Hybrid Heuristic-Based Multi-UAV Route Planning for Time-Dependent Data Collection

Pengfu Wan, Shukang Wang, Gangyan Xu, *Member, IEEE*, Yuying Long, and Runqiu Hu

***Abstract*—Unmanned aerial vehicle (UAV) has been increasingly adopted for Internet of Things (IoT) data collection in large-scale scenarios that are with less or even no network coverage. Efficient UAV route planning is a vital part of such a UAV-based data collection process, which is recognized to be complex and challenging, especially considering that the amount of data collected is dependent on UAV visit time and service time and many coupled decisions are involved. Taking these challenges into consideration, this article proposes a new hybrid heuristics-based UAV route planning method for IoT data collection. Specifically, the relationships among UAV service time, data amount, and data collection time windows of IoT devices are analyzed first, then an integrated route planning model for multiple UAVs is developed. After that, an innovative hybrid tabu search-variable neighborhood descent (HTS-VND) algorithm is developed, with six effective operators that could further improve computing efficiency and solution quality. Finally, extensive experimental case studies are conducted. The proposed method can efficiently improve the collected data amount compared to existing methods in medium-scale and large-scale scenarios.**

***Index Terms*—Data collection, hybrid heuristics, Internet of Things (IoT), route planning, time-dependent value, unmanned aerial vehicles (UAVs).**

I. Introduction

INTERNET of Things (IoT) technologies have been widely adopted for enhanced visibility, improved efficiency, lower cost, and remote operations in many fields, e.g., infrastructure monitoring, manufacturing and logistics systems, and disaster management [1], [2], [3]. Various IoT devices have been developed that can accurately sense the environment and automatically transmit sensing data to back-end systems using wired/wireless communications. In practice, to meet the timeliness requirements and avoid overriding historical sensing data in the on-board storage of IoT devices, the sensing data should be collected in a timely manner. Despite this is always realizable in production shop floors, commercial buildings, and urban infrastructure networks, it is a formidable challenge in scenarios such as transportation infrastructure monitoring [4], [5], [6]. Specifically, this work is motivated by the monitoring of highways, bridges, and high-speed railways, where numerous IoT devices are widely distributed in large, extreme, and wild areas with limited network coverage. Besides, the monitoring video collection issue that arises in wild animal monitoring further amplifies the problems of IoT data collection, where the cameras are always deployed in the virgin forest and national parks without network coverage and are difficult to reach by humans. In these scenarios, it would be economically impossible to build wired networks or base stations that connect all these devices to collect real-time sensing data. Even with access to mobile communication networks, the continuous wireless connections between IoT devices and back-end systems are difficult as they require each IoT device to have a sufficient power supply or frequent swapping of batteries, which poses extra high-maintenance costs, especially for those being sparsely deployed in extreme and wild environments.

The rapid development of unmanned aerial vehicle (UAV) provides new opportunities for such IoT data collection, which is flexible, efficient, ease-to-adopt, and with much lower cost [7], [8], [9]. Meanwhile, many pilot works have been done in UAV-assisted IoT systems to improve the efficiency, security, and quality of data transmissions through UAVs, which provide the solid technological foundation for UAV-based data collection [10], [11], [12]. Besides, to improve UAV data collection efficiency, efforts have been made on effective routing of UAVs, and various heuristic and learning-based solution methods have been developed [13], [14], [15], [16]. These methods can further minimize the UAV flight time [17], [18], [19], energy consumption [20], and maximize UAV coverage [21] and/or data collected [22]. However, considering the above mentioned scenarios of IoT data collection, existing methods are still difficult to apply directly and major challenges still exist.

First, heterogeneous IoT devices are deployed with different on-board storage spaces, data generation rates, and data transmission rates, which would lead to diverse requirements on data collection time windows, frequencies, and the service time of UAVs. Specifically, to avoid overriding historical sensing data, the data should be collected before on-board storage is full. With different data generation rates and on-board storage spaces, the data collection time windows for each IoT

Manuscript received 8 December 2023; revised 17 March 2024; accepted 9 April 2024. Date of publication 22 April 2024; date of current version 26 June 2024. This work was supported in part by the National Natural Science Foundation of China under Grant 72174042; in part by the Natural Science Foundation of Guangdong Province under Grant 2023A1515011402; in part by the Shenzhen Municipal Science and Technology Innovation Commission under Grant JCYJ20230807140406013; and in part by the Startup Fund of The Hong Kong Polytechnic University. *(Corresponding author: Gangyan Xu.)*

Pengfu Wan, Gangyan Xu, and Yuying Long are with the Department of Aeronautical and Aviation Engineering, The Hong Kong Polytechnic University, Hong Kong (e-mail: pengfu.wan@connect.polyu.hk; gangyan.xu@polyu.edu.hk; yuying.long@connect.polyu.hk).

Shukang Wang is with the Department of Mechanical Engineering, The Hong Kong Polytechnic University, Hong Kong (e-mail: shukang.wang@connect.polyu.hk).

Runqiu Hu is with the Department of Civil Engineering, The University of Hong Kong, Hong Kong (e-mail: runqiuhu@connect.hku.hk).

Digital Object Identifier 10.1109/JIOT.2024.3390732

2327-4662 © 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: Hong Kong Polytechnic University. Downloaded on February 04,2026 at 10:11:58 UTC from IEEE Xplore. Restrictions apply.

device, from the time when it accumulates minimum sensing data to the estimated time when on-board storage space is full [23], will differ. Meanwhile, the difference also makes the optimal data collection frequencies of devices different, which further complicates the scheduling of UAVs. Besides, different data transmission rates may affect the amount of data to be collected during unit time, thus making the design of UAV service time over each IoT device complex.

Second, the amount of sensing data that can be collected from each IoT device is dependent on the UAV service time. Here, the service time of UAV determines the amount of data that is collected by each visit and denotes how much storage space could be freed. Thus, compared with classical vehicle routing problems (VRPs), the problem of multiple UAV scheduling in IoT data collection is more complex and difficult as not only the sequence of visits should be determined, but also the service time at each IoT device should be decided.

Third, the scheduling of UAVs involves many coupled decisions that make it challenging to design efficient solution methods. On one hand, considering the limited endurance time of UAV, its service time over each IoT device will affect the number of IoT devices it can visit in one trip. Therefore, the tradeoff between service time and IoT devices to be visited should be considered. On the other hand, for each IoT device, the UAV service time in the current trip will affect its visit time in the next trip, thus affecting the scheduling of UAVs in the next round.

Taking the above challenges into consideration, this work aims to propose an effective multi-UAV-based IoT data collection system, develop an effective UAV route planning method to efficiently cope with various complex scenarios, and finally improve the data collection efficiency. The main contributions of this work lie in the following four aspects.

1) A flexible UAV-based IoT data collection system is proposed, which could cope with various complex scenarios with improved efficiency.
2) An integrated multi-UAV route planning model is developed that well captures the specific features in IoT data collection, including the complex coupled decisions and service time dependent collected data volumes.
3) An effective hybrid tabu search-variable neighborhood descent (HTS-VND) solution algorithm with six tailored operators is proposed for multi-UAV route planning, which could efficiently cope with various scales of scenarios.
4) Extensive experimental case studies are conducted to evaluate the performance of the proposed system in different scenarios, and verify its superiority to existing methods. It can also serve as the benchmark for further studies.

The remainder of this article is organized as follows. Section II reviews relevant works and Section III provides the overall system framework of UAV-based IoT data collection. Section IV presents the mathematical model of the multi-UAV route planning problem while Section V introduces the hybrid heuristic solution approach. Section VI gives the simulation-based case studies, evaluates the performance of the proposed algorithm, and compares the results with existing methods. Section VII concludes this article and outlines the future works.

II. Literature Review

With the technological advancement in recent years, UAV has been widely adopted in many scenarios [35], [36], [37]. Specifically, there is also an increasing trend in using UAVs for IoT data collection [38], and many efforts have been done on both ensuring data transmission quality [10], [11], [12] and improving the overall efficiency of UAV scheduling [39], [40]. Focusing on UAV route planning problem, there are also many works that have been done with various objectives, such as minimizing UAV flight time [19] and energy consumption [11], [20], maximizing the collected data amount [41], etc. Furthermore, a rich set of decisions and constraints have been studied, including the number and sequence of nodes to be visited in each UAV trip, the depart time of UAVs, and the service time of each IoT node [21], [25], [34].

Extensive solution technologies have also been developed to cope with the practical complex environment, including exact methods, learning-based methods, and heuristic-based methods. Exact methods, such as branch-and-bound [42] and Benders branch-and-cut [43], can get optimal solutions in small-scale problems, while performing worse as the scale of the problems increases. Learning-based methods can quickly get solutions even in large-scale scenarios [26], [34]. However, they are always limited by their poor generalization ability to varied scenarios and their overall performance in complex scenarios. Therefore, many heuristic-based methods were developed, e.g., genetic algorithm [44], tabu search (TS) [45], [46], large neighborhood search [14], and variable neighborhood search (VNS) [47], which could get relatively better performance in different scenarios within acceptable time. Notably, variable neighborhood descent (VND) [48], a deterministic form of the VNS, has also been developed, which replaces local search in meta-heuristic and proved to perform better in UAV route planning [49], [50].

In summary, the related works on UAV route planning for data collection can be summarized in Table I. It can be found that although researchers have considered time windows, service time, and service time dependent values for planning the route, there is still limited work considering all these factors simultaneously.

III. System Model

The scenario of UAV-based data collection can be depicted in Fig. 1. A fleet of UAVs will be dispatched to collect the data sensed by various IoT devices that are physically distributed in a large region. Following the central commands about routing decisions, each UAV will visit a subset of IoT devices, hover over each of them to collect the data, then head to another IoT device, and finally return to the depot for recharging and further data processing. The process will repeat over and over again along with the continuous generation of IoT data from these devices.

Authorized licensed use limited to: Hong Kong Polytechnic University. Downloaded on February 04,2026 at 10:11:58 UTC from IEEE Xplore. Restrictions apply.

TABLE I
SUMMARY OF RELATED WORKS

Reference	Problem Considerations				Solution
	Vehicle	Time windows	Service time	Service time dependent value	
[21]	multiple			√	Genetic Algorithm and A* search Algorithm
[22]	multiple			√	Commercial Optimization Solver
[15]	multiple			√	Deep Reinforcement Learning
[24]	multiple			√	ArcGIS VRP Solver
[25]	multiple			√	Deep Reinforcement Learning
[26]	multiple	√		√	Deep Reinforcement Learning
[27]	multiple	√			Non-dominated Sorting Genetic Algorithm II
[28]	multiple	√			Interference-aware Local search
[29]	single		√		Q-learning Algorithm
[30]	multiple		√	√	Threshold-based Assignment Policy
[31]	multiple		√		Multi-Agent Imitation Learning Based Approach
[32]	multiple		√		Hybrid Contract Net Protocol
[33]	multiple	√	√		Greedy Cost Search Algorithm
[34]	multiple		√	√	Deep Reinforcement Learning
this work	multiple	√	√	√	HTS-VND

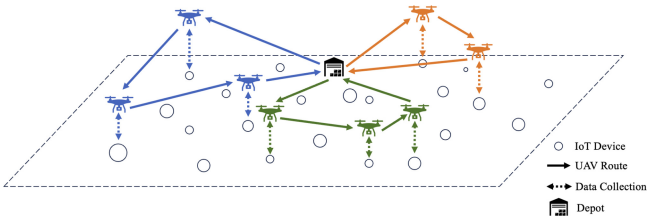

Fig. 1. Multi-UAV route planning for data collection.

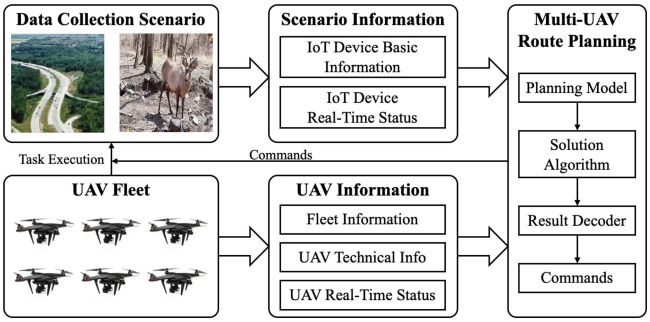

Fig. 2. System model for multi-UAV data collection.

To support the above IoT data collection process, a centralized UAV route planning system is developed, with its system model presented in Fig. 2.

The inputs for UAV route planning consist of two parts: 1) scenario information and 2) UAV information. Scenario information contains the basic information about all IoT devices, e.g., their locations, type of sensing data, on-board storage spaces, and data generation rates. Their real-time status, such as the current volume of sensed data in its on-board storage, is also included. UAV information includes: 1) the fleet information, e.g., the number of different types of UAVs, the depot location, depot size, etc; 2) the technical information of individual UAV, like endurance, speed, hover height, channel bandwidth, etc; and 3) the real-time status of UAVs, including their real-time locations, battery levels, and on-going tasks. These input information will be fed into a centralized route planning module before every round of data collection tasks.

With both scenario information and UAV information, the mathematical model of multi-UAV route planning will be generated, and solved by the route planning algorithm. The results would then be decoded as executable commands for UAVs, which include the information about the planned departure time, the sequence of nodes to be visited, and the service time at each node. These commands will then be sent to each UAV for the execution of data collection tasks. It should be noted that since the routing plan is made centrally, there is no need for individual UAV to optimize its own schedules during the execution process, but to follow commands sent from the central system.

In the following sections, the detailed design of the route planning model will be further discussed.

IV. PROBLEM MODELING

Considering the scenario where a set of n IoT devices are physically distributed within a 2-D area $X \subseteq \mathbb{R}^2$, as illustrated in Fig. 1. Here, the IoT devices are denoted as the target data nodes, each of which is numbered and represented as $i \in 1, 2, \ldots, n$. The positions of each node are represented as a tuple (x_i, y_i), where x_i and y_i are the x and y coordinates of the node in X, respectively. A set of k UAVs is adopted to collect data from these nodes, which will depart from a centralized depot (denoted as node 0), collect data from a set of IoT device nodes, and finally return to the depot for recharging.

Each UAV will visit several IoT device nodes in one trip, and hover above each node to collect the sensing data. The hover time here is defined as service time, and can be understood as the data collection time at each node. Denote the distance between the UAV and the transmitter i as d_i, the average path loss for Air-to-Ground (A2G) channel model [51], [52] is formulated as

$$\overline{PL}(d_i) = 10\alpha \log_{10}\left(\frac{d_i}{d_i^0}\right) + \overline{PL}\left(d_i^0\right) \tag{1}$$

where α is the path loss exponent and d_i^0 is a selected reference point with a known path-loss. Denote the transmit power and

Authorized licensed use limited to: Hong Kong Polytechnic University. Downloaded on February 04,2026 at 10:11:58 UTC from IEEE Xplore. Restrictions apply.

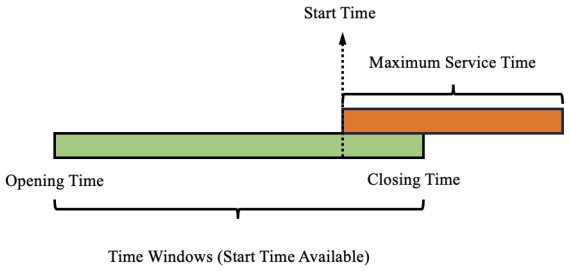

Fig. 3. Time window of each node.

noise power as P_t and P_n, the signal-to-noise ratio (SNR) is

$$\mathrm{SNR}_i = P_t - P_n - \overline{PL}(d_i). \tag{2}$$

Using the Shannon equation, the data transmission rate between the UAV and the IoT device is

$$R_i = W \log_2\left(1 + 10^{\frac{\mathrm{SNR}_i}{10}}\right) \tag{3}$$

where W is the total bandwidth of the channel. Considering that the data to be collected is temporarily stored in the onboard storage in advance, the data collection rate is equivalent to the data transmission rate. Since UAVs could hover in a fixed position during the information collection process [53], the data transmission rate can also be treated as constant during the UAV visit. Therefore, the amount of data that is collected is linearly related to the service time s_i

$$p_i = R_i \cdot s_i. \tag{4}$$

Note that each node $i \in N$ has a known expected maximum amount of stored data $p_i^{\max}$, depending on its onboard storage space. It means there is a limit on the service time s_i that after continuous data collection for a specific time, all data will be collected and no more data can be obtained afterward. Therefore, the parameter of maximum service time $s_i^{\max}$ is introduced

$$s_i^{\max} = p_i^{\max}/R_i. \tag{5}$$

It is a measure of duration that when s_i exceeds $s_i^{\max}$, no more data could be collected. $s_i^{\max}$ is introduced to restrict s_i, which could improve the efficiency of solution algorithms by effectively reducing the searching space of feasible solutions.

Meanwhile, to collect enough data and avoid the data loss caused by overwriting after collection, UAVs should provide a minimum service time $s_i^{\min}$ at node i, which is defined as follows:

$$s_i^{\min} = p_i^{\min}/R_i. \tag{6}$$

In practice, considering the availability of data and other safety concerns at each node, UAV should visit node i during a specific time window, denoted as $[o_i, c_i]$. Here, o_i is the opening time and c_i is the closing time. The UAV can only visit each data node and start to collect data within the time window. After the UAV arrives at an IoT device, it can continuously collect data until its service time exceeds $s_i^{\max}$. The relationship between time window $[o_i, c_i]$ and maximum service time $s_i^{\max}$ is illustrated in Fig. 3.

TABLE II
PROBLEM NOTATIONS

Set	
N	the set of nodes
K	the set of UAVs
Decision Variables	
$x_{i,j,k}$	a binary variable representing the route of UAV k. If UAV k departs from node i to node j, $x_{i,j,k}$=1, 0 otherwise.
s_i	the amount of service time spent at node i
Parameters	
$p_i^{\max}$	the maximum data of node i
$p_i^{\min}$	the minimum data of node i
o_i	the opening time of node i
c_i	the closing time of node i
a_i	the arrival time at node i
$s_i^{\max}$	the maximum service time of node i
$s_i^{\min}$	the minimum service time of node i
$t_{i,j}$	the travel time of UAV from node i to node j
R_i	the data collection rate of UAV i
$T_{\max}$	the endurance time of UAVs
n	the number of nodes
k	the number of UAVs
M	a large enough number

With the above analysis, this work aims to find optimal routes of UAVs, represented by a set of binary variables $x_{i,j,k}$, and their service time s_i at each visited node, to maximize the total amount of data collected from IoT devices. The notations adopted are listed in Table II, and the problem of multi-UAV route planning for IoT data collection is modeled as follows.

$$\max \sum_{i=0}^{n}\sum_{j=0}^{n}\sum_{k=1}^{k} R_i \cdot s_i \cdot x_{i,j,k}. \tag{7}$$

Subject to

$$\sum_{j=1}^{n} x_{0,j,k} = 1 \quad \forall k \in K \tag{8}$$

$$\sum_{i=1}^{n} x_{i,0,k} = 1 \quad \forall k \in K \tag{9}$$

$$\sum_{i=0,i\neq j}^{n}\sum_{k=1}^{k} x_{i,j,k} \leq 1 \quad \forall j \in N, j \neq 0 \tag{10}$$

$$\sum_{i=0,i\neq h}^{n} x_{i,h,k} = \sum_{j=1,j\neq h}^{n} x_{h,j,k} \quad \forall h \in N, k \in K \tag{11}$$

$$a_i + s_i + t_{i,j} - a_j \leq M\left(1 - x_{i,j,k}\right) \quad \forall i, j \in N, k \in K, j \neq 0 \tag{12}$$

$$o_i \cdot \sum_{j=0,j\neq i}^{n}\sum_{k=1}^{k} x_{i,j,k} \leq a_i \leq c_i \cdot \sum_{j=0,j\neq i}^{n}\sum_{k=1}^{k} x_{i,j,k} \quad \forall i \in N \tag{13}$$

$$s_i^{\min} \leq s_i \leq s_i^{\max} \quad \forall i \in N \tag{14}$$

$$a_i \leq T_{\max} - s_i - t_{i,0} \quad \forall i \in N, k \in K \tag{15}$$

$$a_0 = 0 \tag{16}$$

$$s_0 = 0 \tag{17}$$

$$x_{i,j,k} \in \{0, 1\} \quad \forall i, j \in N, k \in K \tag{18}$$

$$s_i \geq 0 \quad \forall i \in N. \tag{19}$$

Authorized licensed use limited to: Hong Kong Polytechnic University. Downloaded on February 04,2026 at 10:11:58 UTC from IEEE Xplore. Restrictions apply.

Formula (7) maximizes the total amount of data collected by all UAVs, determined by the decision variables $x_{i,j,k}$ and s_i. Constraints (8) and (9) ensure that all UAVs depart from and return to the depot (node 0). Constraint (10) ensures that each node can be visited at most once in one data collection round. Constraint (11) ensures the flow balance for each node. Constraint (12) establishes the timeline of each route. The constraint makes sure that the arrival time at node j has to be greater or equal to the sum of the traveling time from node i to j, the arrival time at node i, and the service time at node i. Constraint (13) ensures the arrival time at each node should be within the time window. Constraint (14) ensures that the service time at node i is not less than the minimum service time and does not exceed the maximum service time. Constraint (15) is the maximum endurance time constraint for UAVs. Constraints (16) and (17) ensure that the arrival time and the service time at node 0 are zero. Finally, (18) and (19) present the domain of decision variables.

V. Hybrid Heuristics-Based Solution Approach

To efficiently solve the multi-UAV route planning problem, a HTS-VND method is proposed. In this section, its overall framework will be presented first, then the detailed algorithm modules will be discussed.

A. Overall Framework

The proposed HTS-VND method follows the general TS structure and conducts the local search by VND. TS is a metaheuristic that is widely used in computationally hard optimization problems. A tabu list which maintains a record of recently visited solutions is used in TS algorithms. While searching for local optimal solutions, the TS algorithm checks the tabu list to ensure that the same solution or similar moves are not repeated, thereby avoiding cycling and promoting exploration of different areas of the search space. The length of the tabu list determines the number of prohibited solutions, which can balance the tradeoff between exploration and exploitation. Besides, the VND is a local search algorithm with the characteristic of flexibly handling various neighborhood structures, which can be easily combined with various metaheuristic algorithms. Considering the complexity of the multi-UAV route planning problem, which is a mixed integer programming (MIP) problem with both binary decision variable $x_{i,j,k}$ and continuous decision variable s_i, the proposed algorithm integrates two heuristics methods to solve it, with its overall framework shown in Fig. 4.

In the proposed HTS-VND, an initial solution is obtained first, which is represented as lists of nodes. For each UAV, the nodes to be visited and their sequence are given as a list, and the position of each node denotes the trip to which it belongs (which UAV will serve it) and the sequence to be visited in the trip, rather than the real physical positions in the scenario. Then VND is utilized to explore the neighborhood of current solutions, which is implemented for certain iterations. In each iteration, six neighborhood moves are applied to the current best solution, and the new best solution is preserved. To avoid repeatedly searching the same neighbor space and falling into local optima, a tabu list is used to store the current solution. If the tabu list fills up, the first solution in the list will be removed and the newly visited solution will be added to the back of the list. The algorithm will terminate when it reaches the maximum number of iterations specified or the maximum computation time. The overview of the proposed algorithm is presented in Algorithm 1.

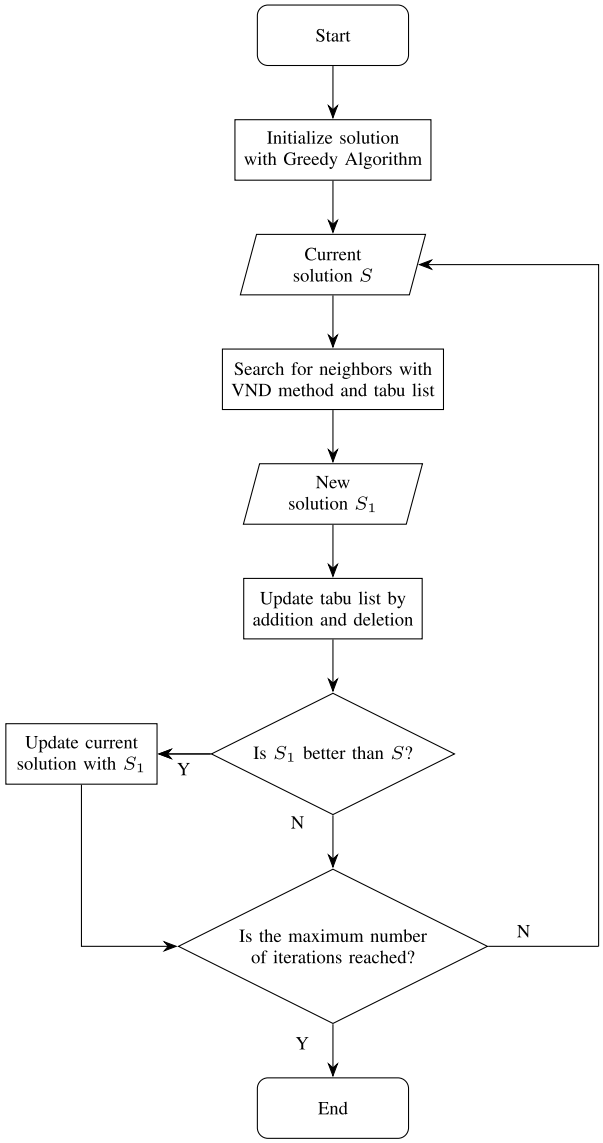

Fig. 4. Framework of the proposed algorithm.

B. Solution Initialization

An enhanced greedy-based heuristics method is used to get the initial solution for the proposed hybrid approach. It is divided into two phases: 1) preprocessing and 2) insertion, which follow the method proposed in [54]. At the beginning, all routes in the solution only have a start node. Then insert nodes into the route one by one according to the greedy strategy. Different from the regular routing problem, it has to find the neighbor that satisfies the time window constraint. Thus, for each node i, a set of nodes j will be included as the

Authorized licensed use limited to: Hong Kong Polytechnic University. Downloaded on February 04,2026 at 10:11:58 UTC from IEEE Xplore. Restrictions apply.

Algorithm 1: HTS-VND

Input: Instance data, Parameters
Output: *sol*

```
sol ← InitialSolution;
TL ← {};
for i ← 0 to Itermax do
    sol_1 ← VND(sol);
    if D(sol_1) > D(sol) then
        if sol_1 ∉ TL then
            sol ← sol_1;
            UpdateTL(sol);
        end
    else
        if sol_1 ∉ TL then
            UpdateTL(sol);
        end
    end
end
return sol;
```

candidates of its next node if

$$a_i + s_i^{\min} + t_{i,j} \le c_j. \tag{20}$$

Equation (20) ensures that when s_i is set to the minimum value $s_i^{\min}$, the UAV can arrive at j before its closing time.

Different from the method in [54], to get a high-quality solution, a node selection ratio (NSR) strategy is formulated in this work to determine which candidate should be inserted into the solution. Here, the waiting time for node j is introduced as the advancement of the arrival time a_j before the opening time o_j

$$\theta(j) \ = \max\left[0, o_j - a_j\right]. \tag{21}$$

Here, UAVs can only start to collect data after the opening time of the IoT device. Moreover, the waiting time equals zero if the UAV arrives at the node within its time window. If node j is inserted, the time consumed is then calculated as follows:

$$\eta(j) \ = t_{i,j} + \theta(j) + s_j^{\min} + t_{j,0} - t_{i,0}. \tag{22}$$

For each node in the candidate list, the time consumption is determined by (22). Then, the NSR of all candidates is calculated and sorted, based on equation

$$\mathrm{NSR}_j = \left(R_j \cdot s_j^{\min}\right)^2 / \eta(j) \tag{23}$$

where the $R_j \cdot s_j^{\min}$ represents the collected data amount in the node j. The candidate with the highest NSR will be inserted into the solution route. Furthermore, the square of the collected data amount is applied to the NSR calculation, which has been proven efficient in [55].

To balance the assignment of data collection tasks among multiple UAVs, one insertion operation is conducted for each route sequentially in each round. Meanwhile, a UAV will not be selected for insertion if it reaches its endurance limit or if all nodes have been served. Algorithm 2 provides the pseudocode for this initialization approach and Algorithm 3 gives the pseudocode to find the node with the highest NSR.

Algorithm 2: Solution Initialization

Input: The nodes set N, The UAV set K, The number of UAVs k
Output: *sol*

```
sol ← {r_i ← {0}, s_i ← {0} | i ∈ K};
U ← N     /*The set of unserved nodes*/
A ← K;
v ← 1;
while A ≠ {} do
    if v > k then
        v ← 1;
    end
    if U ≠ {} then
        c, feasible ← Find the Highest NSR(r_v, U);
        if feasible then
            r_v ← Insert(r_v, {c});
            s_v ← Insert(s_v, {s_c^min});
            U ← U\{c};
            v ← v + 1;
        else
            v ← v + 1;
            if v ∈ A then
                r_v ← Insert(r_v, {0});
                s_v ← Insert(s_v, {0});
                A ← A\{v};
            end
        end
    else
        A ← {};
    end
end
return sol;
```

C. Feasibility Check

Considering the particular time dependency in this problem, it is necessary to check the feasibility of a solution during the local search process. In the proposed algorithm, two time constraints should be checked. The first one is about the total time consumption of the UAV, which should be less than the endurance time of a UAV. Otherwise, the solution is infeasible. The second one is related to the data collection time window constraints of each IoT device. It is mainly because the neighborhood moves can alter the route and schedule, which may lead to a series of changes in arrival times at nodes and breach the time window constraints. The approach to the feasibility check is shown in Algorithm 4.

D. Variable Neighborhood Descent

VND is adopted to improve the neighborhood exploitation process, and its pseudocode is shown in Algorithm 5. In VND, various neighborhoods of a given solution are investigated systematically. Here, six local search operators (neighborhood moves) are developed, as illustrated in Fig. 5. Local search operators can change the positions or service time of nodes in the node lists to get neighborhoods of original solutions. In each iteration, six neighborhood moves are applied to the current

Authorized licensed use limited to: Hong Kong Polytechnic University. Downloaded on February 04,2026 at 10:11:58 UTC from IEEE Xplore. Restrictions apply.

Algorithm 3: Find the Highest NSR

Input: The route r_v, The unserved nodes set U
Output: h, *Feasibility*

$candidate \leftarrow \{\}$;
$NSR \leftarrow 0$;
$h \leftarrow 0$;
for $i \in U$ **do**
  Insert i into the r_v, and get the arrive time a_i based on the equation (20);
  if $a_i \le c_i$ **then**
    $candidate \leftarrow$ Insert $(candidate, \{i\})$;
    $U \leftarrow U \backslash \{i\}$;
  end
end
for $j \in candidate$ **do**
  Get $NSR(j)$ /*NSR is got from the equation (23)*/
  if $NSR(j) \ge NSR$ **then**
    $NSR \leftarrow NSR(j)$;
    $h \leftarrow j$;
  end
end
if $candidate = \{\}$ **then**
  return h, *Infeasible*;
end
$r_v \leftarrow$ Insert$\{r_v, \{h\}\}$;
$T(r_v) \leftarrow$ Get route time of r_v;
if $T(r_v) \le T_{\max}$ **then**
  return h, *Feasible*;
else
  return h, *Infeasible*;
end

Algorithm 4: Feasibility Check

Input: Solution $sol = \{r_1, r_2, \ldots, r_k\}$
Output: *Feasibility*

for $v \leftarrow 1$ *to* k **do**
  $l \leftarrow \mid r_v \mid$;
  $a_{r_v(1)} \leftarrow 0$;
  for $i \leftarrow 2$ *to* $l-1$ **do**
    $a_{r_v(i)} \leftarrow \max(a_{r_v(i-1)} + s_{r_v(i-1)} + t_{r_v(i-1),r_v(i)}, o_{r_v(i)})$;
    if $a_{r_v(i)} > c_{r_v(i)}$ **then**
      return *Infeasible*;
    end
  end
  $T(r_v) \leftarrow a_{r_v(l-1)} + s_{r_v(l-1)} + t_{r_v(l-1),r_v(l)}$;
  if $T(r_v) > T_{\max}$ **then**
    return *Infeasible*;
  end
end
return *Feasible*;

best solution and the solution with the highest data amount will be preserved. The details of these moves are further discussed as follows.

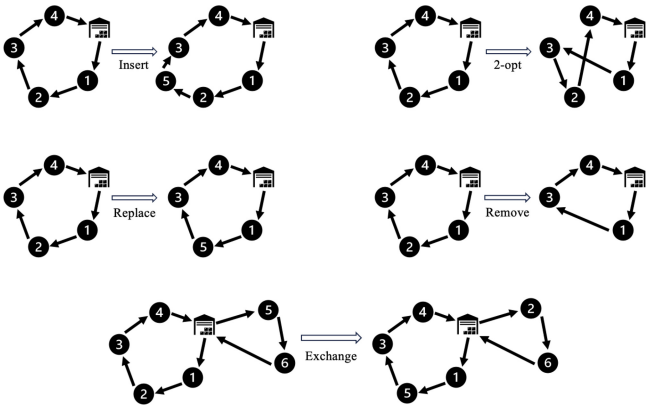

Fig. 5. Insert, 2-opt, replace, remove, and exchange.

Algorithm 5: VND

Input: Set of six neighborhood structures N_j, Solution sol, Tabu list L, Maximum number of iterations $Inter_itermax$
Output: sol_2

$j \leftarrow 1$;
$Inter_iter \leftarrow 1$;
$sol_1 = sol$;
$S \leftarrow \{\}$;
while $j \le 6$ **do**
  $sol_2 \leftarrow$ Apply N_j on sol_1;
  if $Inter_iter \ge Inter_itermax$ **then**
    $sol_2 \leftarrow$ best solution $\in S$;
    return sol_2;
  end
  if $D(sol_2) \ge D(sol_1)$ & $sol_2 \notin L$ **then**
    return sol_2;
  else
    if $sol_2 \notin L$ **then**
      $sol_1 \leftarrow sol_2$;
    else
      $sol_1 \leftarrow sol$;
    end
    if $j = 6$ **then**
      $j \leftarrow 1$;
    else
      $j \leftarrow j + 1$;
    end
    if $sol_1 \notin L$ **then**
      $S \leftarrow S \cup sol_1$;
    end
    $Inter_iter \leftarrow Inter_iter + 1$;
  end
end

1) Insert: Through the *Insert* move, a random node from the unserved list is added to a route with its minimum service time, and the insert position is randomly chosen. The randomness of selecting nodes and inserting positions can help operators systematically discover neighborhoods, thereby

Authorized licensed use limited to: Hong Kong Polytechnic University. Downloaded on February 04,2026 at 10:11:58 UTC from IEEE Xplore. Restrictions apply.

improving search efficiency. If the new solution is feasible, record the new data amount and update the unserved list.

2) 2-opt: It removes two edges and adds two different edges along the same UAV route and keeps service time unchanged. The generated route may have less waiting time and make it possible to add extra nodes. The changed nodes are chosen randomly, and feasible solutions are recorded.

3) Replace: Replace aims to increase the total data amount collected by replacing one random node in routes with an unserved node. When the replacement is feasible and more data can be collected, the move is considered successful, and this solution will be recorded.

4) Exchange: Exchange swaps the nodes on two different UAV routes, but does not modify their service time. Both exchanged routes and nodes are chosen randomly.

5) Prolong: Prolong aims to extend the service time at some nodes to increase the amount of data collected. As mentioned before, the service time of all nodes is set as the minimum value in the initial solution. However, according to (14), the possible service time is between a given minimum and maximum value. In the beginning, we randomly choose a node in the solution route and replace its service time with the maximum value. Then, recheck the new route's feasibility and output whether the new solution is feasible. If the new solution is infeasible, the feasible service time is recalculated based on the constraints about the next node's closing time and the endurance of UAVs. Ultimately, the service time of chosen nodes is updated.

6) Remove: Remove will randomly remove a node from one route in the solution to improve the possibility of other operations and to increase the diversity of neighbors.

E. Shake

To avoid local optima and inspired by [56], a shaking phase is introduced. Three operators are designed for shaking, including *Remove*, *Exchange*, and *2-opt*. Before the VDN process, one of the three moves is randomly chosen for the current solution, which can help expand the VND search space. After feasibility checking, *Shake* will generate a new solution that is not in the tabu list.

VI. Experimental Case Study

In this section, extensive numerical experiments and comparisons are given to prove the effectiveness and evaluate the performance of the proposed multi-UAV route planning method for IoT data collection.

A. Experimental Settings

The open data set generated by [57] is adopted for this experimental case study. The data set is generated for VRPs with time windows (VRPTWs) with 1000 customers, which is available online at https://www.sintef.no/projectweb/top/vrptw/1000-customers/ and has been widely adopted by researchers. In this experimental case study, the coordinate locations in those instances are adopted after changing the order of magnitude. The parameters of UAVs are referenced from [18], [58], [59] and the related hyperparameter table is shown in Table III. To verify the performance of the proposed method in different scales of problems, the number of nodes N_{region} is set from 10 to 1000 while the number of UAVs N_{UAV} ranges from 1 to 10. For each scenario, three data sets are generated and marked as data sets *A*, *B*, and *C*. Besides, each data set is numbered, as "$10-1-A$" denotes the data set *A* with ten nodes and one UAV.

TABLE III
Hyperparameter Table

Hyperparameter	Value
Channel bandwidth of UAV W	$2MHz$
Endurance of UAV	$3600Sec$
Transmit power of UAV P_t	$30dBm$
Noise power P_n	$-110dBm$
Distance between UAV and transmitter i	$40 \sim 100m$
Path loss exponent α	$2.2 \sim 2.6$
Minimum service time	$60 \sim 120s$
Maximum service time	$120 \sim 180s$

The proposed algorithm is developed using Python, and all the experiments are done on a server with 80 Intel Xeon Gold 5218R CPU @ 2.10-GHz processors and a total of 64 GB of memory.

B. Benchmark Methods

To verify the performance of the HTS-VND algorithm, five algorithms are taken for comparison, as listed below.

1) *CPLEX:* A commercial solver, IBM ILOG CPLEX Optimization Studio (CPLEX), is widely adopted for MIP model [60], [61]. CPLEX can get optimal solutions for benchmark purposes. Besides, the linearization of the model for CPLEX can be found in the Appendix.
2) *Benders Decomposition:* Benders Decomposition is a widely adopted exact method for VRPs. By decomposing an original problem into a master problem and several slave subproblems, Benders Decomposition can find the optimal solution efficiently [43].
3) *Deep Reinforcement Learning (DRL):* A state-of-the-art DRL method [62] is taken for comparison. It integrates transformer and pointer network, and has been proven to be effective in solving TOPs with time windows.
4) *Greedy Algorithm:* The Greedy Algorithm is used as the initial method for the HTS-VND algorithm, and the greedy strategy is described in Section V-B.
5) *HTS-VND-Shake:* As introduced in Section V-E, the HTS-VND algorithm with *Shake* operation is tested.

C. Performance Evaluation

A small instance with ten IoT devices and two UAVs is adopted first to show the decoded results of our algorithms, as presented in Fig. 6. The sequence of nodes to be visited and the service time at each node for both UAV-1 and UAV-2 are centrally determined before execution. Besides, the performance of HTS-VND is compared with other benchmark methods in several cases, and results show that HTS-VND is

Authorized licensed use limited to: Hong Kong Polytechnic University. Downloaded on February 04,2026 at 10:11:58 UTC from IEEE Xplore. Restrictions apply.

TABLE IV
COMPARISON BETWEEN CPLEX AND HTS-VND IN SMALL-SCALE INSTANCES

Instance	CPLEX				HTS-VND					
	LB (10^4)	UB (10^4)	Gap (%)	CPU (Sec)	Objective (10^4)		Gap (%)		CPU (Sec)	
					Avg	Max	Avg	Max	Avg	Max
10-1-A	**5.084**	5.084	0.00	0.33	5.062	**5.084**	0.43	0.00	1.94	2.44
10-1-B	**4.756**	4.756	0.00	0.89	**4.756**	**4.756**	0.00	0.00	1.35	1.74
10-1-C	**5.678**	5.678	0.00	0.55	**5.678**	**5.678**	0.00	0.00	1.65	2.00
10-5-A	**6.080**	6.080	0.00	0.84	**6.080**	**6.080**	0.00	0.00	1.43	1.82
10-5-B	**6.008**	6.008	0.00	0.81	**6.008**	**6.008**	0.00	0.00	1.27	1.65
10-5-C	**6.587**	6.587	0.00	0.61	**6.587**	**6.587**	0.00	0.00	1.27	1.66
10-10-A	**6.080**	6.080	0.00	1.59	**6.080**	**6.080**	0.00	0.00	1.97	2.49
10-10-B	**6.008**	6.008	0.00	1.44	**6.008**	**6.008**	0.00	0.00	1.66	2.04
10-10-C	**6.587**	6.587	0.00	2.13	**6.587**	**6.587**	0.00	0.00	1.69	2.05
20-1-A	**8.120**	8.120	0.00	10.5	7.786	8.063	4.11	0.70	2.20	2.79
20-1-B	**7.373**	7.373	0.00	4.8	7.352	7.354	0.28	0.26	2.10	2.56
20-1-C	**8.433**	8.433	0.00	3.32	8.031	8.379	4.77	0.64	1.55	1.86
20-5-A	**12.239**	12.239	0.00	19.34	**12.239**	**12.239**	0.00	0.00	3.05	3.27
20-5-B	**12.110**	12.110	0.00	19.30	**12.110**	**12.110**	0.00	0.00	2.35	2.69
20-5-C	**12.533**	12.533	0.00	14.94	**12.533**	**12.533**	0.17	0.00	2.16	2.43
20-10-A	**12.239**	12.239	0.00	25.84	**12.239**	**12.239**	0.00	0.00	3.16	3.45
20-10-B	**12.110**	12.110	0.00	24.88	**12.110**	**12.110**	0.00	0.00	2.68	2.87
20-10-C	**12.533**	12.533	0.00	20.61	**12.533**	**12.533**	0.00	0.00	2.54	2.98

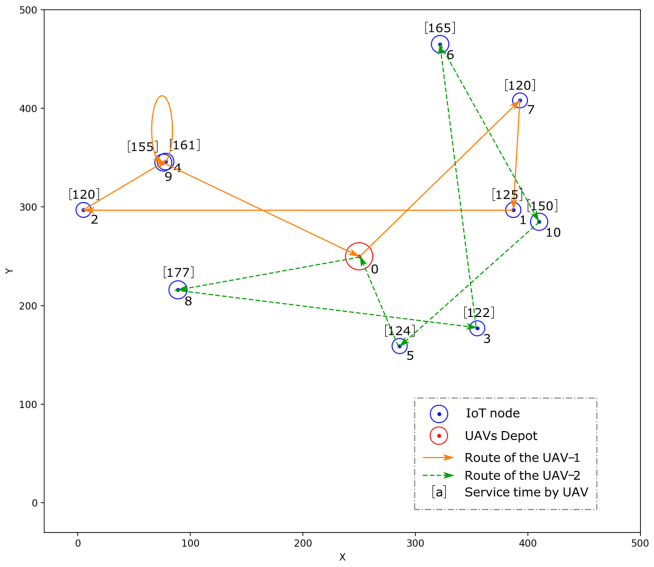

Fig. 6. Generated UAV routes.

superior in solving the multi-UAV route planning problems with different scales.

1) Comparison in Small-Scale Problems: For small-scale problems, CPLEX is adopted for comparison to verify whether the HTS-VND algorithm could generate near-optimal solutions, and the detailed results are presented in Table IV. The CPLEX solver was run once, with a CPU time limit of 3600 s. The HTS-VND algorithm was run for 100 iterations, with an internal iteration termination criterion of 100 VND iterations. To obtain a complete comparison of objectives and CPU times, HTS-VND was run ten times for each instance, and the average and maximum values were recorded. The algorithm's performance can be evaluated by calculating the percentage difference between the optimal solution and the proposed algorithm. This gap was calculated using the following equation:

$$\text{Gap} = (\text{CPLEX} - \text{HTS-VND})/\text{CPLEX} \times 100\%. \quad (24)$$

It is observed that HTS-VND can obtain optimal solutions in most small-scale instances, and the largest gap is only 4.77% (Instance "$20-1-C$"). The gap between the average and maximum values is small, which shows that HTS-VND can perform stably. In addition, our proposed algorithm can get results within 4 s, which performs much more efficiently than CPLEX in some instances.

2) Comparison in Small-Scale and Medium-Scale Problems: To further demonstrate the performance of our proposed algorithm in small-scale and medium-scale problems, the results of HTS-VND and benchmark methods are displayed in Table V. A time limit of 3600 s was set to Benders Decomposition. Similar to HTS-VND, HTS-VND-Shake was run for 100 iterations. CPLEX and Benders Decomposition both achieved optimal solutions for small-scale problems in a short time, but their performance deteriorates rapidly for medium-scale problems. Although Greedy Algorithm can solve problems quickly, the quality of achieved solutions is low. DRL method is competitive and obtains relatively good results for both small-scale and medium-scale problems within 3 s. However, the results of HTS-VND are better than those of DRL. HTS-VND-Shake also takes less time than HTS-VND, but performs worse than HTS-VND in most instances. Considering UAV route planning is usually predetermined and time for decision-making is sufficient, HTS-VND has a reasonable time consumption and shows advantages over other algorithms.

3) Comparison in Large-Scale Problems: Furthermore, we tested our algorithm on large-scale data sets consisting of 500, and 1000 instances. For all large-scale instances, CPLEX and Benders Decomposition were unable to produce results, so we only list the results of Greedy Algorithm, DRL, HTS-VND, and HTS-VND-Shake in Table VI. The iterations and internal

Authorized licensed use limited to: Hong Kong Polytechnic University. Downloaded on February 04,2026 at 10:11:58 UTC from IEEE Xplore. Restrictions apply.

TABLE V
RESULTS OF SOLUTION ALGORITHMS IN SMALL-SCALE AND MEDIUM-SCALE PROBLEMS

Instance	CPLEX		Benders Decomposition		Greedy Algorithm		DRL		HTS-VND		HTS-VND-Shake	
	Objective (10^4)	CPU (Sec)	Objective (10^4)	CPU (Sec)	Objective (10^4)	CPU (Sec)	Objective (10^4)	CPU (Sec)	Objective (10^4)	CPU (Sec)	Objective (10^4)	CPU (Sec)
10-1-A	**5.084**	0.33	**5.084**	0.81	2.481	0.01	4.308	1.28	5.062	1.94	4.999	1.56
10-1-B	**4.756**	0.89	**4.756**	1.34	1.808	0.01	4.317	1.14	**4.756**	1.35	**4.756**	1.01
10-1-C	**5.678**	0.55	**5.678**	0.71	2.644	0.01	4.621	1.17	**5.678**	1.65	**5.678**	1.32
10-5-A	**6.080**	0.84	**6.080**	1.38	3.414	0.01	**6.080**	1.20	**6.080**	1.43	**6.080**	1.35
10-5-B	**6.008**	0.81	**6.008**	2.73	3.653	0.01	**6.008**	1.18	**6.008**	1.27	**6.008**	1.35
10-5-C	**6.587**	0.61	**6.587**	2.84	3.971	0.01	**6.587**	1.23	**6.587**	1.27	**6.587**	1.14
10-10-A	**6.080**	1.59	**6.080**	1.20	3.414	0.01	**6.080**	1.10	**6.080**	1.97	**6.080**	2.06
10-10-B	**6.008**	1.44	**6.008**	1.22	3.653	0.01	**6.008**	1.14	**6.008**	1.66	**6.008**	1.55
10-10-C	**6.587**	2.13	**6.587**	1.48	3.971	0.01	**6.587**	1.11	**6.587**	1.69	**6.587**	1.54
20-1-A	**8.120**	10.50	**8.120**	21.71	4.757	0.01	5.547	1.28	7.786	2.20	7.792	2.19
20-1-B	**7.373**	4.80	**7.373**	9.66	3.351	0.01	6.191	1.36	7.352	2.10	7.077	1.28
20-1-C	**8.433**	3.32	**8.433**	28.77	4.727	0.01	6.608	1.23	8.031	1.55	8.384	1.62
20-5-A	**12.239**	19.34	**12.239**	33.55	6.941	0.01	11.687	1.82	**12.239**	3.05	**12.239**	2.69
20-5-B	**12.110**	19.30	**12.110**	26.51	6.145	0.01	11.768	1.86	**12.110**	2.35	**12.110**	1.98
20-5-C	**12.533**	14.94	**12.533**	57.70	7.672	0.01	**12.533**	2.14	**12.533**	2.16	**12.533**	2.18
20-10-A	**12.239**	25.84	**12.239**	44.49	7.252	0.01	**12.239**	1.65	**12.239**	3.16	**12.239**	2.51
20-10-B	**12.110**	24.88	**12.110**	53.58	7.327	0.01	**12.110**	1.63	**12.110**	2.68	**12.110**	2.05
20-10-C	**12.533**	20.61	**12.533**	60.79	7.672	0.01	12.532	2.01	**12.533**	2.54	**12.533**	2.03
50-1-A	10.790	3600.51	12.507	3609.91	9.617	0.02	10.370	1.22	**13.023**	6.05	12.407	4.04
50-1-B	11.237	3601.28	**13.209**	3600.36	7.475	0.01	9.446	1.17	12.674	3.38	12.815	3.23
50-1-C	10.664	3600.64	12.294	3600.49	8.490	0.01	9.029	1.17	**12.361**	3.34	12.047	2.51
50-5-A	23.555	3600.36	2.834	3601.45	18.122	0.01	27.747	1.98	**30.013**	12.43	29.940	9.45
50-5-B	18.382	3600.99	16.450	3600.76	17.126	0.01	27.029	1.77	**30.070**	11.50	29.907	10.37
50-5-C	18.578	3600.91	16.448	3600.75	18.348	0.01	27.619	1.75	**30.695**	13.95	29.902	9.50
50-10-A	10.231	3607.80	26.610	3602.04	18.122	0.01	29.737	1.64	**30.328**	12.00	**30.328**	8.20
50-10-B	16.805	3614.89	17.074	3612.70	18.018	0.01	29.971	1.21	**30.542**	11.95	30.495	7.52
50-10-C	22.152	3605.81	19.700	3600.09	18.901	0.01	30.964	1.84	**31.094**	9.09	31.079	5.97
100-1-A	10.175	3601.85	7.389	3601.72	11.144	0.02	11.223	1.43	**13.769**	5.23	13.675	4.74
100-1-B	11.285	3600.77	10.513	3601.61	11.263	0.02	11.993	1.40	**13.728**	7.43	13.554	4.63
100-1-C	6.521	3602.51	10.141	3600.37	11.519	0.02	11.646	1.38	**13.480**	6.42	13.340	4.38
100-5-A	14.874	3600.21	14.466	3603.22	33.593	0.04	42.944	1.44	**52.909**	86.22	52.176	75.87
100-5-B	17.309	3600.69	13.130	3600.99	34.404	0.04	42.506	1.62	**53.853**	96.75	53.289	73.28
100-5-C	6.510	3600.11	4.554	3601.67	36.811	0.04	41.370	1.50	**53.694**	99.09	53.062	69.35
100-10-A	6.080	3600.44	6.080	3600.37	35.785	0.03	58.482	2.78	**61.615**	54.33	61.434	36.28
100-10-B	6.008	3600.36	6.008	3600.33	36.626	0.05	58.715	1.81	**60.730**	48.80	60.349	42.82
100-10-C	6.587	3600.45	6.587	3602.42	38.563	0.03	59.431	1.78	**61.506**	41.53	61.420	30.21

TABLE VI
RESULTS OF SOLUTION ALGORITHMS IN LARGE-SCALE PROBLEMS

Instance	Greedy Algorithm		DRL		HTS-VND		HTS-VND-Shake	
	Objective (10^4)	CPU (Sec)	Objective (10^4)	CPU (Sec)	Objective (10^4)	CPU (Sec)	Objective (10^4)	CPU (Sec)
500-1-A	13.526	0.13	13.160	1.08	**14.456**	11.27	14.294	8.95
500-1-B	13.353	0.17	13.783	1.09	**14.356**	9.37	14.125	6.95
500-1-C	13.935	0.17	14.154	1.07	**14.434**	10.08	14.381	7.69
500-5-A	62.652	0.77	63.784	1.44	**67.573**	308.48	67.352	206.21
500-5-B	63.782	0.72	65.542	1.49	67.392	299.85	**67.574**	187.25
500-5-C	64.066	0.74	65.424	1.41	**68.476**	299.74	68.168	214.68
500-10-A	120.097	1.32	124.407	1.73	**131.026**	1257.97	130.480	1040.51
500-10-B	120.877	1.07	126.481	1.66	**131.329**	1635.05	131.171	1116.63
500-10-C	121.873	1.16	127.700	1.72	132.527	1556.35	**132.665**	920.83
1000-1-A	13.735	0.33	13.894	1.43	**14.610**	12.59	14.582	8.39
1000-1-B	13.723	0.34	14.020	1.20	**14.419**	10.75	14.282	7.02
1000-1-C	13.835	0.30	14.508	1.36	**14.568**	10.84	14.473	7.22
1000-5-A	67.110	1.40	66.124	1.53	70.078	342.04	**70.080**	224.31
1000-5-B	66.959	1.42	66.651	1.56	**70.173**	384.29	70.074	285.59
1000-5-C	66.053	1.40	67.953	1.68	69.470	350.31	**69.607**	253.45
1000-10-A	130.700	2.81	132.032	1.86	**136.991**	1739.29	136.808	1158.52
1000-10-B	130.315	2.75	132.684	1.90	**136.855**	1516.74	136.548	1088.73
1000-10-C	130.772	2.73	134.424	1.86	**137.426**	1676.36	136.904	1097.25

iterations for both HTS-VND and HTS-VND-Shake were the same as in the above settings. HTS-VND and HTS-VND-Shake can find better solutions than Greedy Algorithm and DRL in large-scale problems. Although HTS-VND takes more time than HTS-VND-Shake, it usually gets the best results among the four methods.

Authorized licensed use limited to: Hong Kong Polytechnic University. Downloaded on February 04,2026 at 10:11:58 UTC from IEEE Xplore. Restrictions apply.

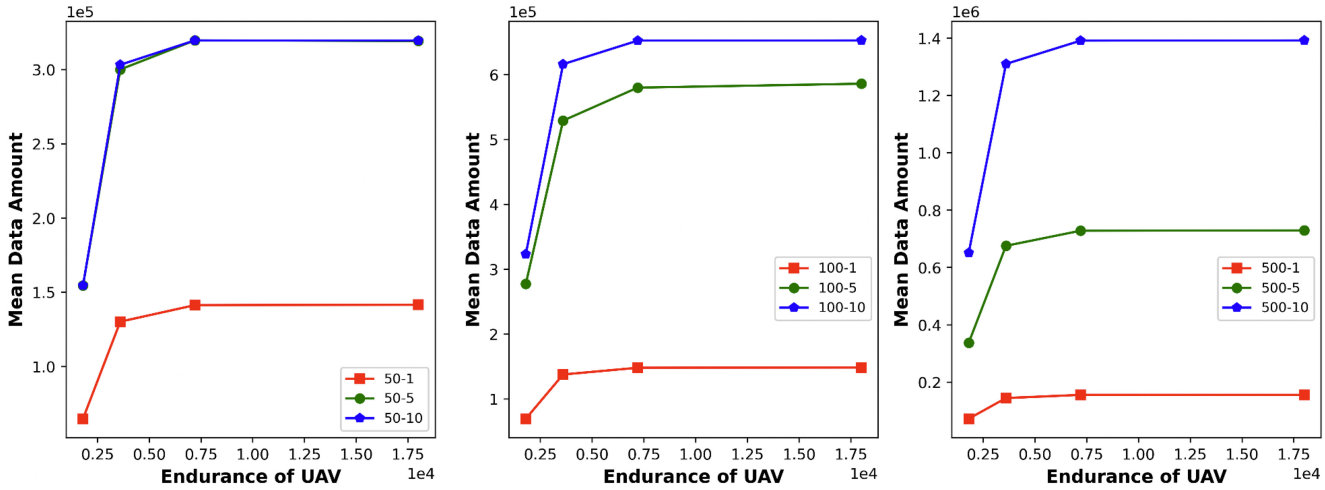

Fig. 7. Collected data amount obtained by proposed algorithm with different UAV endurance.

TABLE VII
RESULTS OF HTS-VND WITH DIFFERENT ITERATIONS

Instance	50 Iterations		100 Iterations		200 Iterations	
	Objective (10^4)	CPU (Sec)	Objective (10^4)	CPU (Sec)	Objective (10^4)	CPU (Sec)
50-1-A	12.814	2.40	13.023	6.05	**13.097**	9.27
50-5-A	29.838	5.37	30.013	12.43	**30.165**	20.49
50-10-A	**30.328**	5.32	**30.328**	12.00	**30.328**	19.98
100-1-A	13.464	2.79	13.769	5.23	**13.918**	9.04
100-5-A	51.730	34.51	52.909	86.22	**53.731**	152.08
100-10-A	61.112	18.63	61.615	54.33	**61.904**	76.15
500-1-A	14.234	5.44	14.456	11.27	**14.595**	18.59
500-5-A	66.965	127.02	67.573	308.48	**68.412**	467.90
500-10-A	129.321	690.03	131.026	1257.97	**132.775**	2499.30

TABLE VIII
RESULTS OF HTS-VND WITH DIFFERENT TABU LIST LENGTH

Instance	20 Length		50 Length		100 Length	
	Objective (10^4)	CPU (Sec)	Objective (10^4)	CPU (Sec)	Objective (10^4)	CPU (Sec)
50-1-A	**13.023**	6.05	12.940	5.50	12.861	4.63
50-5-A	30.013	12.43	**30.053**	10.60	30.033	11.07
50-10-A	**30.328**	12.00	**30.328**	10.399	**30.328**	9.80
100-1-A	13.769	5.23	**13.778**	4.86	13.357	4.68
100-5-A	52.909	86.22	52.770	66.48	**53.135**	73.38
100-10-A	61.615	54.33	61.573	38.46	**61.628**	42.45
500-1-A	14.456	11.27	**14.523**	10.30	14.391	10.19
500-5-A	67.573	308.48	**67.778**	298.35	67.447	253.69
500-10-A	131.026	1257.97	131.508	1388.97	**131.586**	1195.64

D. Impact of Important Parameters

In addition to the number of UAVs and IoT devices, other parameters of HTS-VND are also important, such as iterations, the length of tabu list, the endurance of UAVs, and the mobility of IoT devices. To determine the effect of those parameters on the performance of algorithms, we conducted a sensitivity analysis on data set *A*.

1) Iteration: We examined the effects of iterations on the algorithm, the results are shown in Table VII. As the iterations increase, so does the amount of data collected and the time spent. The time spent shows a linear upward trend, but it is hard for our algorithm to get better solutions after 100 iterations. To tradeoff between data amount and time spent, 100 or fewer iterations are enough.

2) Length of Tabu Lists: Solutions under different tabu list lengths are displayed in Table VIII. It can be found that when the problem scale increases, longer tabu lists are usually needed. Overall, the results obtained from different tabu lists are relatively stable.

3) Endurance of UAVs: Considering the impact of UAV endurance on data collection tasks, the collected data amount under different endurance is displayed in Fig. 7. Longer UAV endurance can bring larger data amount. However, when endurance reaches 2 h, the increment brought about by UAV endurance is negligible, as the collected data amount is also affected by time windows and available service time. In real scenarios, UAVs with suitable endurance rather than ultrahigh endurance should be selected.

4) Mobility of IoT Devices: In real scenarios, the locations of IoT devices are not always unchanged. To test the performance of HTS-VND under various IoT device distributions, 10 instances with different node locations are randomly generated for each scenario and boxplots for data amount and time spent are drawn in Figs. 8 and 9. It can be found that the objective values obtained by HTS-VND with different IoT device distributions are stable. Although there are fluctuations in the solution time for large-scale instances (instances with 500 nodes), HTS-VND is efficient enough and can obtain solutions within 30 min.

VII. CONCLUSION

Considering the time-dependent data amount in UAV-based IoT data collection, this work proposes a hybrid heuristic-based approach for efficient multi-UAV route planning. The approach integrates the strengths of TS and VND algorithms that could efficiently search for high-quality solutions. The proposed algorithm HTS-VND is evaluated on various instances of different scales and compared with the state-of-the-art methods. The results demonstrate that the proposed algorithm has advantages over other methods in different scenarios. Furthermore, sensitivity analysis is conducted to

Authorized licensed use limited to: Hong Kong Polytechnic University. Downloaded on February 04,2026 at 10:11:58 UTC from IEEE Xplore. Restrictions apply.

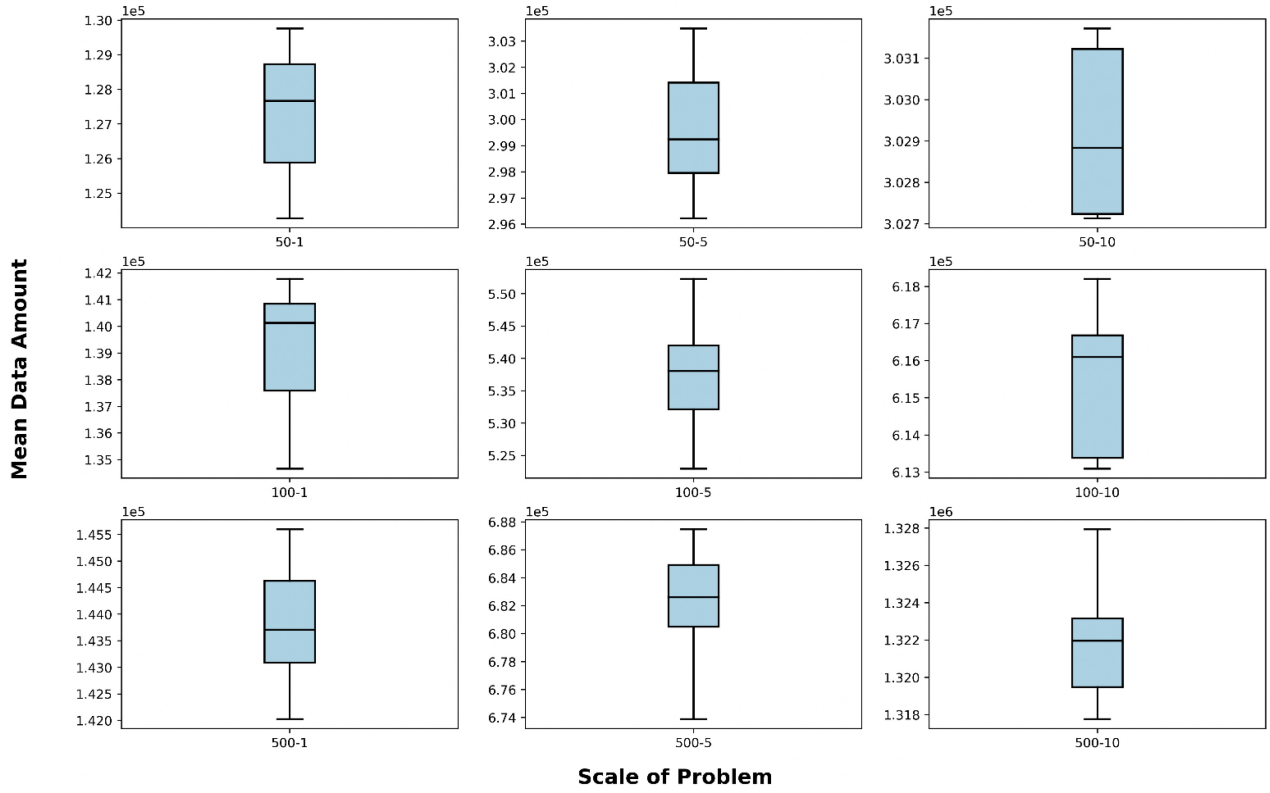

Fig. 8. Collected data amount obtained by proposed algorithm with different IoT device distributions.

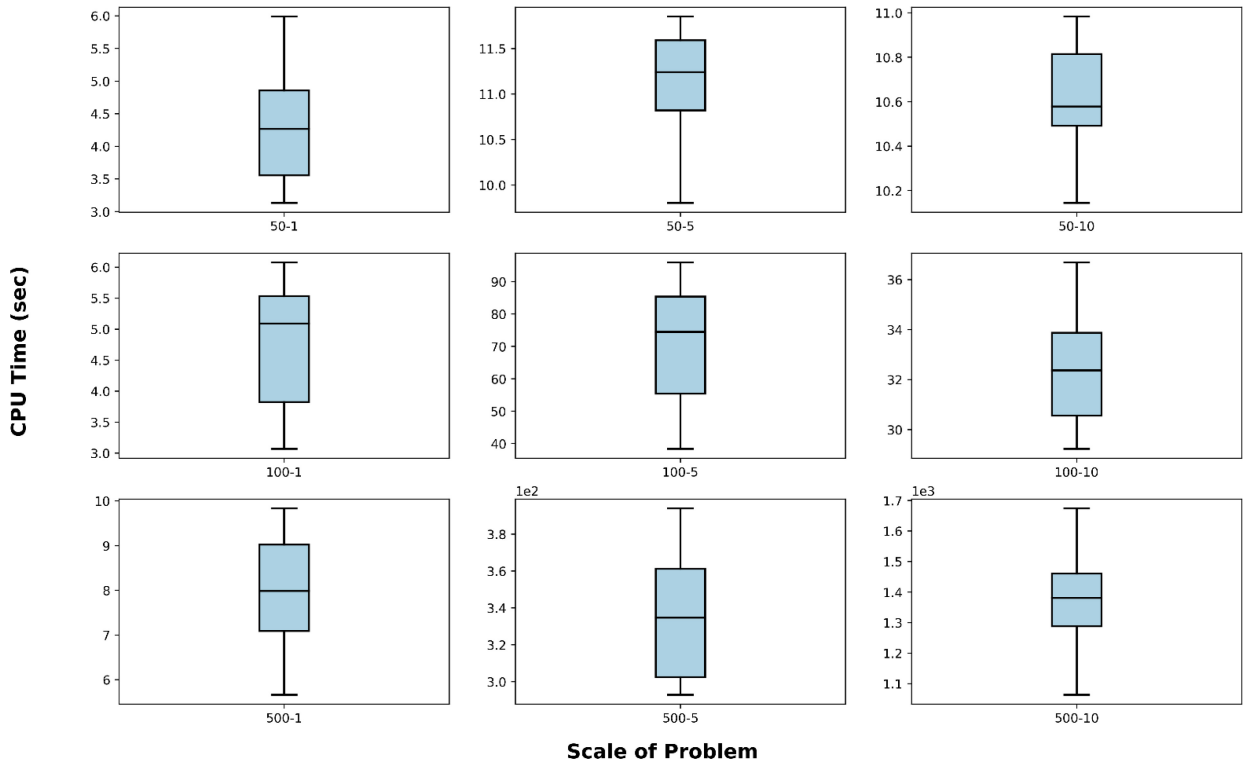

Fig. 9. CPU time of proposed algorithm with different IoT device distributions.

investigate the impact of various parameters on the algorithm's performance.

This work provides a practical solution to the multi-UAV route planning problem for IoT data collection in infrastructure monitoring, with potential applications in other real-world scenarios. Future work could incorporate more complex constraints and objectives, such as weather conditions and communication range, and explore other meta-heuristic

Authorized licensed use limited to: Hong Kong Polytechnic University. Downloaded on February 04,2026 at 10:11:58 UTC from IEEE Xplore. Restrictions apply.

algorithms and machine learning techniques to further improve the algorithm's performance.

APPENDIX
MODEL LINEARIZATION

The multi-UAV route planning model is linearized so that it can be solved by CPLEX. Here, the auxiliary variables $l_{i,j,k}$, defined in (25), are adopted to linearize the original model

$$l_{i,j,k} = s_i \cdot x_{i,j,k} \quad \forall i \in N, k \in K. \tag{25}$$

So that the model can be linearized by the below functions

$$\max \sum_{i=1}^{n} \sum_{j=0}^{n} \sum_{k=1}^{k} R_i \cdot l_{i,j,k} \tag{26}$$

subjected to

$$l_{i,j,k} - s_i \leq 0 \quad \forall i, j \in N, i \neq j, k \in K \tag{27}$$

$$s_i - l_{i,j,k} + M \cdot x_{i,j,k} \leq M \quad \forall i, j \in N, i \neq j, k \in K \tag{28}$$

$$l_{i,j,k} \leq M \cdot x_{i,j,k} \quad \forall i, j \in N, i \neq j, k \in K \tag{29}$$

$$l_{i,j,k} \geq 0 \quad \forall i, j \in N, k \in K. \tag{30}$$

The above linear constraints can guarantee that (26) holds. If $x_{i,j,k} = 1$, $l_{i,j,k} = s_i$ based on (27) and (28), while $x_{i,j,k} = 0$, $l_{i,j,k} = 0$ according to (29) and (30).

REFERENCES

[1] J. A. Stankovic, "Research directions for the Internet of Things," *IEEE Internet Things J.*, vol. 1, no. 1, pp. 3–9, Feb. 2014.

[2] J. H. Nord, A. Koohang, and J. Paliszkiewicz, "The Internet of Things: Review and theoretical framework," *Expert Syst. Appl.*, vol. 133, pp. 97–108, Nov. 2019.

[3] G. Xu, G. Q. Huang, and J. Fang, "Cloud asset for urban flood control," *Adv. Eng. Inform.*, vol. 29, no. 3, pp. 355–365, 2015.

[4] A. Verma, S. Prakash, V. Srivastava, A. Kumar, and S. C. Mukhopadhyay, "Sensing, controlling, and IoT infrastructure in smart building: A review," *IEEE Sensors J.*, vol. 19, no. 20, pp. 9036–9046, Oct. 2019.

[5] C. Du, S. Dutta, P. Kurup, T. Yu, and X. Wang, "A review of railway infrastructure monitoring using fiber optic sensors," *Sens. Actuators A, Phys.*, vol. 303, Mar. 2020, Art. no. 111728.

[6] F. Orellana, J. M. D. Blasco, M. Foumelis, P. J. D'Aranno, M. A. Marsella, and P. Di Mascio, "Dinsar for road infrastructure monitoring: Case study highway network of rome metropolitan (Italy)," *Remote Sens.*, vol. 12, no. 22, p. 3697, 2020.

[7] Z. Wei et al., "UAV-assisted data collection for Internet of Things: A survey," *IEEE Internet Things J.*, vol. 9, no. 17, pp. 15460–15483, Sep. 2022.

[8] M. T. Nguyen et al., "UAV-assisted data collection in wireless sensor networks: a comprehensive survey," *Electronics*, vol. 10, no. 21, p. 2603, Oct. 2021.

[9] Z. Wang, R. Liu, Q. Liu, J. S. Thompson, and M. Kadoch, "Energy-efficient data collection and device positioning in UAV-assisted IoT," *IEEE Internet Things J.*, vol. 7, no. 2, pp. 1122–1139, Feb. 2020.

[10] Q. Wang, H. Wang, W. Sun, N. Zhao, H.-N. Dai, and W. Zhang, "Aerial bridge: A secure tunnel against eavesdropping in terrestrial-satellite networks," *IEEE Trans. Wireless Commun.*, vol. 22, no. 11, pp. 8096–8113, Nov. 2023.

[11] S. R. Yeduri, N. S. Chilamkurthy, O. J. Pandey, and L. R. Cenkeramaddi, "Energy and throughput management in delay-constrained small-world UAV-IoT network," *IEEE Internet Things J.*, vol. 10, no. 9, pp. 7922–7935, May 2023.

[12] T. Bose, A. Suresh, O. J. Pandey, L. R. Cenkeramaddi, and R. M. Hegde, "Improving quality-of-service in cluster-based UAV-assisted edge networks," *IEEE Trans. Netw. Service Manag.*, vol. 19, no. 2, pp. 1903–1919, Jun. 2022.

[13] H. Hu, K. Xiong, G. Qu, Q. Ni, P. Fan, and K. B. Letaief, "AoI-minimal trajectory planning and data collection in UAV-assisted wireless powered IoT networks," *IEEE Internet Things J.*, vol. 8, no. 2, pp. 1211–1223, Jan. 2021.

[14] X. Li, P. Li, Y. Zhao, L. Zhang, and Y. Dong, "A hybrid large neighborhood search algorithm for solving the multi depot UAV swarm routing problem," *IEEE Access*, vol. 9, pp. 104115–104126, 2021.

[15] X. Wang, M. C. Gursoy, T. Erpek, and Y. E. Sagduyu, "Learning-based UAV path planning for data collection with integrated collision avoidance," *IEEE Internet Things J.*, vol. 9, no. 17, pp. 16663–16676, Sep. 2022.

[16] F. Zandieh, S. F. Ghannadpour, and M. M. Mazdeh, "New integrated routing and surveillance model with drones and charging station considerations," *Eur. J. Oper. Res.*, vol. 313, no. 2, pp. 527–547, 2024.

[17] J. Gong, T.-H. Chang, C. Shen, and X. Chen, "Flight time minimization of UAV for data collection over wireless sensor networks," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 9, pp. 1942–1954, Sep. 2018.

[18] C. Zhan and Y. Zeng, "Completion time minimization for multi-UAV-enabled data collection," *IEEE Trans. Wireless Commun.*, vol. 18, no. 10, pp. 4859–4872, Oct. 2019.

[19] M. Li, S. He, and H. Li, "Minimizing mission completion time of UAVs by jointly optimizing the flight and data collection trajectory in UAV-enabled WSNs," *IEEE Internet Things J.*, vol. 9, no. 15, pp. 13498–13510, Aug. 2022.

[20] X. Zhu et al., "Path planning of multi-UAVs based on deep Q-network for energy-efficient data collection in UAVs-assisted IoT," *Veh. Commun.*, vol. 36, Aug. 2022, Art. no. 100491.

[21] N. Bolourian and A. Hammad, "LiDAR-equipped UAV path planning considering potential locations of defects for bridge inspection," *Autom. Constr.*, vol. 117, Sep. 2020, Art. no. 103250.

[22] B. Z. Hilyatur Rozaliya, I.-L. Wang, and A. Muklason, "Multi-UAV routing for maximum surveillance data collection with idleness and latency constraints," *Procedia Comput. Sci.*, vol. 197, pp. 264–272, Jan. 2022.

[23] K. Li, W. Ni, and F. Dressler, "Continuous maneuver control and data capture scheduling of autonomous drone in wireless sensor networks," *IEEE Trans. Mobile Comput.*, vol. 21, no. 8, pp. 2732–2744, Aug. 2022.

[24] X. Zheng, F. Wang, and Z. Li, "A multi-UAV cooperative route planning methodology for 3D fine-resolution building model reconstruction," *ISPRS J. Photogramm. Remote Sens.*, vol. 146, pp. 483–494, Dec. 2018.

[25] H. Bayerlein, M. Theile, M. Caccamo, and D. Gesbert, "Multi-UAV path planning for wireless data harvesting with deep reinforcement learning," *IEEE Open J. Commun. Soc.*, vol. 2, pp. 1171–1187, 2021.

[26] O. Ghdiri, W. Jaafar, S. Alfattani, J. B. Abderrazak, and H. Yanikomeroglu, "Offline and online UAV-enabled data collection in time-constrained IoT networks," *IEEE Trans. Green Commun. Netw.*, vol. 5, no. 4, pp. 1918–1933, Dec. 2021.

[27] I. Khoufi, A. Laouiti, C. Adjih, and M. Hadded, "UAVs trajectory optimization for data pick up and delivery with time window," *Drones*, vol. 5, no. 2, p. 27, Apr. 2021.

[28] J. Lee and V. Friderikos, "Interference-aware path planning optimization for multiple UAVs in beyond 5G networks," *J. Commun. Netw.*, vol. 24, no. 2, pp. 125–138, Apr. 2022.

[29] P. Liu, H. He, T. Fu, H. Lu, A. Alelaiwi, and M. W. I. Wasi, "Task offloading optimization of cruising UAV with fixed trajectory," *Comput. Netw.*, vol. 199, Nov. 2021, Art. no. 108397.

[30] X. Wang and L. Duan, "Economic analysis of unmanned aerial vehicle (UAV) provided mobile services," *IEEE Trans. Mobile Comput.*, vol. 20, no. 5, pp. 1804–1816, May 2021.

[31] X. Wang, Z. Ning, S. Guo, M. Wen, L. Guo, and H. V. Poor, "Dynamic UAV deployment for differentiated services: A multi-agent imitation learning based approach," *IEEE Trans. Mobile Comput.*, vol. 22, no. 4, pp. 2131–2146, Apr. 2023.

[32] Z. Zhang, H. Liu, and G. Wu, "A dynamic task scheduling method for multiple UAVs based on contract net protocol," *Sensors*, vol. 22, no. 12, p. 4486, Jun. 2022.

[33] H. Wang and H. Du, "Time sensitive sweep coverage with minimum UAVs," *Theor. Comput. Sci.*, vol. 928, pp. 197–209, Sep. 2022.

[34] P. Wan, G. Xu, J. Chen, and Y. Zhou, "Deep reinforcement learning enabled multi-UAV scheduling for disaster data collection with time-varying value," *IEEE Trans. Intell. Transp. Syst.*, early access, Jan. 2, 2024, doi: 10.1109/TITS.2023.3345280.

Authorized licensed use limited to: Hong Kong Polytechnic University. Downloaded on February 04,2026 at 10:11:58 UTC from IEEE Xplore. Restrictions apply.

[35] Y. Long, G. Xu, J. Zhao, B. Xie, and M. Fang, "Dynamic truck–UAV collaboration and integrated route planning for resilient urban emergency response," *IEEE Trans. Eng. Manag.*, early access, Aug. 24, 2023, doi: 10.1109/TEM.2023.3299693.
[36] C. Zhang, P. Li, Q.-C. He, and F. Wang, "A literature review of perishable medical resource management," *Front. Eng. Manage.*, vol. 10, no. 4, pp. 710–726, 2023.
[37] Z. Shao, G. Cheng, J. Ma, Z. Wang, J. Wang, and D. Li, "Real-time and accurate UAV pedestrian detection for social distancing monitoring in COVID-19 pandemic," *IEEE Trans. Multimedia*, vol. 24, pp. 2069–2083, Apr. 2022.
[38] K. Messaoudi, O. S. Oubbati, A. Rachedi, A. Lakas, T. Bendouma, and N. Chaib, "A survey of UAV-based data collection: Challenges, solutions and future perspectives," *J. Netw. Comput. Appl.*, vol. 216, Jul. 2023, Art. no. 103670.
[39] Y. Wang et al., "Multi-UAV collaborative data collection for IoT devices powered by battery," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, 2020, pp. 1–6.
[40] X. Li, J. Tan, A. Liu, P. Vijayakumar, N. Kumar, and M. Alazab, "A novel UAV-enabled data collection scheme for intelligent transportation system through UAV speed control," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 4, pp. 2100–2110, Apr. 2021.
[41] Y. Li et al., "Data collection maximization in IoT-sensor networks via an energy-constrained UAV," *IEEE Trans. Mobile Comput.*, vol. 22, no. 1, pp. 159–174, Jan. 2023.
[42] J. Pietz and J. O. Royset, "Generalized orienteering problem with resource dependent rewards," *Nav. Res. Logist.*, vol. 60, no. 4, pp. 294–312, 2013.
[43] Q. Yu, Y. Adulyasak, L.-M. Rousseau, N. Zhu, and S. Ma, "Team orienteering with time-varying profit," *INFORMS J. Comput.*, vol. 34, no. 1, pp. 262–280, 2022.
[44] F. Carrabs, "A biased random-key genetic algorithm for the set orienteering problem," *Eur. J. Oper. Res.*, vol. 292, no. 3, pp. 830–854, 2021.
[45] Q. Yu, K. Fang, N. Zhu, and S. Ma, "A matheuristic approach to the orienteering problem with service time dependent profits," *Eur. J. Oper. Res.*, vol. 273, no. 2, pp. 488–503, 2019.
[46] X. Chou, L. M. Gambardella, and R. Montemanni, "A tabu search algorithm for the probabilistic orienteering problem," *Comput. Oper. Res.*, vol. 126, Feb. 2021, Art. no. 105107.
[47] N. Labadie, R. Mansini, J. Melechovskỳ, and R. W. Calvo, "The team orienteering problem with time windows: An LP-based granular variable neighborhood search," *Eur. J. Oper. Res.*, vol. 220, no. 1, pp. 15–27, 2012.
[48] N. Mladenović and P. Hansen, "Variable neighborhood search," *Comput. Oper. Res.*, vol. 24, no. 11, pp. 1097–1100, 1997.
[49] H. Liu et al., "An iterative two-phase optimization method based on divide and conquer framework for integrated scheduling of multiple UAVs," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 9, pp. 5926–5938, Sep. 2021.
[50] N. A. Kyriakakis, M. Marinaki, N. Matsatsinis, and Y. Marinakis, "A cumulative unmanned aerial vehicle routing problem approach for humanitarian coverage path planning," *Eur. J. Oper. Res.*, vol. 300, no. 3, pp. 992–1004, Aug. 2022.
[51] A. Al-Hourani and K. Gomez, "Modeling cellular-to-UAV path-loss for suburban environments," *IEEE Wireless Commun. Lett.*, vol. 7, no. 1, pp. 82–85, Feb. 2018.
[52] G. E. Athanasiadou and G. V. Tsoulos, "Path loss characteristics for UAV-to-ground wireless channels," in *Proc. 13th Eur. Conf. Antennas Propagat. (EuCAP)*, 2019, pp. 1–4.
[53] M. Chen, W. Liang, and S. K. Das, "Data collection utility maximization in wireless sensor networks via efficient determination of UAV hovering locations," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun. (PerCom)*, 2021, pp. 1–10.
[54] C. Verbeeck, P. Vansteenwegen, and E.-H. Aghezzaf, "The time-dependent orienteering problem with time windows: A fast ant colony system," *Ann. Oper. Res.*, vol. 254, no. 1, pp. 481–505, 2017.
[55] P. Vansteenwegen, W. Souffriau, G. V. Berghe, and D. Van Oudheusden, "Iterated local search for the team orienteering problem with time windows," *Comput. Oper. Res.*, vol. 36, no. 12, pp. 3281–3290, 2009.
[56] M. Khodadadian, A. Divsalar, C. Verbeeck, A. Gunawan, and P. Vansteenwegen, "Time dependent orienteering problem with time windows and service time dependent profits," *Comput. Oper. Res.*, vol. 143, Jul. 2022, Art.no. 105794.
[57] H. Gehring and J. Homberger, "A parallel two-phase metaheuristic for routing problems with time windows," *Asia-Pacfic J. Oper. Res.*, vol. 18, no. 1, p. 35, 2001.
[58] Y. Liu, K. Xiong, Y. Lu, Q. Ni, P. Fan, and K. B. Letaief, "UAV-aided wireless power transfer and data collection in rician fading," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 10, pp. 3097–3113, Oct. 2021.
[59] H. Mei, K. Yang, Q. Liu, and K. Wang, "3D-trajectory and phase-shift design for RIS-assisted UAV systems using deep reinforcement learning," *IEEE Trans. Veh. Technol.*, vol. 71, no. 3, pp. 3020–3029, Mar. 2022.
[60] C.-H. Nguyen and K.-V. Nguyen, "Multi-UAV assisted data gathering in WSN: A MILP approach for optimizing network lifetime," in *Proc. RIVF Int. Conf. Comput. Commun. Technol. (RIVF)*, 2021, pp. 1–5.
[61] K.-V. Nguyen, C.-H. Nguyen, T. Van Do, and C. Rotter, "Efficient multi-UAV assisted data gathering schemes for maximizing the operation time of wireless sensor networks in precision farming," *IEEE Trans. Ind. Informat.*, vol. 19, no. 12, pp. 11664–11674, Dec. 2023.
[62] R. Gama and H. L. Fernandes, "A reinforcement learning approach to the orienteering problem with time windows," *Comput. Oper. Res.*, vol. 133, Sep. 2021, Art. no. 105357.

Authorized licensed use limited to: Hong Kong Polytechnic University. Downloaded on February 04,2026 at 10:11:58 UTC from IEEE Xplore. Restrictions apply.