **probabilisticGreedyRemove operator:** To prevent the same target from being removed repeatedly—as can happen with a purely greedy strategy—a probabilistic selection mechanism is introduced. For each target, the cost reduction from its removal is calculated, ranked, and normalized to derive a probability. Targets are then selected for removal based on this probability distribution, and those with higher cost impact are more likely to be removed. This approach strikes a balance between the advantages of greedy selection and the diversity benefits of randomness.
- **similarityRemove operator:** In routing problems, two geographically closer nodes are more likely to be included together in an optimal route. Thus, removing two nearby targets simultaneously increases the likelihood of generating improved solutions (Haghi et al. 2023, Voigt 2025). Motivated by this insight and the specific characteristics of our problem, we propose a similarity-driven removal

operator as outlined in Algorithm 4. The similarity between two targets j and j' is calculated based on both spatial proximity and coverage redundancy. Let i and i' denote the drone stations that can cover targets j and j', respectively. The similarity metric is defined as:

$$\mathrm{Sim}(j,j') = \frac{\lambda_k}{\delta(j,j')} \quad \text{with} \quad k = \begin{cases} 3 & \text{if station } i \text{ can cover } j' \textit{and} \text{ station } i' \text{ can cover } j, \\ 2 & \text{if station } i \text{ can cover } j' \textit{or} \text{ station } i' \text{ can cover } j, \\ 1 & \text{otherwise,} \end{cases}$$

where $\delta(j,j')$ is the Euclidean distance between targets j and j', and $\lambda_3 > \lambda_2 > \lambda_1$ are predefined weight parameters. This metric encourages the selection of target pairs that are not only geographically close but also share similar station coverage. An illustrative example of similarity calculation is presented in Figure 4.

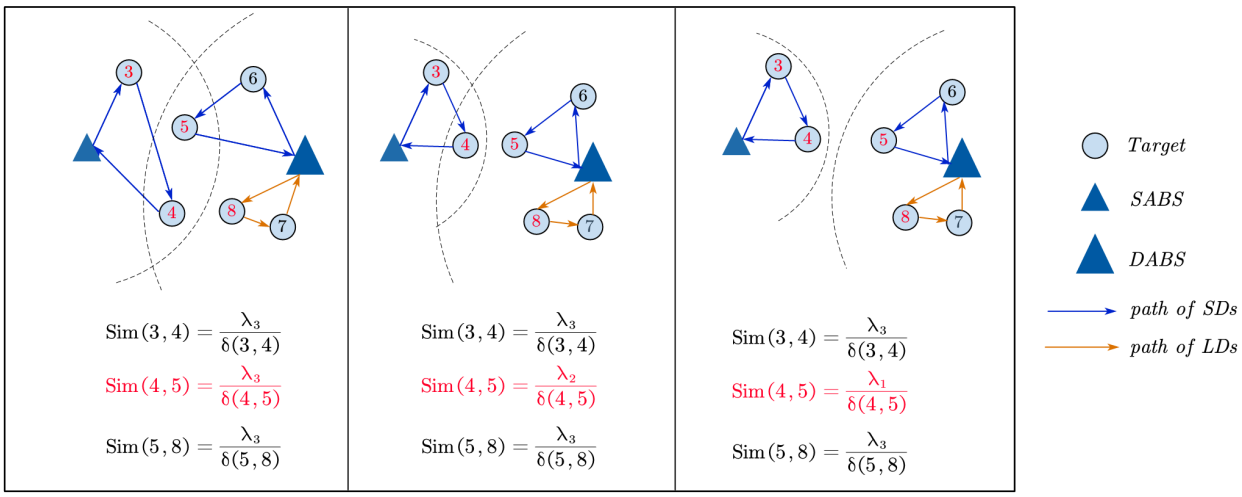

Figure 4 **An example for calculating similarity**

In Algorithm 4, pairwise similarities are computed for all targets in the current solution, generating a list of similarity scores. These scores are then normalized using min-max scaling to produce removal probabilities. Targets with higher similarity have a greater chance of being removed simultaneously. Based on these probabilities, targets are selected and removed, yielding an updated routing solution.

4.3.2. Repair operators. The repair operators reconstruct a route by strategically reinserting previously removed targets into the partially destroyed solution. Since some destroy operators remove multiple targets at once, the targets to be inserted form a set; thus, the insertion process begins with the first target in this set and proceeds sequentially. The designed repair operators are as follows:

• **cheapestInsertion operator:** Insert each removed target into the position where the increase in overall cost is minimal.

• **maxRegretInsertion operator:** The k-regret insertion strategy prioritizes targets that risk losing their best insertion opportunities if deferred, and has demonstrated effectiveness in routing

Algorithm 4 Procedure of the similarityRemove operator

Input: Routes P, station locations and types K, coverage radii R_l and R_s, station set I.
Output: Removed nodes J, modified routes P'.
for each station $i \in I$ **do**
  if $K_i = 0$ **then**
    continue
  for each sub-route $\mathcal{T} \in P_i$ (route of station i) **do**
    $L \leftarrow \emptyset$, $S \leftarrow \emptyset$ ▷ Initialize candidates set L and similarity scores set S
    for each target $j \in \mathcal{T}$ **do**
      $s_{jk} \leftarrow \sum_{j \neq k} \mathrm{Sim}(j,k)$ ▷ Calculate similarity between j and k
      if $s_j > 0$ **then**
        $L \leftarrow L \cup \{j,k\}$, $S \leftarrow S \cup \{s_{jk}\}$
    if $L = \emptyset$ **then**
      continue
    $\hat{S} \leftarrow \mathrm{Normalize}(S)$ ▷ Min-max normalization
    $j^*, k^* \leftarrow \mathrm{RouletteWheelSelect}(L, \hat{S})$
    $J \leftarrow \{j^*, k^*\} \cup J$, $P' \leftarrow P \setminus \{j^*, k^*\}$ ▷ Remove selected target
return J and P'.

optimization (Sarasola & Doerner 2020). For each target j to be inserted, the method evaluates all feasible insertion positions p, computing the corresponding insertion costs C_j^p and identifying the minimal cost C_j^{min}. The k-regret value is then calculated as

$$k\text{-Regret}_j = \sum_{p=1}^{k} (C_j^p - C_j^{\min}).$$

The value of k is problem-dependent. In our context, the limited battery capacity of drones restricts the number of targets per trip, making it appropriate to set $k = 1$. This results in a *maxRegretInsertion* operator. The algorithm selects the insertion position with the largest regret value, representing the option whose omission would lead to the greatest potential loss.

• **loadBalanceInsertion operator:** This operator evaluates each target to be inserted by exploring all feasible insertion positions across drone routes at all stations. For each position, it assesses the impact on workload distribution and system cost. Among the feasible options, a composite score is calculated to guide the decision:

$$\text{score} = \rho \frac{T_{\max} - T_{\text{after}}}{T_{\max}} + (1-\rho) \frac{1}{1 + |C_{\text{after}} - C_{\text{before}}|}, \tag{23}$$

where T_{after} and C_{after} denote the drone's task time and system cost after insertion, respectively. $T_{\max}$ is the maximum allowed work duration for a drone (T_l or T_s). The parameter ρ balances the emphasis between workload equity and cost efficiency. This strategy promotes assigning tasks to drones with lighter loads and minimal cost impact, thereby improving solution quality and operational balance.

• **prioritizedInsertion operator:** This operator prioritizes assigning targets to underutilized stations and drones. It proceeds in the following order: First, all stations are traversed to identify idle ones. If such a station exists, the target is inserted directly, and a new trip is initialized. If no idle station is available, the algorithm searches for idle drones within partially utilized stations; if found, the target is inserted and a new trip is created. If neither condition is met, a local insertion is attempted into existing routes. The insertion position is selected based on the highest score computed using Equation (23). If no feasible insertion position exists, the drone with the lightest workload is identified, and a new trip is created to accommodate the target.

5. Computational results

We first introduce the instances and the parameter settings in Section 5.1, followed by an evaluation of the TSALNS algorithm in Section 5.2. The benefits of heterogeneity are examined in Section 5.3, and the impacts of key parameters are analyzed in Section 5.4. Finally, a real-world case study is presented in Section 5.5. All experiments were conducted on a laptop equipped with a 2.4 GHz Intel(R) Core(TM) i5-10200H processor and 16 GB of memory under the Windows 10 operating system. All algorithms were implemented in the Java programming language. The MILP model presented in Section 3 was solved using Gurobi 9.5.1, with a time limit of 3600 seconds per instance. All computation times are reported in seconds. All the data used in this study are available at https://github.com/lps11735/LRPHSD.

5.1. Instances and parameters

Instances are generated by varying the numbers of potential facility locations and targets. Reflecting real-world infrastructure spacing standards, the following spatial constraints are imposed: (i) the distance between any two targets is at least 200 meters; (ii) the distance between any two candidate locations must exceed 2000 meters; and (iii) the distance between each target and its nearest candidate site cannot exceed the coverage radius of a DABS. All nodes are uniformly distributed within a square area of 14000 meters by 14000 meters.

Instances are labeled as $|J|\_|I|\_l$, where $|J|$ is the number of candidate locations, $|I|$ is the number of targets, and l is the instance index. For each $(|J|, |I|)$ combination, three instances are generated. Small-sized instances include 3–5 candidate locations and 10–35 targets, while medium- and large-sized instances consist of 6–8 candidate locations and 50–80 targets.

Since facility location is a strategic-level decision and drone routing is an operational-level decision, fixed facility costs are amortized on a daily basis. According to the pricing information from a technology company (https://www.zhifei.tech/), the fixed costs of a DABS and an SABS are set at 160000 RMB and 120000 RMB, respectively. Unless otherwise specified, all monetary values are in RMB, and all time-related parameters are measured in minutes. Assuming a 15-year service life and

350 operational days per year, the daily amortized costs are 30.4 RMB for a DABS and 22.8 RMB for an SABS. Similarly, the daily amortized costs for LDs and SDs are set at 8 RMB and 5 RMB, respectively. The unit time cost is set at 0.5 RMB per minute.

We calculate drone energy consumption based on average energy consumption rates. Following the field study by Cai et al. (2025), which identifies 15 m/s as the optimal flight speed, and referencing specifications from the DJI Matrice 350 RTK (https://www.dji.com/cn), we normalize the battery capacity of SDs to 1 (i.e., 100%). Based on the drone's maximum flight and hovering durations, we estimate energy consumption rates as 0.018 units per minute for flight and 0.020 units per minute for hovering. These rates are applied to both drone types, but SDs are assigned a lower battery capacity. According to the DJI Mavic 3E specifications, the SD battery capacity is set to 0.81. Coverage radii and operational endurance vary by station type: DABSs have a coverage radius of 5,000 meters and an endurance of 10 hours, while SABSs cover up to 4,000 meters with an endurance of 8 hours.

5.2. Algorithm performance

We evaluate the performance of TSALNS by comparing its solutions with those obtained using Gurobi on LRPHSD instances and by benchmarking against the best-known results from exact methods in the literature across multiple families of LRP instances. Since the performance of ALNS is sensitive to parameter settings, preliminary experiments were conducted to tune parameters. The final values of these parameters are summarized in Table A1 in Appendix A.

5.2.1. TSALNS performance on LRPHSD instances. Tables 1–2 report the comparative results of TSALNS and Gurobi across instances of different sizes. Columns *LB* and *UB* show the lower and upper bounds obtained by Gurobi, respectively. Columns *Best* and *Average* report the best and average results produced by TSALNS over five independent runs. Columns *Time* and *AvgTime* respectively indicate the computational times of Gurobi and TSALNS. The performance gaps between TSALNS and Gurobi are measured using $BstGAP = (Best - UB)/UB \times 100$ and $AvgGAP = (Average - UB)/UB \times 100$. TSALNS results that match or outperform Gurobi, indicated by zero or negative gaps, are marked in bold.

Table 1 shows that for instances with 3 or 4 candidate facilities and 10–20 targets, Gurobi solves all instances to optimality within 4.75 to 2514.33 seconds. TSALNS also identifies the optimal solutions for these instances but requires significantly less time—under 4 seconds. For larger instances with more facilities and targets, Gurobi yields high-quality feasible solutions within the 3600-second time limit. TSALNS produces solutions that are either better than or within 1% of Gurobi's results, requiring only 4 to 6 seconds. In terms of average performance, the performance gap is within 1.12% for all 21 instances and below 0.5% for 17 instances, highlighting the performance stability of TSALNS.

Table 1 **Performance comparison between TSALNS and Gurobi on small-sized instances**

	Gurobi			TSALNS				
Instance	LB	UB	Time	Best	Average	AvgTime	BstGAP (%)	AvgGAP (%)
3_10_1	142.58	142.58	4.75	142.58	142.58	1.21	**0.00**	**0.00**
3_10_2	141.07	141.07	11.29	11.07	141.07	1.98	**0.00**	**0.00**
3_10_3	126.85	126.85	5.82	126.85	126.85	1.32	**0.00**	**0.00**
3_15_1	173.89	173.89	18.86	173.89	173.89	2.03	**0.00**	**0.00**
3_15_2	203.13	203.13	29.12	203.13	203.65	3.67	**0.00**	0.25
3_15_3	173.63	173.63	33.52	173.63	175.23	3.94	**0.00**	0.92
3_20_1	204.67	204.67	281.08	204.67	205.13	3.79	**0.00**	0.22
3_20_2	230.94	230.94	121.45	230.94	230.94	3.16	**0.00**	**0.00**
3_20_3	223.70	223.70	2514.33	223.70	223.80	3.84	**0.00**	0.04
4_20_1	277.67	277.67	169.50	277.67	277.67	3.82	**0.00**	**0.00**
4_20_2	220.23	220.23	336.28	220.23	221.37	3.71	**0.00**	0.51
4_20_3	253.07	253.07	163.58	253.07	254.19	3.47	**0.00**	0.44
4_25_1	288.11	289.11	3602.79	289.11	290.17	4.78	**0.00**	0.37
4_25_2	294.66	300.38	3602.76	300.50	301.14	5.03	0.04	0.25
4_25_3	287.87	292.01	3607.21	294.34	295.31	4.79	0.80	1.12
4_30_1	337.80	359.49	3606.31	359.78	361.59	4.52	0.08	0.31
4_30_2	317.04	321.17	3602.81	321.22	321.44	4.11	0.02	0.05
4_30_3	336.74	339.96	3602.20	339.83	240.13	3.99	**-0.04**	0.05
5_35_1	405.27	413.71	3601.12	413.92	414.77	5.34	0.05	0.26
5_35_2	377.77	379.27	3604.92	383.02	383.46	5.03	0.98	1.09
5_35_3	405.24	414.89	3601.61	413.56	414.63	5.63	**-0.32**	**-0.06**

Table 2 **Performance comparison between TSALNS and Gurobi on medium- and large-sized instances**

	Gurobi			TSALNS				
Instance	LB	UB	Time	Best	Average	AvgTime	BstGAP (%)	AvgGAP (%)
6_50_1	515.48	531.09	3603.88	527.78	528.73	5.95	**-0.62**	**-0.45**
6_50_2	465.51	484.45	3604.60	484.45	484.45	5.55	**0.00**	**0.00**
6_50_3	530.26	546.33	3608.82	546.33	547.29	6.62	**0.00**	0.18
7_65_1	589.77	624.51	3607.23	618.65	620.89	7.22	**-0.94**	**-0.58**
7_65_2	622.83	650.27	3607.31	644.35	645.89	7.64	**-0.91**	**-0.67**
7_65_3	588.69	629.44	3607.43	627.97	628.40	8.98	**-0.23**	**-0.16**
8_80_1	675.97	725.85	3609.12	720.16	721.12	8.61	**-0.78**	**-0.65**
8_80_2	670.87	724.22	3601.32	719.35	720.33	9.23	**-0.67**	**-0.54**
8_80_3	737.02	778.10	3612.38	771.66	774.72	9.45	**-0.83**	**-0.43**

Table 2 further compares the performance of Gurobi and TSALNS on medium- and large-sized instances. TSALNS outperforms Gurobi in 7 out of 9 instances and matches its performance in the remaining 2 instances. Whereas, the computational time shows a significant difference. Specifically, Gurobi reaches the 3600-second time limit, while TSALNS only consumes a few seconds. In summary, the results presented in Tables 1 and 2 collectively demonstrate the effectiveness of the proposed TSALNS for solving the LRPHSD.

5.2.2. TSALNS performance on classical LRP instances. This section further evaluates the performance of TSALNS by applying it to classical LRP instances and comparing its solutions with the best-known results reported by exact methods (Baldacci et al. 2011, Contardo et al. 2014, Liguori et al. 2023). The experiments are conducted on five instance sets. The first two sets, $\mathcal{F}_1$ and $\mathcal{F}_2$, are proposed by dos Santos Barreto (2004) and Prins et al. (2004), respectively. The third set ($\mathcal{F}_3$), developed by Akca et al. (2008), features instances with up to 40 customers and 5 facilities. The fourth

and fifth sets, $\mathcal{F}_4$ and $\mathcal{F}_5$, are from Tuzun & Burke (1999) and Baldacci et al. (2011). All instances are publicly available at http://prodhonc.free.fr/ and https://github.com/alberto-santini/lrp-instances. A summary of the results by instance size is presented in Table 3, and the detailed results for each instance family are provided in Appendix A.

Table 3 A summary of TSALNS performance on LRP instances

				Best-known results from exact approaches		TSALNS			
Instance size	Family	#Fac	#Tar	AvgTime	OptGAP (%)	MinGAP (%)	AvgGAP (%)	MaxGAP (%)	AvgTime
Small	$\mathcal{F}_1$	2∼5	12∼36	1.67	0.00	0.00	0.00	0.00	2.36
	$\mathcal{F}_2$	5	20	0.30	0.00	0.00	0.00	0.00	1.15
	$\mathcal{F}_3$	5	30∼40	5.71	0.00	0.00	0.23	0.92	3.98
Medium to large	$\mathcal{F}_1$	5∼10	50∼100	1797.39	0.00	0.00	1.16	2.87	10.56
	$\mathcal{F}_2$	5∼10	50∼200	28049.71	0.08	0.00	1.88	4.36	10.18
	$\mathcal{F}_4$	10	100∼150	91191.54	0.00	0.18	2.33	4.56	37.92
	$\mathcal{F}_5$	14	150∼199	633427.75	0.00	3.31	3.34	3.39	77.43

In Table 3, Columns *#Fac* and *#Tar* represent the numbers of facilities and targets, respectively. Column *OptGAP* reports the average optimality gap of exact algorithms for instances within each family. Columns *MinGAP*, *AvgGAP*, and *MaxGAP* report the minimal, average, and maximal gaps relative to the best known exact solutions in the literature. For small-sized instances in sets $\mathcal{F}_1$ and $\mathcal{F}_2$, TSALNS consistently finds optimal solutions. For small-sized instances with 30–40 targets in set $\mathcal{F}_3$, the average gap is 0.23%, with a maximum of 0.92%. For medium- and large-sized instances, the average gap ranges from 1.16% to 3.34%, and the maximum gap ranges from 2.87% to 4.56%. TSALNS also demonstrates high computational efficiency, with an average runtime of less than 2 minutes for instances with more than 150 targets. These results confirm the robustness and competitiveness of TSALNS for LRPs, despite not being specifically designed for this type of problem.

5.3. Benefit of heterogeneity

This section explores the possible benefits of heterogeneous station-drone configurations. Two performance indicators are used: the total cost and the latest task completion time.

5.3.1. Benefit of drone heterogeneity. To evaluate the impact of drone heterogeneity, we relax the restrictions on drone types while keeping the station configurations unchanged. Specifically, we examine three drone deployment schemes: (i) *L-L configuration*: All stations are equipped with LDs, with two LDs assigned to each DBAS and one LD to each SABS; (ii) *S-S configuration*: All stations are equipped with SDs, with two SDs assigned to each DBAS and one SD to each SABS; (iii) *L-S configuration*: Our initial heterogeneous deployment. Experiments are conducted on instances containing 35, 50, 65, and 80 targets. For each target size, three distinct instances are generated. TSALNS is executed five times per instance to account for randomness. The results are averaged across runs and instances.

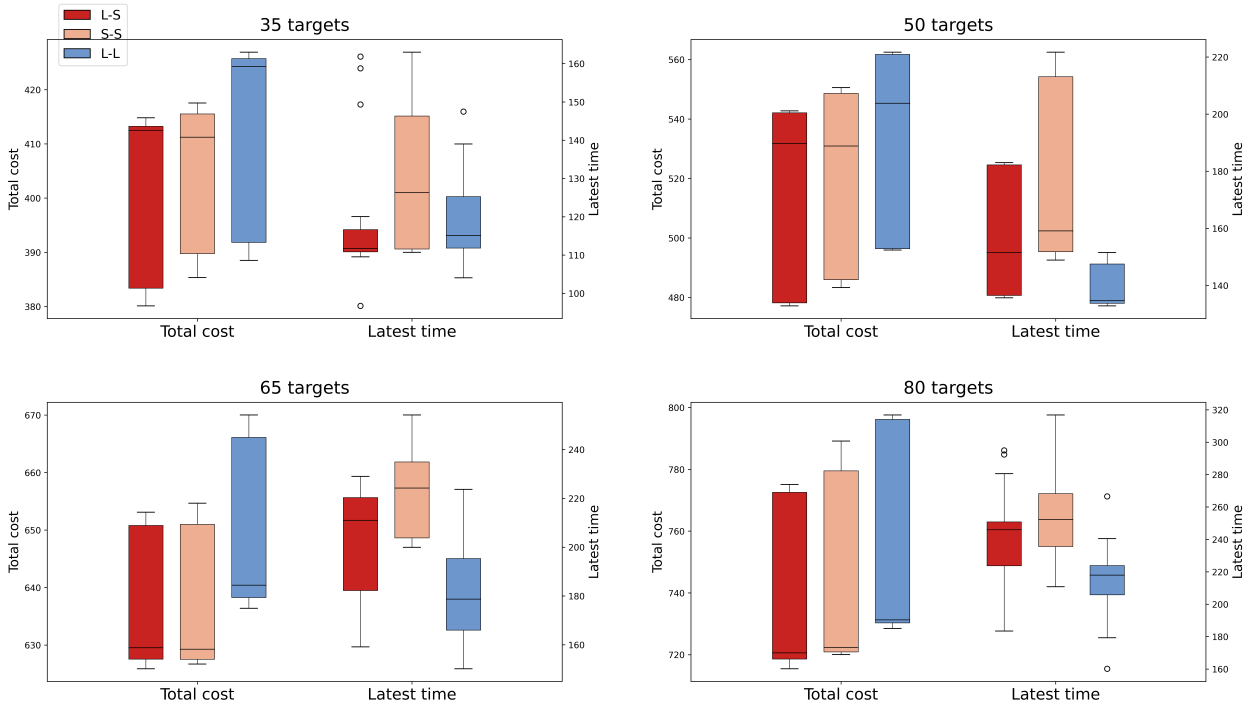

Figure 5 **Average results of three different drone configurations**

Figure 5 shows that the L-L configuration incurs the highest total cost. However, it also leads to a shorter task completion time. This advantage stems from the extended flight endurance of LDs, which enables each drone to visit more targets per trip and reduces the need for frequent returns to the base station. Thus, the L-L configuration is better suited for time-critical missions, such as infrastructure failure responses and disaster assessments, where minimizing response time takes precedence over cost efficiency. Compared to the L-S configuration, the S-S configuration offers no clear advantage in terms of either cost or time. While its total cost is comparable to that of the L-S setup, it results in longer task completion times, indicating lower overall efficiency. These results indicate that, compared to the homogeneous configurations (L-L and S-S), the heterogeneous (L-S) setup achieves a more balanced trade-off between cost and time, making it better suited for routine inspection tasks such as maintenance checks and periodic monitoring.

5.3.2. Benefit of station heterogeneity. We evaluate the impact of station heterogeneity by comparing three configurations: Only D (using only DBASs), Only S (using only SABSs), and D-S (a hybrid deployment of both). As in previous experiments, three instances are generated for each target size, and TSALNS is executed five times per instance. The average results are presented in Figure 6.

In terms of total cost, the D-S configuration consistently outperforms the other two setups, yielding the lowest cost across all scenarios. In contrast, the Only D configuration results in the highest overall cost. We note that the cost advantage of the hybrid setup primarily arises from reductions

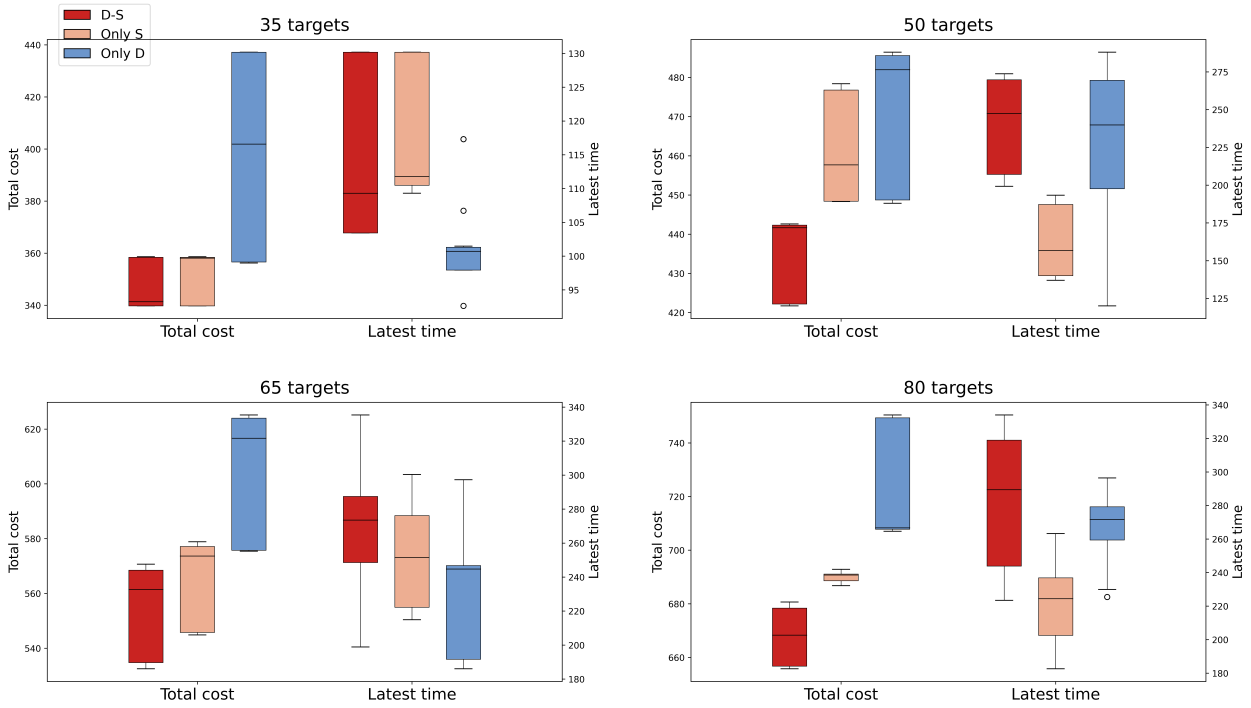

Figure 6 **Average results of three different station configurations**

in the fixed costs associated with both stations and drones. Although the only SABS configuration benefits from a lower unit fixed cost compared to the other two setups, its limited operating hours and smaller coverage radius necessitate the location of a greater number of stations, ultimately leading to higher total costs. Regarding task completion time, the Only D and Only S configurations generally outperform the hybrid configuration. However, no consistent pattern is observed, i.e., no single configuration consistently dominates in this performance metric.

Based on the results presented in Sections 5.3.1–5.3.2, we conclude that a heterogeneous configuration of stations and drones offers a more effective balance between cost and service completion time. Configurations involving dual-drone stations or large drones demonstrate superior performance in terms of task completion time, making them well-suited for time-critical applications.

5.4. Sensitivity analysis

In this section, we perform sensitivity analyses on key parameters, including the unit travel cost (c_t), service time per task (t_f), and the battery capacities of drones (Q_s and Q_l). The experiments are conducted on instances containing 20, 25, and 30 targets, and the average results are reported.

Unit travel cost. In LRPs, routing cost parameters influence the balance between fixed facility costs and variable travel costs. When routing costs are excessively high, they can strongly impact location decisions by favoring configurations that minimize travel distances. Figure 7 illustrates the impact of the unit travel cost (c_t) on the system's total cost, fixed cost, and total travel time. As c_t increases, the total cost rises, while the fixed cost and travel time remain relatively stable. This trend

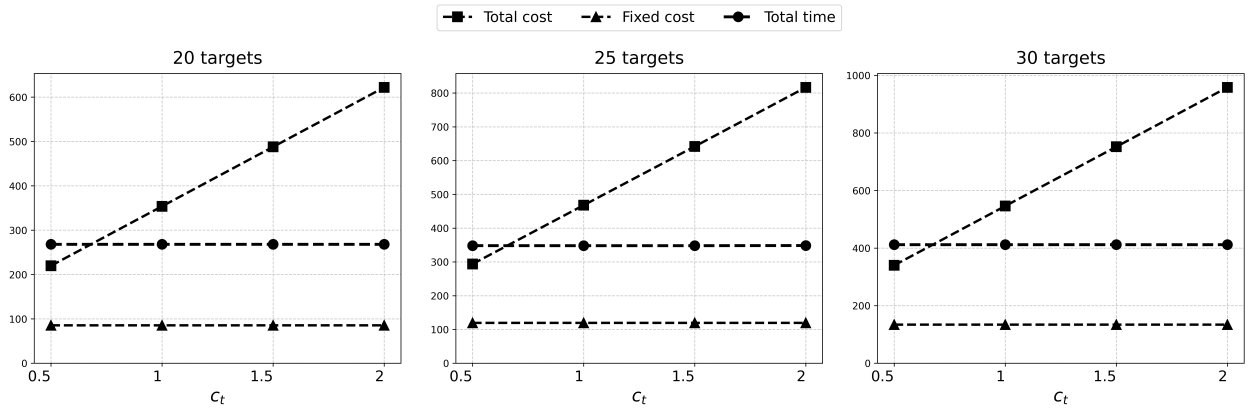

Figure 7 **The impact of unit travel cost c_t**

reflects the strategic nature of location and fleet configuration decisions, which are less sensitive to marginal changes in unit travel cost. More importantly, the results support the setting of the initial travel cost, i.e., $c_t = 0.5$.

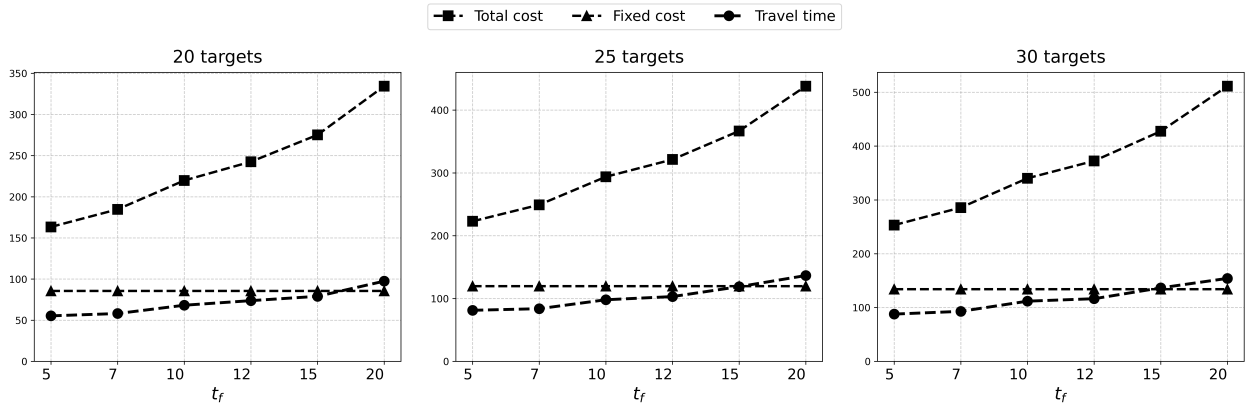

Figure 8 **The impact of the inspection time at a target**

Service time. The service time at each target varies depending on the application scenario. To reflect this variability, we vary the inspection time at each target from 5 to 20 minutes, simulating a range of scenarios—from quick aerial photography to more detailed inspection tasks. The impacts on cost and time are depicted in Figure 8. The figure shows that as inspection time increases, total travel time—and consequently, travel cost—also rises, leading to an increase in overall system cost. This is primarily due to higher energy consumption during prolonged hovering, which reduces the number of targets a drone can visit per trip and shortens its effective flight range. As a result, drones must return to stations more frequently for battery replacement, further increasing total travel time.

Drone battery capacity. This section examines the impact of battery capacity by varying the energy limits of both small and large drones. Specifically, when the energy capacity of small drones, Q_s, is fixed at 0.8, we vary the capacity of large drones, Q_l, from 0.8 to 1.2 in increments of 0.1.

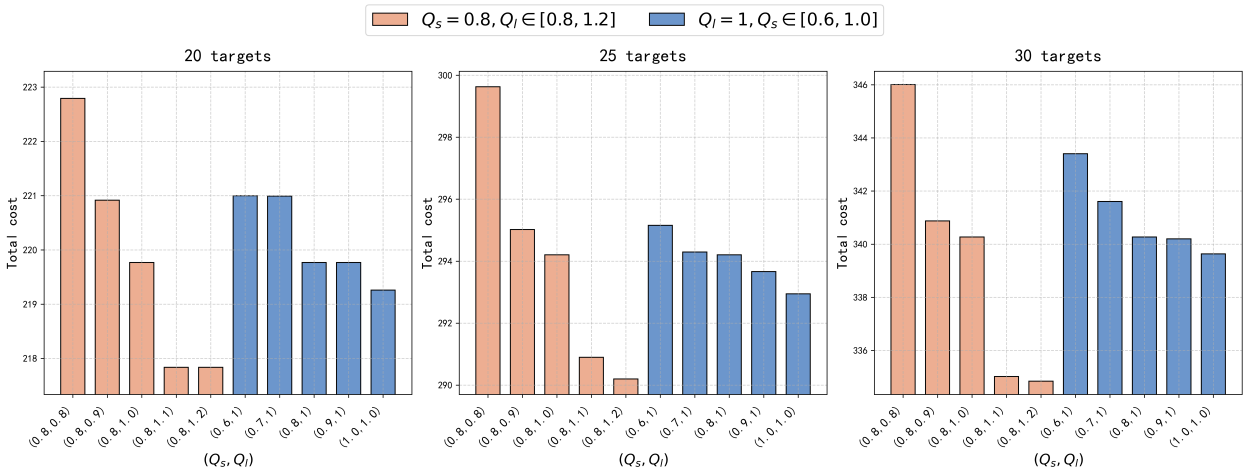

Figure 9 **The impact of drone battery capacity**

Similarly, when Q_l is fixed at 1, Q_s is adjusted from 0.6 to 1.0. The results are presented in Figure 9. The figure indicates that increasing battery capacities leads to a reduction in the system's total cost. This is because higher capacities extend the drones' flight ranges, allowing them to serve more targets within a trip, thereby reducing the number of required trips. Moreover, we observe that the impacts exhibit distinct trends: the cost reduction is more pronounced when varying the energy capacity of large drones. This is because small drones can only serve a limited number of targets per trip, and marginal increases in their capacity do not substantially extend their flight range.

5.5. A real-world case study

This section presents a real-world case study from Dongping County, Taiwan Province, China. The dataset is sourced from the Taiwan Provincial Government's open data platform (https://data.gov.tw/), where geographic information on grid infrastructure is available. A total of 30 inspection targets are selected. Following the method in Section 5.1, 4 candidate sites for drone stations are generated, and related parameters are set. The geographic distribution of the targets and candidate sites is shown in Figure 10. To evaluate the performance of different station and drone configurations, we consider three schemes: homogeneous drones (as configured in Section 5.3.1), homogeneous stations (as configured in Section 5.3.2), and heterogeneous combinations. Figure 11 reports the comparison results on total cost and task completion time.

Figure 11 shows that the heterogeneous configuration has the lowest total cost, and the L-L configuration (i.e., all stations equipped with large drones) achieves the shortest task completion time. However, the service completion time under the heterogeneous configuration is only slightly longer, indicating a favorable trade-off between cost and efficiency. These observations are consistent with those from the simulation tests. In contrast, the S-S configuration, where all stations are equipped with small drones, results in high costs and the longest service completion times. The Only

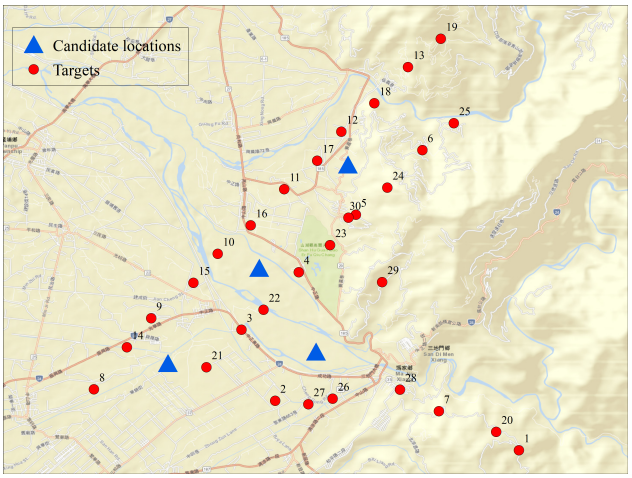

Figure 10 **Geographic distribution of inspection targets and candidate facilities in the case study**

S configuration—relying solely on SABSs—fails to generate a feasible solution due to the constrained coverage radius. The Only D configuration—using only DABSs—incurs the highest cost and does not offer any advantage in time.

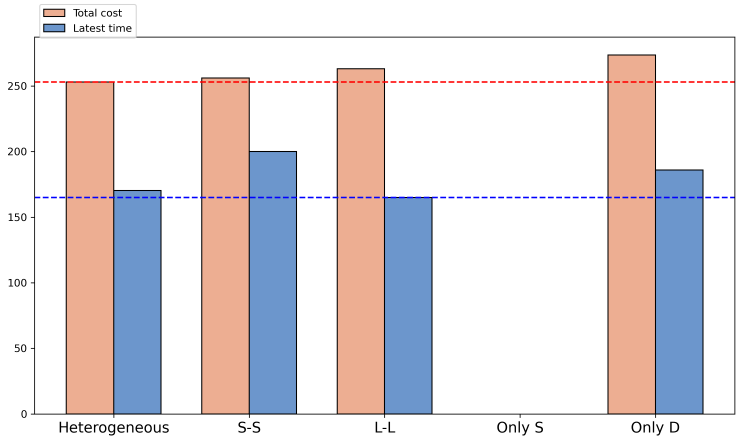

Figure 11 **The comparison results of five different station-drone configurations**

Figure 12 provides details on the location and routing decisions under the four station-drone configurations. It shows that the key difference in decisions lies in the routing part, including the number of trips and the order of visits. When all stations are equipped with large drones, i.e., the L-L setup, more customers can be included in a trip, leading to the shortest service completion time. In contrast, under the S-S configuration, fewer customers are visited per trip, resulting in a higher number of trips and the longest completion time. Comparing the heterogeneous and Only D configurations, we find that the drone resource is underutilized in the latter. Specifically, in the heterogeneous setup, a single SABS with a small drone is sufficient to serve all targets in the upper region of the figure. In contrast, the Only D configuration installs a DABS with both a large and a

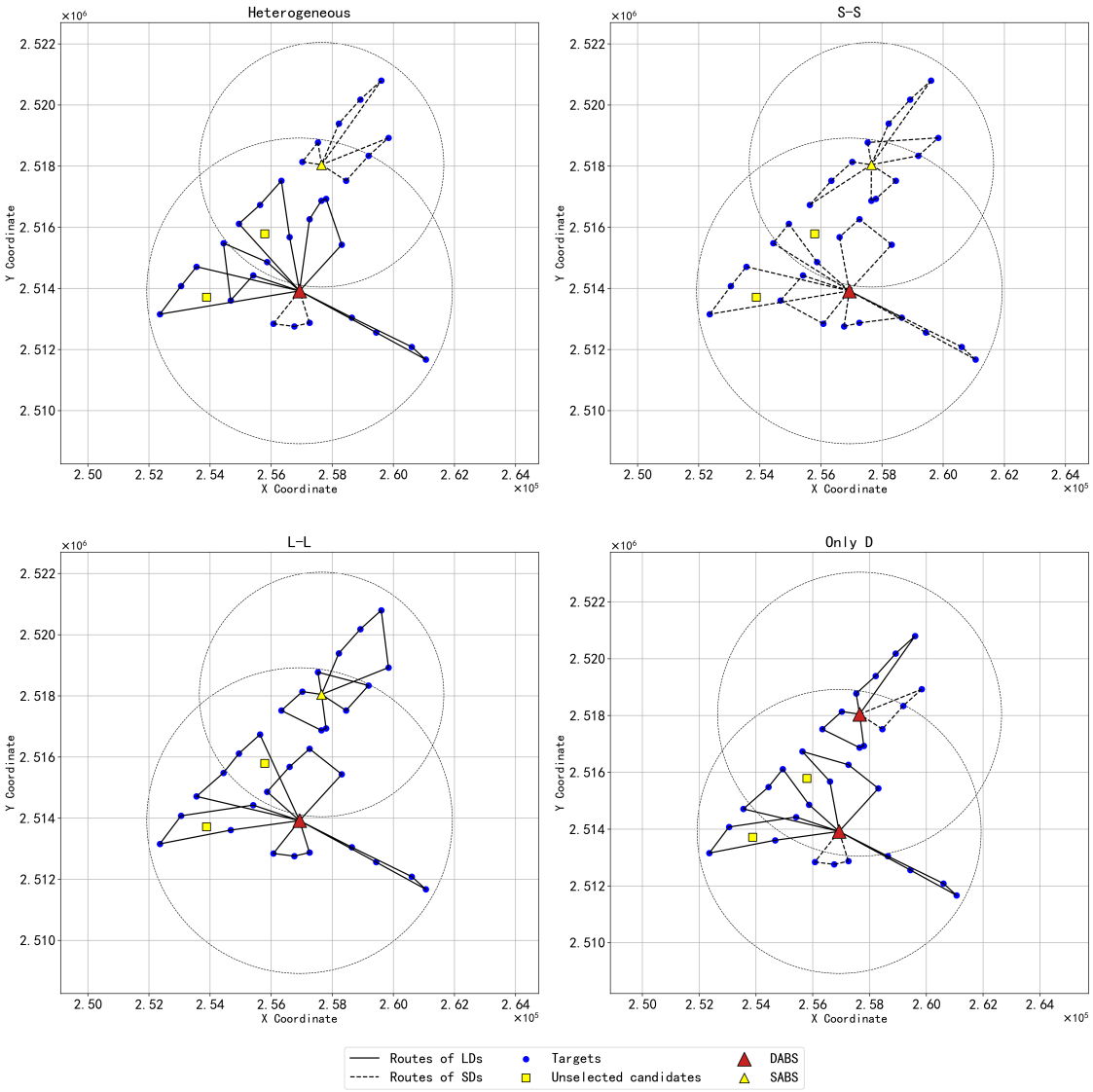

Figure 12 **Comparisons of location and routing decisions under four configurations**

small drone in the same area, resulting in redundant capacity and unnecessary expenditure, ultimately leading to the highest total cost.

6. Conclusions

Motivated by the applications of drones in infrastructure inspection, this study introduces an LRP with heterogeneous battery swapping stations and drones. Practical constraints such as the coverage radius and operational duration of both stations and drones are considered. We formulate the problem as an MILP model and develop a two-stage ALNS algorithm with tailored operators to efficiently solve large-sized instances. To evaluate the performance of the algorithm, simulation tests are first conducted using real-world station and drone data. The results show that the proposed TSALNS

algorithm can identify the optimal solutions for most small-sized instances. For medium- and large-sized instances, TSALNS can provide solutions that are comparable to or better than those obtained by Gurobi within a 3600-second time limit. Moreover, we use the TSALNS to solve the classical LRP instances and benchmark its performance with the best-known results reported by exact methods in the literature. The results show that TSALNS can also generate high-quality solutions, achieving optimality gaps of around 3% for instances with more than 150 targets, with a runtime of less than 2 minutes. In addition to simulation experiments, a real-world case study is also presented. The results collectively show that the heterogeneous station-drone configuration can achieve a better balance between cost and task completion time compared to various homogeneous configurations.

Future research can be conducted from the following aspects. First, inspection targets can be divided into different groups, each associated with varying priorities and inspection times, which is particularly relevant in post-disaster relief scenarios. Second, the deployment of drone stations can be made more flexible; for example, multiple stations can be installed at a single location. Finally, the problem can be extended to a stochastic setting to account for possible uncertainties in travel times and inspection durations.

References

Akca, Z., Berger, R., & Ralphs, T. (2008). Modeling and solving location routing and scheduling problems. In *Proceedings of the Eleventh INFORMS Computing Society Meeting* (pp. 309–330).

AOPA (2025). China's AOPA leads the transformation of low altitude economy to empower power intelligence. Accessed April 11, 2025, https://www.cpnn.com.cn/epaper/index.html/.

Baik, H., & Valenzuela, J. (2021). An optimization drone routing model for inspecting wind farms. *Soft Computing*, *25*, 2483–2498.

Baldacci, R., Mingozzi, A., & Wolfler Calvo, R. (2011). An exact method for the capacitated location-routing problem. *Operations Research*, *59*, 1284–1296.

Bruni, M. E., Khodaparasti, S., & Perboli, G. (2023). The drone latency location routing problem under uncertainty. *Transportation Research Part C: Emerging Technologies*, *156*, 104322.

Cai, L., Li, J., Wang, K., Luo, Z., & Qin, H. (2025). Optimal allocation and route design for station-based drone inspection of large-scale facilities. *Omega*, *130*, 103172.

Cheng, C., Adulyasak, Y., & Rousseau, L.-M. (2020). Drone routing with energy function: Formulation and exact algorithm. *Transportation Research Part B: Methodological*, *139*, 364–387.

Cheng, C., Adulyasak, Y., & Rousseau, L.-M. (2024). Robust drone delivery with weather information. *Manufacturing & Service Operations Management*, *26*, 1402–1421.

Chowdhury, S., Shahvari, O., Marufuzzaman, M., Li, X., & Bian, L. (2021). Drone routing and optimization for post-disaster inspection. *Computers & Industrial Engineering*, *159*, 107495.

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

Cicek, C. T., Shen, Z.-J. M., Gultekin, H., & Tavli, B. (2021). 3-d dynamic uav base station location problem. *INFORMS Journal on Computing*, *33*, 839–860.

Collins, M. (2023). Autonomous drones are inspecting and mapping infrastructure in Spain. Accessed August 30, 2023, https://www.geoweeknews.com/news/hovering-solutions-autonomous-drones-inspections-wastewater-tunnels?

Contardo, C., Cordeau, J.-F., & Gendron, B. (2014). An exact algorithm based on cut-and-column generation for the capacitated location-routing problem. *INFORMS Journal on Computing*, *26*, 88–102.

Dukkanci, O., Campbell, J. F., & Kara, B. Y. (2024). Facility location decisions for drone delivery with riding: A literature review. *Computers & Operations Research*, (p. 106672).

Dukkanci, O., Koberstein, A., & Kara, B. Y. (2023). Drones for relief logistics under uncertainty after an earthquake. *European Journal of Operational Research*, *310*, 117–132.

Escobar, J. W., Linfati, R., & Toth, P. (2013). A two-phase hybrid heuristic algorithm for the capacitated location-routing problem. *Computers & Operations Research*, *40*, 70–79.

Fang, C., Han, Z., Wang, W., & Zio, E. (2023). Routing uavs in landslides monitoring: A neural network heuristic for team orienteering with mandatory visits. *Transportation Research Part E: Logistics and Transportation Review*, *175*, 103172.

De la Fuente, R., Aguayo, M. M., & Contreras-Bolton, C. (2024). An optimization-based approach for an integrated forest fire monitoring system with multiple technologies and surveillance drones. *European Journal of Operational Research*, *313*, 435–451.

Haghi, M., Arslan, O., & Laporte, G. (2023). A location-or-routing problem with partial and decaying coverage. *Computers & Operations Research*, *149*, 106041.

Hell, P., Mezei, M., & Varga, P. J. (2017). Drone communications analysis. In *2017 IEEE 15th International Symposium on Applied Machine Intelligence and Informatics (SAMI)* (pp. 000213–000216). IEEE.

Kyriakakis, N. A., Marinaki, M., Matsatsinis, N., & Marinakis, Y. (2022). A cumulative unmanned aerial vehicle routing problem approach for humanitarian coverage path planning. *European Journal of Operational Research*, *300*, 992–1004.

Lejeune, M. A., & Ma, W. (2025). Drone-delivery network for opioid overdose: Nonlinear integer queueing-optimization models and methods. *Operations Research*, *73*, 86–108.

León, J. G. S. (2025). Spain-Portugal blackouts: what actually happened, and what can Iberia and Europe learn from it? Accessed: May 2, 2025, https://theconversation.com/spain-portugal-blackouts-what-actually-happened-and-what-can-iberia-and-europe-learn-from-it-255666.

Liguori, P. H., Mahjoub, A. R., Marques, G., Sadykov, R., & Uchoa, E. (2023). Nonrobust strong knapsack cuts for capacitated location routing and related problems. *Operations Research*, *71*, 1577–1595.

Liu, Y., Liu, Z., Shi, J., Wu, G., & Chen, C. (2019). Optimization of base location and patrol routes for unmanned aerial vehicles in border intelligence, surveillance, and reconnaissance. *Journal of Advanced Transportation*, *2019*, 9063232.

Mara, S. T. W., Kuo, R., & Asih, A. M. S. (2021). Location-routing problem: a classification of recent research. *International Transactions in Operational Research*, *28*, 2941–2983.

Meng, S., Chen, Y., & Li, D. (2024a). The multi-visit drone-assisted pickup and delivery problem with time windows. *European Journal of Operational Research*, *314*, 685–702.

Meng, Z., Zhu, N., Zhang, G., Yang, Y., Liu, Z., & Ke, G. Y. (2024b). Data-driven drone pre-positioning for traffic accident rapid assessment. *Transportation Research Part E: Logistics and Transportation Review*, *183*, 103452.

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

Nguyen, M. A., Dang, G. T.-H., Hà, M. H., & Pham, M.-T. (2022). The min-cost parallel drone scheduling vehicle routing problem. *European Journal of Operational Research*, *299*, 910–930.

Nin, C. S. (2025). Nokia leads EU project to protect critical infra with drones. Accessed June 4, 2025, https://www.rcrwireless.com/20250604/chips/nokia-drones?

Prins, C., Prodhon, C., & Calvo, R. W. (2004). Nouveaux algorithmes pour le problème de localisation et routage avec contraintes de capacité. In *MOSIM'04 (4ème Conf. Francophone de Modélisation et Simulation)*.

Richard, C. (2022). EDF deploys underwater drone to inspect offshore wind turbine foundations. Accessed 23 May, 2022, https://www.windpowermonthly.com/article/1756708/edf-deploys-underwater-drone-inspect-offshore-wind-turbine-foundations.

Ropke, S., & Pisinger, D. (2006). An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation Science*, *40*, 455–472.

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

Santos, A. H. M., de Lima, R. M., Pereira, C. R. S., Osis, R., Medeiros, G. O. S., de Queiroz, A. R., Flauzino, B. K., Cardoso, A. R. P. C., Junior, L. C., dos Santos, R. A. et al. (2019). Optimizing routing and tower spotting of electricity transmission lines: An integration of geographical data and engineering aspects into decision-making. *Electric Power Systems Research*, *176*, 105953.

dos Santos Barreto, S. (2004). *Análise e Modelização de Problemas de localização-distribuição*. Ph.D. thesis Universidade de Aveiro (Portugal).

Sarasola, B., & Doerner, K. F. (2020). Adaptive large neighborhood search for the vehicle routing problem with synchronization constraints at the delivery location. *Networks*, *75*, 64–85.

Schiffer, M., & Walther, G. (2018). An adaptive large neighborhood search for the location-routing problem with intra-route facilities. *Transportation Science*, *52*, 331–352.

Solomon, M. M. (1987). Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, *35*, 254–265.

Tuzun, D., & Burke, L. I. (1999). A two-phase tabu search approach to the location routing problem. *European Journal of Operational Research*, *116*, 87–99.

Vidović, A., Štimac, I., Mihetec, T., & Patrlj, S. (2024). Application of drones in urban areas. *Transportation Research Procedia*, *81*, 84–97.

Voigt, S. (2025). A review and ranking of operators in adaptive large neighborhood search for vehicle routing problems. *European Journal of Operational Research*, *322*, 357–375.

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

Xia, J., Wang, K., & Wang, S. (2019). Drone scheduling to monitor vessels in emission control areas. *Transportation Research Part B: Methodological*, *119*, 174–196.

Yakıcı, E. (2016). Solving location and routing problem for UAVs. *Computers & Industrial Engineering*, *102*, 294–301.

Yakıcı, E., Karatas, M., Eriskin, L., & Cicek, E. (2024). Location and routing of armed unmanned aerial vehicles and carrier platforms against mobile targets. *Computers & Operations Research*, *169*, 106727.

Zandieh, F., Ghannadpour, S. F., & Mazdeh, M. M. (2024). New integrated routing and surveillance model with drones and charging station considerations. *European Journal of Operational Research*, *313*, 527–547.

Supplementary material

Appendix A Parameter setting of TSALNS and detailed numerical results

The values of parameters used in TSALNS are summarized in Table A1.

Table A1 Parameter setting of ALNS algorithm

Category	Parameter	Description	Value
Adaptive mechanism	γ	Score decay factor	0.9
	ξ	Base reward	1
	τ	Enhancement factor	2
	ϕ	Rejection penalty	0.2
	β	Weight smoothing factor	0.8
	h	Weight update period	50
	w^{lb}	Lower weight bound	0.1
	w^{ub}	Upper weight bound	0.5
Simulated annealing	α	Cooling factor	0.95
	T_0	Initial temperature	1000
Operators	λ_3	Similarity parameter 1	0.8
	λ_2	Similarity parameter 2	0.5
	λ_1	Similarity parameter 3	0.2
	ρ	Parameter in loadBalanceInsertion operator	0.6
Stopping criteria	$K_{\max}$	The number of maximal iterations	500
	$K'_{\max}$	The number of maximal iterations in local search	500
	$K''_{\max}$	The number of maximal iterations in ALNS	5000

The columns in Tables A2–A6 are defined as follows:

- *Instance*: instance name;
- $(|J|, |I|)$: numbers of targets $(|J|)$ and candidate locations $(|I|)$;
- *UB*: best known solution (BKS) obtained by exact methods from Baldacci et al. (2011), Contardo et al. (2014), and Liguori et al. (2023);
- *Time*: CPU time (in seconds) for the exact method;
- *Best*: best objective value obtained by TSALNS over 5 runs;
- *Average*: average objective value obtained by TSALNS over 5 runs;
- *AvgTime*: average CPU time (in seconds) for TSALNS over 5 runs;
- *BstGAP*: calculated as $(\text{Best} - \text{UB})/\text{Best} \times 100$;
- *AvgGAP*: calculated as $(\text{Average} - \text{UB})/\text{Average} \times 100$.

Table A2 Results on family $\mathcal{F}_1$ instances

		BKS from exact approaches		TSALNS				
Instance	$(\|J\|,\|I\|)$	UB	Time	Best	Average	AvgTime	BstGAP (%)	AvgGAP (%)
Perl83-12x2	12,2	204.00	0.03	204.00	204.00	0.61	**0.00**	**0.00**
Gas67-21x5	21,5	424.90	0.23	424.90	424.90	0.93	**0.00**	**0.00**
Gas67-22x5	22,5	585.11	0.33	585.11	585.11	1.11	**0.00**	**0.00**
Min92-27x5	27,5	3062.02	0.31	3062.02	3062.02	1.72	**0.00**	**0.00**
Gas67-29x5	29,5	512.10	4.20	512.10	512.10	3.30	**0.00**	**0.00**
Gas67-32x5	32,5	562.22	6.52	562.22	563.43	3.51	**0.00**	0.21
Gas67-32x5-2	32,5	504.33	0.70	504.33	506.73	3.85	**0.00**	0.47
Gas67-36x5	36,5	460.37	1.04	460.37	460.37	3.89	**0.00**	**0.00**
Chr69-50x5ba	50,5	565.62	12.08	567.64	569.12	8.21	0.36	0.61
Perl83-55x15	55,15	1112.06	240.75	1112.06	1121.22	6.57	**0.00**	0.82
Chr69-75x10b	75,10	844.40	7006.42	857.99	861.86	11.52	1.58	2.03
Perl83-85x7	85,7	1622.50	177.92	1638.69	1639.68	11.41	0.99	1.05
Chr69-100x10	100,10	833.43	1549.77	858.02	861.87	15.10	2.87	3.30
Average		868.69	692.33	873.04	874.80	5.52	0.45	0.65

Table A3 Results on family $\mathcal{F}_2$ instances

		BKS from exact approaches		TSALNS				
Instance	$(\|J\|,\|I\|)$	UB	Time	Best	Average	AvgTime	BstGAP (%)	AvgGAP (%)
20-5-1a	20,5	54793	0.60	54793	54935	0.82	**0.00**	0.26
20-5-1b	20,5	39014	0.03	39014	39281	1.12	**0.00**	0.68
20-5-2a	20,5	48908	0.56	48908	48908	1.21	**0.00**	**0.00**
20-5-2b	20,5	37542	0.02	37542	37542	1.46	**0.00**	**0.00**
50-5-1	50,5	90111	17.08	90111	90396	3.81	0.00	0.21
50-5-1b	50,5	63242	478.03	63781	64321	4.29	0.85	1.68
50-5-2	50,5	88298	13.49	89012	89241	5.02	0.80	1.06
50-5-2b	50,5	67340	444.33	67698	68221	4.76	0.53	1.29
50-5-2bis	50,5	84055	29.53	84055	85977	7.12	**0.00**	2.24
50-5-2bbis	50,5	51822	7.25	52912	53993	6.02	2.06	4.02
50-5-3	50,5	86023	94.24	87380	88251	5.37	1.55	2.52
50-5-3b	50,5	61830	86.73	61830	62488	9.92	**0.00**	1.05
100-5-1	100,5	274814	446.13	282911	283212	8.73	2.86	2.97
100-5-1b	100,5	214392	47846.20	218762	220281	7.41	2.00	2.67
100-5-2	100,5	193671	56.53	201387	202812	9.01	3.83	4.51
100-5-2b	100,5	157173	12576.30	160182	162103	8.46	1.88	3.04
100-5-3	100,5	200079	123.74	204821	209763	8.30	2.32	4.62
100-5-3b	100,5	152441	469.89	153322	155811	9.36	0.57	2.16
100-10-1	100,10	289017	46434.60	296575	297611	7.93	2.55	2.89
100-10-1b	100,10	230989	3760.00	238651	239201	8.76	3.21	3.43
100-10-2	100,10	243590	2252.91	251012	251918	10.73	2.96	3.31
100-10-2b	100,10	203988	324.72	207011	208027	9.92	1.46	1.94
100-10-3	100,10	250882	12656.00	257633	259857	8.66	2.62	3.45
100-10-3b	100,10	203114	40740.00	210728	211235	8.43	3.61	3.84
200-10-1	200,10	474702	25298.00	480259	486322	15.82	1.16	2.39
200-10-1b	200,10	375177	120447.00	391271	392121	18.99	4.11	4.32
200-10-2	200,10	450468	31241.20	471018	471928	20.71	4.36	4.55
200-10-2b	200,10	374435	92495.60	380621	389927	19.35	3.05	3.97
200-10-3	200,10	472898	18738.80	482718	490732	20.71	2.75	4.34
200-10-3b	200,10	364178	45013.90	372243	377928	22.09	2.17	3.64
Average		196986	16736.45	201458	203074	9.14	1.63	2.39

Table A4 Results on family $\mathcal{F}_3$ instances

		BKS from exact approaches		TSALNS				
Instance	$(\|J\|,\|I\|)$	UB	Time	Best	Average	AvgTime	BstGAP (%)	AvgGAP (%)
r30x5a-1	30,5	819.52	2.45	819.52	823.71	3.60	**0.00**	0.51
r30x5a-2	30,5	821.46	3.72	821.46	824.61	3.52	**0.00**	0.38
r30x5a-3	30,5	702.29	0.50	706.81	713.65	3.10	0.64	1.59
r30x5b-1	30,5	880.02	4.57	880.02	880.06	3.76	**0.00**	**0.00**
r30x5b-2	30,5	825.32	1.24	825.32	829.97	3.66	**0.00**	0.56
r30x5b-3	30,5	884.58	1.23	892.81	897.81	3.12	0.92	1.47
r40x5a-1	40,5	928.10	14.67	929.25	934.33	4.86	0.12	0.67
r40x5a-2	40,5	888.42	11.88	888.42	894.33	4.24	**0.00**	0.66
r40x5a-3	40,5	947.30	11.36	952.88	956.78	4.67	0.59	0,99
r40x5b-1	40,5	1052.04	10.49	1057.66	1063.87	4.81	0.53	1.11
r40x5b-2	40,5	981.54	3.77	981.54	983.62	4.24	**0.00**	0.21
r40x5b-3	40,5	964.33	2.68	964.33	975.245	4.20	**0.00**	1.12
Average		891.25	5.71	893.34	898.17	3.98	0.23	0.77

Table A5 Results on family $\mathcal{F}_4$ instances

		BKS from exact approaches		TSALNS				
Instance	$(\|J\|,\|I\|)$	UB	Time	Best	Average	AvgTime	BstGAP (%)	AvgGAP (%)
P111112	100,10	1467.68	4467.07	1492.05	1526.17	26.01	1.63	3.83
P111212	100,10	1394.80	21548.30	1433.53	1458.32	23.85	2.70	4.36
P112112	100,10	1167.16	310.77	1189.20	1217.75	25.35	1.85	4.15
P112212	100,10	791.66	1839.47	815.52	823.89	28.11	2.93	3.91
P113112	100,10	1238.24	171702.98	1297.39	1310.39	24.98	4.56	5.51
P113212	100,10	902.26	153.23	916.77	919.42	26.15	1.58	1.87
P131112	150,10	1892.17	349797.10	1943.22	1974.55	54.89	2.63	4.17
P131212	150,10	1960.02	252301.80	2017.61	2029.20	61.98	2.94	3.53
P132112	150,10	1443.33	18604.02	1445.96	1497.21	70.01	0.18	3.60
Average		1361.92	91194.54	1394.58	1417.43	37.92	2.33	3.88

Table A6 Results on family $\mathcal{F}_5$ instances

		BKS from exact approaches		TSALNS				
instances	$(\|J\|,\|I\|)$	UB	Time	Best	Average	AvgTime	BstGAP (%)	AvgGAP (%)
M-n150x14a	150,14	1352.93	385883.92	1399.82	1415.34	67.20	3.35	4.41
M-n150×14b	150,14	1212.46	392271.48	1253.99	1271.23	69.01	3.31	4.62
M-n199x14a	199,14	1644.35	680945.21	1701.98	1731.23	83.82	3.39	5.02
M-n199x14b	199,14	1480.43	1074614.48	1531.43	1549.12	89.67	3.33	4.43
Average		1422.54	633427.75	1471.81	1491.73	77.42	3.34	4.62