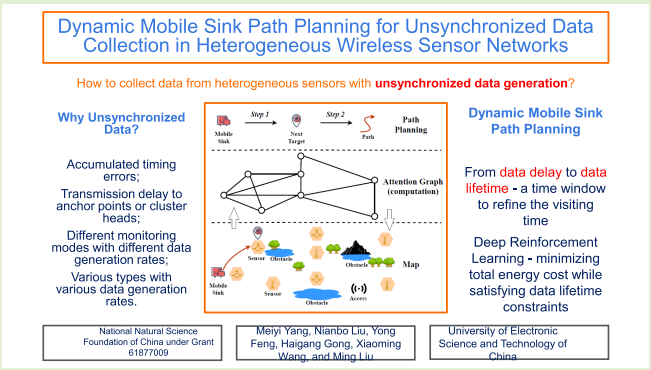
# Dynamic Mobile Sink Path Planning for Unsynchronized Data Collection in Heterogeneous Wireless Sensor Networks

Meiyi Yang, Nianbo Liu, Yong Feng, *Member, IEEE*, Haigang Gong, Xiaoming Wang, and Ming Liu

***Abstract*—In heterogeneous wireless sensor networks (HWSNs), heterogeneous sensors may follow various data-generating distributions, which makes data collection a very challenging task. Although mobile sinks (MSs) are widely used to collect data from wireless sensors, existing MS-assisted approaches are often based on the assumption of synchronized data, for example, all sensors generate data at the same time, in which data are considered delay-tolerant and -sensitive data according to delay limits, while the generation time of data is ignored. This article focuses on unsynchronized data collection for HWSNs, in which unsynchronized data generation of sensors is allowed as real as actual monitoring applications. First, to reflect the timeliness of data, we use a rigid collection window to represent the lifetime of sensing data to refine the visiting time of the MS. Second, a graph attention network (GAT) structure is adopted to describe node locations, accessible paths, and data with collection windows for path planning. Third, a new deep reinforcement learning (DRL)-based MS path planning (MSPP) framework is proposed to tackle the path of the MS by minimizing total energy cost while satisfying data lifetime constraints. MSPP first uses the Target Selector module to plan the moving targets and adopts the MS Controller module to control MS mobility for achieving fast convergence and better optimality. Finally, extensive simulations show that our scheme provides explicit data collection guarantees and minimum energy consumption.**




***Index Terms*— Deep reinforcement learning (DRL), heterogeneous wireless sensor networks (HWSNs), mobile sink (MS), path planning, unsynchronized data collection.**

## I. Introduction

IN RECENT years, wireless sensors and Internet-of-Things (IoT) devices have been deployed in remote locations for environmental monitoring, habitat monitoring, weather monitoring, and other monitoring applications for agricultural, military, research, and other purposes [1]. Since there are no base stations or internet services in wild areas and remote depopulated zones, mobile sinks (MSs), including robots, mobile vehicles (MVs), unmanned aerial vehicles (UAVs), and so on, are widely used to collect data from sensors, IoT devices, or disconnected sensor networks in these areas [2], [3], [4], [5]. One MS can periodically visit some sensors, collect data from them, and carry and forward data to nearby base stations. However, MS has a limited energy supply and thus intelligently navigating to the most appropriate places for data collection becomes a critical but challenging problem.

In traditional wireless sensor networks (WSNs), many mobile nodes-assisted data collection schemes [6], [7], [8], [9] are often based on the assumption of synchronized data, for example, all sensors generate data at the same time, which means we only consider the delay limits of data and ignore the generation time of data, as shown in Fig. 1(a). Thus, many MS-assisted approaches [2], [3], [10] are proposed for delay-tolerant and delay-sensitive data collection, for performing efficient MS path planning while satisfying all data delay constraints of sensors.

Manuscript received 23 February 2023; revised 10 April 2023 and 28 May 2023; accepted 28 June 2023. Date of publication 14 July 2023; date of current version 31 August 2023. This work was supported in part by the National Natural Science Foundation of China under Grant 61877009; in part by the National Key Research and Development Program under Grant 2022YFB3104600; in part by the Science and Technology Program of Quzhou through Project 377 under Grant 2021D007, Grant 2021D008, Grant 2021D015, and Grant 2021D018; and in part by the Project 378 under Grant LGF22G010009. The associate editor coordinating the review of this article and approving it for publication was Dr. Vinay Chakravarthi Gogineni. *(Corresponding author: Nianbo Liu.)*

Meiyi Yang, Haigang Gong, Xiaoming Wang, and Ming Liu are with the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China, and also with the Yangtze Delta Region Institute (Quzhou), University of Electronic Science and Technology of China, Quzhou 324000, China (e-mail: meiyiyang@std.uestc.edu.cn; liunb@uestc.edu.cn; hggong@uestc.edu.cn; xiaomin.wang@126.com; csmliu@uestc.edu.cn).

Nianbo Liu is with the Quzhou People Hospital, Wenzhou Medical University, Quzhou 324000, China, and also with the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China (e-mail: liunb@uestc.edu.cn).

Yong Feng is with the Department of Computer Science, Kunming University of Science and Technology, Kunming 650093, China (e-mail: fybraver@163.com).

Digital Object Identifier 10.1109/JSEN.2023.3294232

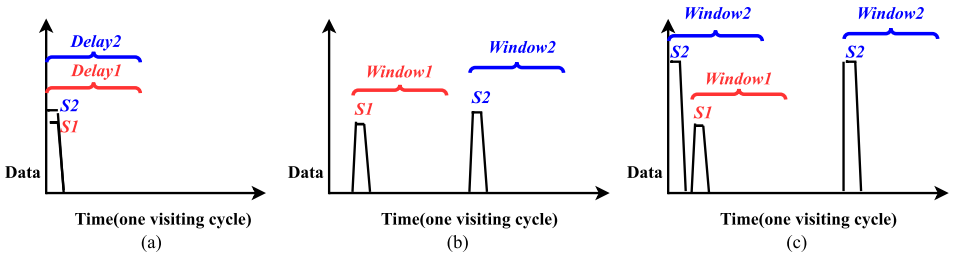



Fig. 1. Various data generations in homogeneous and heterogeneous sensors. (a) Sensors generate synchronized data. (b) Sensors generate unsynchronized data. (c) Sensors generate data with different rates.

Compared to traditional WSNs consisting of homogeneous sensors, heterogeneous WSNs (HWSNs) show a very challenging network environment, in which heterogeneous sensors may have various computing capabilities, communications, power supplies, or even various sensor types. Early researches on HWSNs concentrate often on security operations [11], coverage [12], costs [13], energy saving [14], and so on. In recent years, data fusion schemes [15], [16], [17], [18], [19] draw more and more attention to avoid the problems of long delay and data nonfresh in HWSNs, which often build carefully designed data-gathering path on some clustering strategies, $t$ to cope with different sensor groups following different data-generating distributions separately. To some extent, such clustering provides a flexible and adaptive data collection method for the HWSNs with limited heterogeneity.

However, in realistic monitoring scenes, the assumption of synchronized data does not always exist. To the HWSNs with high heterogeneity, data generation of sensors in a visiting cycle can be unsynchronized, which is often caused by the reasons as follows.

1) After long-time running, the accumulated timing errors in sensors lead to unsynchronized data generation, as shown in Fig. 1(b).
2) Sometimes, data are first delivered to some anchor points or cluster heads for collection, in which the transmission delays cause unsynchronized data, as shown in Fig. 1(b).
3) Sensors have different monitoring modes with different data generation rates, as shown in Fig. 1(c).
4) Sensors have various types with various data generation rates, as shown in Fig. 1(c).

When existing data collection schemes based on the assumption of synchronized data are deployed in the above scenes, the MS may visit sensors too early or too late, miss data, wait for data, or revisit the same node due to a lack of knowledge of data generation time. It will result in poor performance, and even failures, in data collection. Thus, the lifetime of data, not only the delay limit, should be considered to cope with the dynamic and heterogeneous nature of HWSNs.

In this article, we do not use a delay to define data transmission requirements but adopt a rigid collection window to represent the lifetime of data, for example, from data generation time to its maximum delay limit. For unsynchronized data collection, the MS should visit sensors within their data collection windows. Our goal is to design the path of the MS with minimizing the total energy cost while satisfying data lifetime constraints, and the overall study design is captured in Fig. 2. Compared with conventional path planning tasks, this problem poses nontrivial challenges in that the presence of obstacles and restricted areas makes paths among sensors not always accessible. To tackle this problem, a novel two-stage deep reinforcement learning (DRL) algorithm based on a graph attention network (GAT) network is proposed.

## II. Related Work

The main work and contributions of this study are as follows.

1) We reconsider the assumption of synchronized data in HWSNs and find that current data collection schemes may lead to unexpected performance deterioration in heterogeneous sensors with unsynchronized data generation.
2) We use data collection windows to replace delay constraints, as a more precise metric to indicate the available visiting time for heterogeneous sensors in MS path planning (MSPP).
3) We propose a novel DRL algorithm called MSPP, including a Target Selector for selecting target sensors and an MS Controller for training MS to visit target sensors, and the simulation results show that our method is effective and has a better performance.

The detailed structure of this article is as follows. Related work is introduced in Section II. The system model and problem definition are described in Section III. The solution to the problem is expressed in Section IV. The evaluation results are discussed in Section V. The conclusion of this article is summarized in Section VI.

### A. Data Collection in WSNs

To balance energy consumption and the packet-delay constraint, Dasgupta and Yoon [20] designed an optimization model to minimize delay and energy, which is an integer linear programming. The optimal trajectory for MVs can be obtained by solving this linear programming model. Lan et al. [21] took the joint optimization of sensor charging and data collection as the goal and transformed the path planning problem into a quadratic programming problem, obtaining the asymptotically optimal path through the approximate minimization algorithm. Tashtarian et al. [22] designed a practical path for

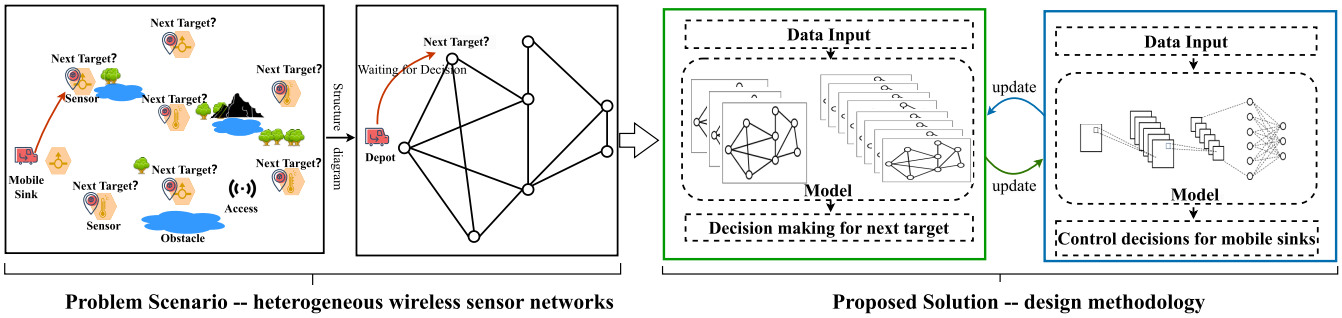



Fig. 2. Flowchart of the method design, encompassing problem scenarios and corresponding solutions. Our objective is to minimize overall energy consumption while ensuring the fulfillment of data transmission requirements for all sensors in HWSMs. Our proposed algorithm comprises two essential elements: a task selector and an MS controller. The cross-execution of these components effectively manages MS operations, facilitating efficient data collection.

MVs to collect data from sensor nodes. To minimize energy consumption and maximize data transmission, this article converted the path planning problem to a mixed-integer nonlinear programming model, which is calculated by a two-stage heuristic. Ojha and Chanak [23] proposed a multiobjective gray wolf optimized routing mechanism to alleviate the pressure of sensor nodes near the data-receiving center, which imitates the behavior of gray wolf to select the optimal intersection node for each cluster. Then, the MS visited each intersection node along the optimal path to collect sensing data.

In addition to MVs, UAVs are also used to collect data in some remote areas. Abdulla et al. [24] converted the problem of maximizing throughput in UAV-assisted data acquisition into a potential game. Monwar et al. [1] designed a novel path-planning algorithm with data collection to minimize overall energy consumption. Wu et al. [2] considered the problem with a communication delay of collected information. The delay-sensitive trajectory design problem was transformed into a symmetrical TSP problem, and a heuristic algorithm was proposed to find the optimal trajectory. Yoon et al. [10] focused on delay-sensitive information delivery in disaster scenarios. This article programed the optimal path of multiple UAVs to successfully transmit the data packets in the sensors that need to be visited. Yoon et al. [25] proposed a 3-D dc trajectory design for the dc-assisted IoT delay-sensitive data collection. Some methods suggested that the MS only access the cluster head of the sensor to receive data within the delay range and reduce the energy consumption of the UAVs [26], [27]. Several articles specifically examine challenges related to scenarios involving drones assisting in real-time or dynamic data collection [28], [29], [30].

### B. Data Collection in HWSNs

Sun et al. [15] considered the problem of quickest change detection in anonymous HWSNs, for example, an event occurs in the network and changes the data-generating distribution of the sensors at some unknown time, constructed a mixture CuSum algorithm, and proved that it is optimal under Lorden's criterion. Palanisamy et al. [16] proposed a multisensor data synchronization scheduling framework for efficient data aggregation at the sink in HWSNs, in which in-network aggregation sensor data routing is developed based on the dynamic routing with reliable data transmission.

Since energy is vital to the network's lifetime, energy-saving designs are still crucial in data collection schemes for HWSNs. Lin et al. [17] developed an MS-assisted data collection mechanism to collect data in HWSNs, for prolonging the network lifetime while improving the surveillance coverage, in which data collection points are dynamically selected in each round for balancing the lifetime and improving the surveillance quality, and each sensor and its parent are dynamically assigned to construct a tree topology for further reducing and balancing the energy consumptions of sensor nodes. Osamy et al. [18] determined energy-aware disjoint dominating sets that work as data collection nodes in each round, to improve overall network lifetime, in which an energy-aware algorithm based on swarm intelligence is proposed to construct disjoint dominating sets, and data gathering path is determined for achieving maximal data collection efficiency and reduced energy consumption. Kumar et al. [19] designed an enhanced energy-efficient clustering approach to increase the network lifetime and reduce the energy consumption in HWSNs, in which various parameters for selecting the cluster head, the initial energy of the sinks, the number of alive/dead nodes, and the remaining energy are considered. Chen and Tang [31], [32] proposed an energy-saving framework for UAV-assisted data collection in a heterogeneous wireless sensor WSN and introduced a 1-D search method to identify the optimal energy threshold.

Generally, existing approaches focus on synchronized data but ignore unsynchronized data in HWSNs with high heterogeneity. In addition, many solutions convert the trajectory design problem to a task assignment problem from an optimization theory perspective or propose heuristic strategies generally. It is known that heuristic methods cannot achieve state-of-the-art performance.

## III. Problem Formulation

### A. System Model

As shown in Fig. 3, this system consists of $I$ wireless sensors, $M$ MSs, one MS depot, and one data station. These sensors are deployed in a remote area with obstacles and restricted access. The data station can be a nearby base station, Wifi AP, satellite station, or others, and is located as the destination of data collection. Our model is built on the

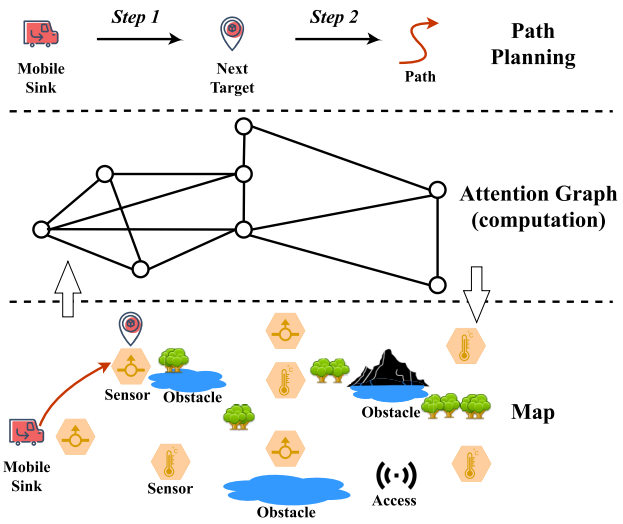



Fig. 3. Overview of HWSN environment.

following assumptions and conventions of sensors, MSs, and MS depot.

*1) Assumption of Sensors:* $I$ wireless sensors are sparsely distributed in an area, and each sensor can only be connected to one MS at a time. When these sensors generate sensory data, they will release some collection tasks with time windows. The collection window of data in $i$th$(i \in I)$ sensor is expressed as $[t_i^{\text{start}}, t_i^{\text{end}}]$, where $t_i^{\text{start}}$ indicates data generation time and $t_i^{\text{end}}$ is determined by maximum delay limit of data (excluding average transmission delay from the sensor to the data center).

Based on the collected historical sensor data, the data generation cycle of the sensor is predictable, which constructs the basis of MS trajectory planning. But the triggering of emergency mode, for example, the occurrence of abnormal events, cannot be predicted. It means that MS may encounter sensors in emergency mode. The MSPP algorithm should be updated online for such MS behavior changes.

*2) Assumption of MS:* With base station links to back servers, MS can receive sensor locations and data generation settings and updates of the MSPP algorithm, control messages, and other corrections. During the experiment, we install the MS's moving speed, energy consumption, and total available energy as $v$ m/s, $q_m$ J/m, and $E_m$ J, respectively. Each MS can access only one sensor node to collect data at a time. This study does not discuss the details of MS types, related mechanisms, or environments but assumes that MS can collect data from sensors.

*3) Assumption of MS Depot:* In an area, there is only one depot with the location $L_s$ as the start point of the MS trajectory. The depot can replenish energy for MS at any time.

We use an abstract layer of GAT to model system nodes and accessible paths in the map. To each sensor node, data with collection windows are explicitly stipulated as visiting constraints in a visiting cycle. Due to obstacles, the graph is not fully connected, so path planning is restricted within the accessible paths in the graph. Thus, a two-stage path-planning scheme is proposed to collect data with MSs, which is designed to reduce the complexity of the planning problem in smaller state space. In Step 1, we choose the next target from available sensors as the current visiting destination for the MS, while in Step 2, we train the MS to move to the target along an accessible path.

## B. Energy Cost of MSs

In this system, data transmitting and trajectory of each MS $m(m \in M)$ are the main energy consumption.

According to [3] and [33], we adopt the following energy consumption model of data transmission:

$$E_{\text{trans}} = \sum_{i=1}^{n} P_m^{\text{trans}} \frac{\Phi_i}{c_{mi}^{\text{trans}}}. \tag{1}$$

We define $P_m^{\text{trans}}$ as the transmission power of MSs. $\Phi_i$ is transmission data. $c_{mi}^{\text{trans}} = B_{mi} \log(1 + P_{\text{re}}^M/(k_B T_s B_{mi}))$ represents the channel capacity (in bits/s) from the sensor to the MS. For more details of the equation, refer to [34].

In addition to the energy consumption during data transmission, the main energy consumption of the MS is on the running trajectory. Suppose the MS is under uniform speed in a windless environment, its total energy consumption is calculated as

$$E^u(\Psi(t)) = \left(F_f + F_w\right) \cdot V_{\max} \cdot \frac{1}{\eta} \cdot t \tag{2}$$

where $F_f = \text{mg} \cdot f$ is the rolling resistance and $F_w = (1/2) C_D A \rho V^2$ is the windward resistance. mg is the gravity of the MS, $C_D$ is the wind resistance coefficient, $A$ is the windward area, and $\eta$ is the mechanical efficiency.

For the convenience of calculation, the time $t$ is discretized into $N$ time slots $\tau$. By this method, the continuous design space of a trajectory $\Psi(t)$ is discretized into $\Psi(t) = \sum_\tau \Psi[\tau](\tau = 0, 1, \ldots, N-1)$. We take $\tau$ as the time step of state update in Section IV-C.

Therefore, the total energy consumption of the MS is as follows:

$$E_{\text{total}} = E_{\text{tans}} + E^u(\Psi(t)). \tag{3}$$

## C. Problem Formulation

The fundamental problem is defined on graph $G = (V, E)$, where sensors in the area can be seen as vertices, and an edge exists if two vertices are reachable.

Since an MS's energy budget is limited, and both trajectory and data transmission consume a large amount of energy, we try to minimize the total energy consumption while satisfying all sensors' data transmission demands. This problem is formulated as follows:

$$\max \sum_{m \in M} \sum_{(i,j) \in E} \lambda_1 x_{ij} z_{mij} \Phi_j - \lambda_2 E_{mij} \tag{4}$$

$$\text{s.t.} \sum_{m \in M} \sum_{(i,j) \in E} z_{mij} = 1 \tag{5}$$

$$\sum_{j \in I} z_{m0j} = 1 \tag{6}$$

$$\sum_{j\in I} z_{mij} - \sum_{i\in I} z_{mij} = 0 \tag{7}$$

$$\sum_{i\in I} z_{mij} = 1 \tag{8}$$

$$\sum_{(i,j)\in E} E_{mij} \le E_m^{\text{init}} \tag{9}$$

$$t_{im}^{\text{arrive}} \le t_i^{\text{end}} \tag{10}$$

$$t_{im}^{\text{arrive}} + w_{im} + t_{im}^{\text{trans}} + \frac{d_{ij}}{v} \le t_j^{\text{end}} \tag{11}$$

$$w_{im} = \max\left\{0, t_i^{\text{start}} - t_i^{\text{arrive}}\right\} \tag{12}$$

where $\forall(i, j) \in E$, $\forall m \in M$, $\forall i \in I$, $x_{ij} \in \{0, 1\}$ and $z_{mij} \in \{0, 1\}$. $x_{ij} = 1$ indicates that an MS moves to sensor $j$ from sensor $i$ and accesses sensor $j$ successfully; $z_{mij}$ means an MS $m$ is being used. $t_{im}^{\text{trans}} = (\Phi_i / c_{im}^{\text{trans}})$ is the transmission time at the sensor $i$. Equation (5) denotes that each sensor can only be connected with one MS at the same time. Equations (6)–(8) are flow constraints. Equation (9) refers to the fact that the energy cost cannot exceed the energy capacity of the MS. Equations (10)–(12) are time constraints.

## IV. Proposed Solution

The greatest challenge for the combinatorial optimization problem is that the search space of solutions will be grown exponentially with the problem size, and it will result in hard to find the optimal solution, especially a large problem. DRL shows its potential to solve the NP-hard combinatorial optimization problem and has achieved several successes. Therefore, in this section, we first briefly introduce the proposed MSPP based on DRL and then demonstrate each component in detail.

### A. Overview

Fig. 4(a) shows the overview of the proposed MSPP. There are two key components: Task Selector and MS Controller. The Task Selector will select one of the candidate tasks (i.e., sensors that need to transmit data) as the target of one MS. The MS Controller controls the MS to move from the current position to the target position successfully.

As shown in Fig. 4(b), the Task Selector and MS controller have a similar pair of networks, for example, actor and critic networks. The actor and critic networks share some layers to reduce computation costs. We use a deep Q-learning network (DQN) [35] to train the Task Selector. In its actor network, three GAT layers [36] with PReLU activation are used to embed each node of the input graph, and the output layer of the actor network is a fully connected layer whose size depends on the number of sensors. Its critic network is the same as the actor network based on the GAT layer, and the output of the final layer is an estimation of the $Q$-values. On the other side, proximal policy optimization (PPO) [37] and related methods [38], [39] are utilized for training the MS Controller. The actor network consists of three linear layers with ReLU activation and one linear layer with Softplus activation. The critic network has five layers: three linear layers with ReLU activation shared with the parameters of the critic network and two linear layers with a single output.

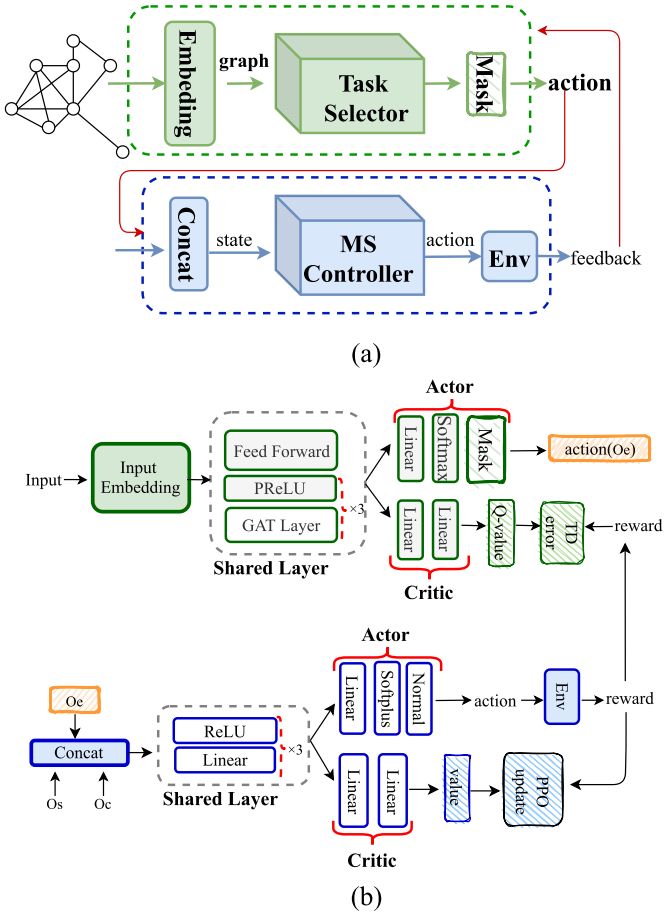



Fig. 4. Proposed MSPP framework. (a) Overview diagram. (b) Detail of the network.

### B. Task Selector

The purpose of the Task Selector is to select the most appropriate target in the current environment. The input state $S_{\text{selector}}$ is a five-tuple $S_{\text{selector}} = \langle G, V_{\text{mv}}, V_{\text{lv}}, T_{\text{cur}} \rangle$, where $G$ is an undirected graph based on sensor coordinates, $V_{\text{mu}}$ is the set that MS must visit, $V_{\text{lv}}$ is the set that the MS last visited, and $T_{\text{cur}}$ is the tour that has been traveled.

Since this problem has a natural graph structure, the input is an undirected graph, not a sequence, where the edge is characterized by distance, and the node is characterized by tasks' coordinates, time windows, and other features (if the task is accessed, the feature is 0, otherwise it is 1). Besides, the graph structure can handle instances of the same problem with a different number of variables (i.e., able to operate on nonfixed-dimensional feature vectors) and encode various input permutations to the same prediction [40]. We utilize the GAT network to embed the input, and a fully connected layer with softmax is used to output the target sensor that the MS will visit. However, one target may be invalid due to some constraints. According to (5)–(12), the set of valid actions can be expressed as

$$a_i \in V_{\text{mv}} \quad \forall i \in I \tag{13}$$

$$t_i^{\text{end}} \ge t_{i-1}^{\text{finish}} + \frac{d_{a_{i-1}a_i}}{v} + t_i^{\text{trans}} \quad \forall i \in I \tag{14}$$

$$t_i^{\text{finish}} \ge t_j^{\text{end}} (j \in V_{\text{mv}}) \quad \forall i \in I \tag{15}$$

where (13) represents a set of nonvisited sensors and (14) is a time constraint: the MS must visit the sensor within the task time window, and remove all sensors, whose time windows exceed the current time from $V_{\text{mv}}$ in (15). A valid action must satisfy (13)–(15), and the invalid action will be masked.

Since the action aims at obtaining a sensor whose data can be successfully collected by the MS with minimum energy consumption, we use the feedback of the MS Controller as the reward function

$$R_s = \lambda_1 x_{im} \Phi_i - \lambda_2 \sum_{\tau=0}^{N-1} p_2. \tag{16}$$

If the $i$th sensor is successfully served by the MS, then $x_{im} = 1$, otherwise $x_{im} = 0$. Coefficients $\lambda_1$ and $\lambda_2$ are introduced to standardize the reward. $\tau \in [0, N-1]$ is the time step of the MS Controller state update. Energy consumption $p_2$ is calculated following (19).

The initial state is enforced to start at the depot at $\tau = 0$, and all the sensors with data transmission demands will be visited. When one MS serves a target, the state is updated as follows: 1) the completed tasks are removed from $V_{\text{mv}}$; 2) it will be inserted into set $V_{\text{lv}}$; and 3) update MS travel time and task time windows.

### C. MS Controller

The objective of the MS Controller is to control the MS to move from the current position to the target position automatically. The real-time control of MSs can better adapt to an unknown environment, such as encountering obstacles and getting more accurate energy consumption and travel time.

The DRL agent needs to get all relevant information about the current state of the environment to fulfill the task successfully. Therefore, the state $S_{\text{control}} = \{o_s, o_c, o_e\}$ is made up of three parts: starting coordinates $o_s = (x, y, z)$, current environmental information $o_c = \{\text{velocity:}(v_x, v_y, v_z); \text{coordinates:}(x', y', z')\}$, and target coordinates $o_e = (x'', y'', z'')$. We have considered some aircraft manoeuvering restrictions in the action space of the MS, which is formulated as $\mathcal{A} = (v_x, v_y, v_z) \in (-1, 1)$.

We incorporate fuel consumption and distance from a starting coordinates $S(x_S, y_S, z_S)$ to the target coordinates $T(x_T, y_T, z_T)$, a heuristic reward function $r_t$ is constructed as follows:

$$r_t = \lambda_3 p_1 - \lambda_4 p_2 \tag{17}$$

where $\lambda_3$ and $\lambda_4$ are coefficients, and $p_1$ and $p_2$ are the evaluation factors of path performance. The concrete meaning and calculation methods are as follows.

1) $p_1$ is related to the distance, which provides direction guidance for MSs' action selection policy. We use an example to explain how to convert the distance into a part of the reward function. There are starting coordinates $S(x_S, y_S, z_S)$, target coordinates $T(x_T, y_T, z_T)$, and current coordinates $C(x_C, y_C, z_C)$. $p_1$ is given by

$$\max p_1 = (d_{\text{ST}} - d_{\text{CT}}) - (d_{\text{SC}} + d_{\text{CT}} - d_{\text{ST}}). \tag{18}$$

Among them, the calculation of distance is $d_{\text{ST}} = |(x_T - x_S) + (y_T - y_S) + (z_T - z_S)|$. Distance $d_{\text{ST}}$ is a fixed value, and maximizing $(d_{\text{ST}} - d_{\text{CT}})$ means that the closer the MS is to the destination, the more rewards the agent will get. We expect the trajectory of the MS to be close to a straight line from $S$ to $T$. According to the trilateral relationship of the triangle, minimizing $(d_{\text{SC}} + d_{\text{CT}} - d_{\text{ST}})$ is well suited to meet the requirements.

2) To reduce energy consumption, we use $p_2$ to represent the maneuvering cost of the MS's executing action $a_t$. Following (2), $p_2$ is given:

$$\min p_2 = \left\{ \left(F_f + F_w\right) \cdot V_\tau \cdot \frac{1}{\eta} \right\} \tau. \tag{19}$$

The reward mechanism is used to convey the tasks the agent needs to complete. Supposing that, when MS encounters obstacles during movement, a negative reward value $r_t = -100$ will be fed back to the MS Controller. Finally, the reward function is designed by

$$r_t = \begin{cases} -100, & \text{collision} \\ \lambda_3 p_1 - \lambda_4 p_2, & \text{otherwise.} \end{cases} \tag{20}$$

We implemented the MS Controller based on PPO, whose procedure is displayed in Algorithm 1.

**Algorithm 1** MS Controller

**Input:** Environment $E$, Action Space $A$, Initial Status $S = \{o_s, o_c, o_e\}$
**Output:** $< IsDone, \sum p_2 >$

**repeat**
  Generate corresponding action $a_t$ by action network
  Execute action $a_t$ and observe new observation $o_c$
  $\tau \leftarrow \tau + 1$
**until** Service success or failure
Use Equation (17) to compute reward $R$
Estimate the reward by $V(s) = F(s|\omega_V)$
Estimate advantages $\hat{A} = \sum R - V(s)$
Update the critic network: $\nabla_{\omega_V} E[(R^i - V^i)^2]$
Update the actor network: $\nabla_{\omega_G} E[min(r_\theta \hat{A}, clip(r(\theta), 1-\epsilon, 1+\epsilon)\hat{A})]$
**if** the service successes **then**
  Compute feedback reward $\sum p_2$
  IsDone = True
**else**
  Compute feedback reward $\sum p_2$
  IsDone = False
**end if**
**return** $< IsDone, \sum p_2 >$

When the MS reaches the destination, if the arrival time of the MS is less than $t_i^{\text{end}} - t_i^{\text{trans}}$, the service is successful, and a positive reward signal $\sum p_2$ are given to the Task Selector, otherwise a negative reward $-\sum p_2$ are responded.

### D. MSPP Algorithm

After choosing an MS's next destination as the target by the Task Selector, the MS Controller will automatically control the MS moving to the target. We implemented the MSPP algorithm based on DQN and the learning procedure is outlined in Algorithm 2.

**Algorithm 2** MSPP Algorithm

**Input:** $< G, V_{mv}, V_{lv}, T_{cur} >$
**Output:** Best Path Found

**for** episode $k = 1, \ldots, K$ **do**
  Generate a random instance $Q_p$
  Get task set $V$
  $V_{mv} = V$ and $T_{cur} = \{0\}$
  **while** $V_{mv} \neq \emptyset$ **do**
    With probability $\varepsilon - greedy$ policy to select an action $a_t$ based on the proposed constraints
    otherwise select $a_t = argmax_a Q(s, a)$
    Get destination $f(a_t)$
    Get information $< IsDone, \sum p_2 >$ by Algorithm 1
    **if** IsDone == True **then**
      Reward $R_s = \lambda_1 \Phi_i - \lambda_2 \sum_\tau p_2$
      $V_{mv} = V_{mv} - [f(a_t)]$
      $V_{lv} = [f(a_t)]$
      $T_{cur} = T_{cur} + [f(a_t)]$
    **else**
      Reward $R_s = -\lambda_2 \sum_\tau p_2$
    **end if**
    Store transition tuple to replay memory $\mathcal{D}$
    Sample random mini-batch from $\mathcal{D}$
    Update actor network with gradient descent.
    Update critic network with TD-error.
  **end while**
**end for**

## V. Experiments

In this section, we will evaluate our proposed method on two common datasets air quality and meteorological data in Beijing. And, more experimental details will be provided.

### A. Experimental Settings

*1) Data:* We evaluate our model based on two real datasets: air quality and meteorological data in Beijing from 2014/05/01 to 2015/04/30 [41], which has 8759 timestamps, respectively. The air quality data is collected at 36 air quality monitoring stations, and the meteorological data is collected by 16 sensors. Fig. 5 shows four days of data from four sensors. From Fig. 5, we can see that the start time and end time of node data generation are different for different sensors. Therefore, it is crucial to obtain precise data to establish a data collection window based on real-time data generation and maximum delay.

Sensors are drawn on the map based on location information of air quality sensors and meteorological sensors. The distance between two locations is calculated by Euclidean distance. The data size of the sensor is given by $\Phi_i \sim [10, 50]$ Mbits. Consulting the data generation mode of air quality data and the meteorological data, the time windows of data in sensor $i$ are represented as $[t_i^{\text{start}}, t_i^{\text{end}}]$.

*2) Simulation Settings:* We use Airsim [42] to set up an MS simulation environment. In the experimental parameter setting, we use some common assumptions in the field of wireless sensor data collection. The experimental parameters are shown in Table I.

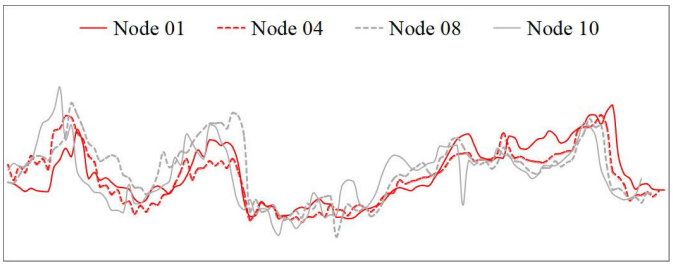



Fig. 5. Four days time-series plot of random four sensors.

TABLE I
Experimental Parameters

| Parameter | Value |
|---|---|
| Number of MSs | $1 \sim 8$ |
| $E^i$, MS Initial Energy | 300KJ |
| $v$, Speed of the MS | $0 \sim 1$m/s |
| $a_{max}$, Acceleration | $1m/s^2$ |
| $P^m_{trans}$, Transmission power from MS | 10W |
| $m$, The mass of MS | 1000 Kg |
| $f$, The rolling friction coefficient | 0.032 |
| $C_D$, The drag coefficient | 0.28 |
| $A$, The windward area | 1.87 $m^2$ |
| $\rho$, Coefficient of correction | 0.05 |
| $\eta$, The mechanical efficiency | 0.8 |
| $\lambda_1, \lambda_2, \lambda_3, \lambda_4$ | 0.6, 0.4, 0.6, 0.4 |
| $c_1$, $c_2$ | $9.264 \times 10^{-4}$, 2250 |
| $\delta^2$ | -150dBm |

TABLE II
Hyperparameter Values for the MSPP Algorithm

| Parameter | Task Selector | | MS Controller | |
|---|---|---|---|---|
| | Area one | Area two | Area one | Area two |
| Batch size | 32 | 32 | 64 | 64 |
| Learning rate | 0.0001 | 0.0001 | 0.001 | 0.001 |
| GAT dimension | 32 | 32 | - | - |
| Hidden dimension | 32 | 64 | 50 | 100 |
| Softmax(temperature) | 10 | 10 | - | - |
| Episode | 100 | 400 | 50 | 50 |
| PPO clip | - | - | 0.2 | 0.2 |
| Time step $\tau$ | - | - | 2s | 2s |

*3) Training Settings:* The proposed model is implemented by using PyTorch 1.0.1 and Deep graph library [43]. We run the experiments on a Ubuntu 16.04.3 server with four NVIDIA GeForce GTX 1080 cards. More hyperparameters used are summarized in Table II.

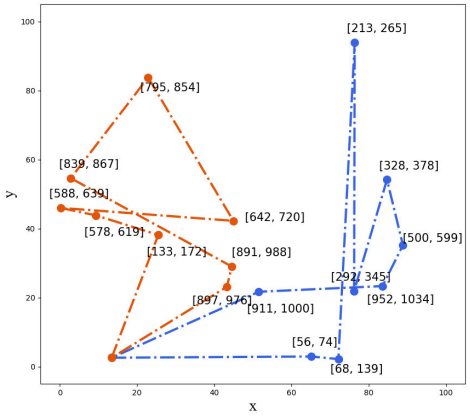
*4) Evaluation Metrics:* To achieve the goals defined in (4), we use the following two metrics: data collection ratio ($\Phi_{\text{recive}}/\Phi_{\text{total}}$) and total energy consumption ($E_{\text{total}}/I$).

### B. Illustrative Movement Trace

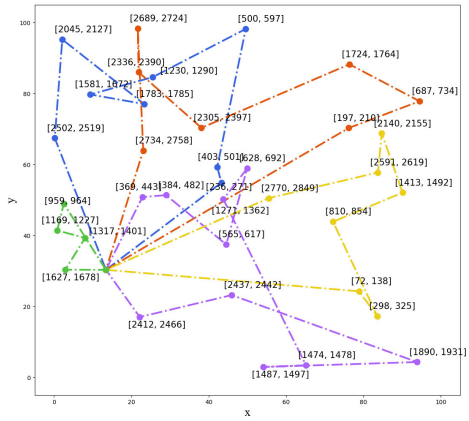
The movement trace of two areas solved by the MSPP algorithm are illustrated in Fig. 6(a) and (b), respectively. The first area has 16 meteorological sensors, and the second area has 36 air quality sensors. The details of 16 instances are presented in Table III. We have 16 sensors indexed 1 and 16, and the MS depot is located at index 0.

TABLE III
SOLUTIONS FOUND FOR SAMPLE 16

| |
|---|
| Sample instance for 16:<br>Sensor coordinates: x = [13.4, 84.7, 76.3, 25.5, 49.5, 44.9, 65.1, 78.8, 9.3, 2.8, 83.5, 43.2, 76.2, 0.2, 44.5, 72.1, 22.8]<br>y = [2.5, 54.1, 93.9, 38.1, 21.6, 42.2, 2.9, 22.1, 43.7, 49.5, 23.3, 23.0, 21.8, 45.9, 28.9, 2.1, 83.7]<br>Time window:[0, 1000], [328, 378], [213, 265], [133, 172], [911, 1000], [642, 720], [56, 74], [500, 599],<br>+ [578, 619],[839, 867],[952, 1034],[897, 976], [292, 345],[588, 639], [891, 988], [68, 139], [795, 854]<br>Data size: [0, 15.1, 15.7, 11.8, 25.4, 25.1, 10.1, 48.5, 13.6, 10.0, 35.6, 35.9, 26.5, 25.5, 48.5, 35.6, 29.5] |
| Tour for 16 sensors:<br>Tour for MS 1: $0 \rightarrow 6 \rightarrow 15 \rightarrow 2 \rightarrow 12 \rightarrow 1 \rightarrow 7 \rightarrow 10 \rightarrow 4 \rightarrow 0$, Energy of the route: 287KJ<br>Tour for MS 2: $0 \rightarrow 3 \rightarrow 8 \rightarrow 13 \rightarrow 5 \rightarrow 16 \rightarrow 9 \rightarrow 14 \rightarrow 11 \rightarrow 0$, Energy of the route: 223KJ |




(a)




(b)

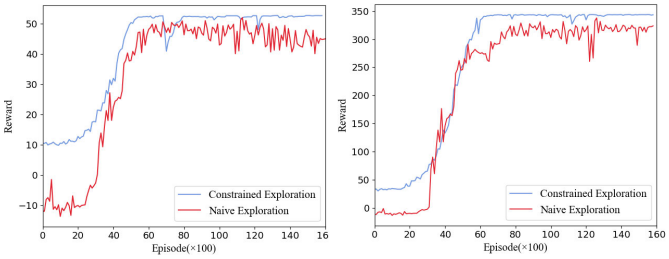
Fig. 6. Path planning of MSs' visiting sensor nodes. (a) Path planning for 16 sensors using the MSPP algorithm costs 510 KJ. (b) Path planning for 36 sensors using the MSPP algorithm costs 1076 KJ.

Here you can see that, to ensure all data are collected, 16 sensors need two MSs, and 36 sensors in the same size area need five MSs. Experimental results show that it is required for multiple MSs to collect data cooperatively in remote areas.

## C. Comparison With Naive Exploration

To evaluate the performance of the designed exploration, we compare it with naive exploration, which uses a greedy selection policy without constraints. Fig. 7 shows the average rewards of the learning process in areas one and two.

We can see clearly that the proposed exploration strategy obtains more rewards. During the initial episodes of the naive




(a) (b)

Fig. 7. Average reward per episode with DQN in two areas. (a) Area one. (b) Area two.

exploration, the value of the reward may be minus. That is because: 1) the cost of energy consumption outweighs the reward of collected data and 2) when the MS fails to provide service, the agent receives a negative reward signal. Furthermore, the differences between the two explorations are more significant with the increase in network size. It is because the probability of constructing a valid path by blind searching is smaller in a larger network. Our proposed exploration strategy can effectively reduce search space and make the MSPP algorithm converge to a better solution faster.

## D. Comparison With Baselines

We compare the performance of our proposed method with the classical Greedy algorithm, Genetic algorithm, and an industrial solver developed by Google OR-Tools.

*1) Greedy Algorithm:* In the Greedy algorithm, the MS's speed $v$ is set to 0.1 m/s. Therefore, the MSs' energy consumption from coordinate $S$ to coordinate $T$ can be calculated offline by the MS Controller. The Greedy algorithm greedily selects the next target vertex based on the constraints (7)–(14).

*2) Genetic Algorithm:* The genetic algorithm we compared comes from the article [44]. Each chromosome represents a valid path. After chromosome coding, a set of individuals (paths) are initialized. Then, the chromosomes with a crossover and a mutation are substituted into the reward function (16), and the chromosomes with large rewards are selected as the next generation until they have the maximum reward. The path with the maximum reward in (16) is regarded as the final solution.

*3) OR-Tools:* A hybrid model based on constraint programming and local search using the OR-Tools solver. We refer to

TABLE IV
RESULTS FOR COMPARISON. THE AVERAGE CONSUMPTION REPORTS THE AVERAGE ENERGY CONSUMPTION AND TIME REPORTS THE AVERAGE EXECUTION TIME TO COMPLETE THE SEARCH (IN MINUTES, AND ONLY INCLUDING THE INSTANCES WHERE THE SEARCH HAS BEEN COMPLETED)

| Approaches | Area one (16 sensors) | | | Area two (36 sensors) | | |
|---|---|---|---|---|---|---|
| | Data receiving rates | Average consumption | Time | Data receiving rates | Average consumption | Time |
| Greedy | 0.98 | 36.1 | 0.34 | 0.87 | 39.1 | 1.85 |
| Genetic | 0.93 | 33.6 | 0.37 | 0.93 | 35.5 | 2.03 |
| OR-Tools | 0.98 | 32.5 | 0.30 | 0.93 | 33.0 | 1.81 |
| MSPP (our) | 1 | 31.8 | 0.35 | 1 | 30.5 | 2.15 |

the VRPTW example for revision (https://developers.google.com/optimization/routing/vrptw).

Table IV shows the results of comparison with baselines on maps of different sizes. In terms of data-receiving rate, we observe that the heuristic solution also performed well when the network size is small (e.g., 16 sensors) because heuristics can get the same feasible solution in less time. However, the result of the MSPP algorithm outperforms these heuristic algorithms on larger maps with more sensors. It means that with the increasing scale of the network, MSPP begins to show its superiority. Meanwhile, Table IV shows average energy consumption, where the performance of the MSPP algorithm is better than the baselines on different maps.

### E. Impacts of the Time Window

We study the impacts of the proposed time window mode and baseline mode without time windows on the system, as shown in Figs. 8 and 9. We compare them from two aspects: data-receiving rate and energy consumption. From Figs. 8 and 9, we can see that our mode has the highest data-receiving rate and lower energy consumption. Through analysis, we find that the baseline scheme does not consider the generation time of sensor data, and the MS may arrive at the destination in advance, resulting in no sensor data can be received. Therefore, the baseline mode cannot obtain all the data in the current cycle. In addition, according to the setting of its data window, our scheme will judge that the MS should wait in place when a sensor generates multiple data in a short time. Our mode differs from the baseline scheme in which MS multiple trips to the same sensor, reducing energy consumption.

### F. Ablation Study

In this section, we mainly discuss how the MS Controller affects data collection performance. Our experiment uses the naive algorithm without the MS Controller module as the baseline. Results are shown in Table V. Here, we report the relative improvement of the MS Controller over the baseline, where the MS Controller helps the system collect more data and obtain consistent improvements in all target networks. Note that the MS Controller benefits data collection, possibly because the trajectory planning based on the MS Controller is more suitable for the complex real world.

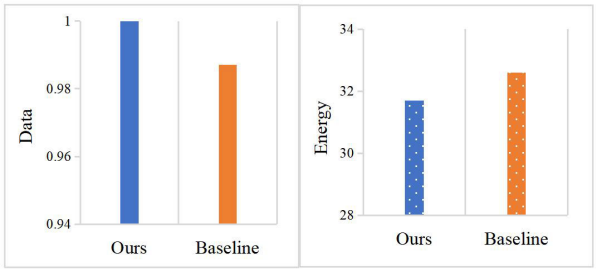



Fig. 8. Impacts of the time window under area one. (a) Data-receiving rate. (b) Average consumption.

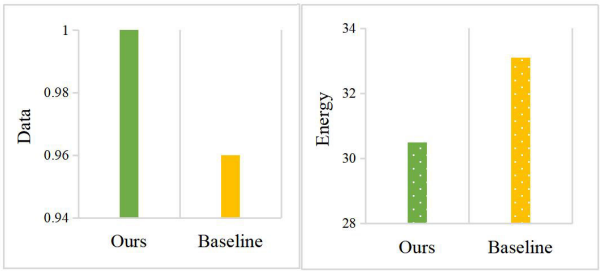



Fig. 9. Impacts of time window under area two. (a) Data-receiving rate. (b) Average consumption.

TABLE V
ABLATION STUDY OF THE MS CONTROLLER MODULE

| Parameter | Data receiving rate | | Average consumption | |
|---|---|---|---|---|
| | Area one | Area two | Area one | Area two |
| Task Selector | 0.96 | 0.91 | 32.6 | 31.3 |
| +MS Controller | 1 | 1 | 31.8 | 30.5 |

## VI. CONCLUSION

Considering the characteristics of real-time data generation, this article proposes a new data collection scheme. Based on two real-world datasets, the scheme establishes a data collection window for each data, whose start and end times of the time window are composed of real-time data generation and maximum delay. And then, a two-stage DRL-based algorithm MSPP is proposed for MS trajectory planning, in which a novel exploration strategy is designed

to meet various time windows of data in sensors for better efficiency and optimality. The simulation results prove that our method is capable of guaranteeing data transmission of sensors by only using minimal energy costs. Since a general frame has been investigated under realistic settings of tasks, sensors, MSs, and networks, our method provides insights into various monitoring applications, such as the on-demand detection of temperature, sound, pollution levels, humidity, and so on. Future work would be interesting to extend this method to more complex network environments, where a distributed multi-MS cooperation may be a practical solution.

## REFERENCES


[1] M. Monwar, O. Semiari, and W. Saad, "Optimized path planning for inspection by unmanned aerial vehicles swarm with energy constraints," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2018, pp. 1–6.

[2] P. Wu, C. Sha, H. Huang, and H. Wang, "Delay-sensitive trajectory designing for UAV-enabled data collection in Internet of Things," in *Proc. ACM Turing Celebration Conf.-China*, May 2020, pp. 77–81.

[3] Z. Jia, M. Sheng, J. Li, D. Niyato, and Z. Han, "LEO-satellite-assisted UAV: Joint trajectory and data collection for Internet of Remote Things in 6G aerial access networks," *IEEE Internet Things J.*, vol. 8, no. 12, pp. 9814–9826, Jun. 2021.

[4] X. Jiao, W. Lou, S. Guo, N. Wang, C. Chen, and K. Liu, "Hypergraph-based active minimum delay data aggregation scheduling in wireless-powered IoT," *IEEE Internet Things J.*, vol. 9, no. 11, pp. 8786–8799, Jun. 2022.

[5] X. Jiao et al., "Delay efficient scheduling algorithms for data aggregation in multi-channel asynchronous duty-cycled WSNs," *IEEE Trans. Commun.*, vol. 67, no. 9, pp. 6179–6192, Sep. 2019.

[6] Y. V. Pehlivanoglu, O. Baysal, and A. Hacioglu, "Path planning for autonomous UAV via vibrational genetic algorithm," *Aircr. Eng. Aerosp. Technol.*, vol. 79, no. 4, pp. 352–359, Jul. 2007.

[7] V. Roberge, M. Tarbouchi, and G. Labonte, "Comparison of parallel genetic algorithm and particle swarm optimization for real-time UAV path planning," *IEEE Trans. Ind. Informat.*, vol. 9, no. 1, pp. 132–141, Feb. 2013.

[8] Q. Wang and X. Chang, "The optimal trajectory planning for UAV in UAV-aided networks," in *Proc. Int. Conf. Cloud Comput. Secur.*, Nov. 2016, pp. 192–204.

[9] M. D. S. Arantes, J. D. S. Arantes, C. F. M. Toledo, and B. C. Williams, "A hybrid multi-population genetic algorithm for UAV path planning," in *Proc. Genetic Evol. Comput. Conf.*, Jul. 2016, pp. 853–860.

[10] J. Yoon, Y. Jin, N. Batsoyol, and H. Lee, "Adaptive path planning of UAVs for delivering delay-sensitive information to ad-hoc nodes," in *Proc. IEEE Wireless Commun. Netw. Conf.*, Mar. 2017, pp. 1–6.

[11] X. Du, Y. Xiao, M. Guizani, and H.-H. Chen, "An effective key management scheme for heterogeneous sensor networks," *Ad Hoc Netw.*, vol. 5, no. 1, pp. 24–34, Jan. 2007.

[12] L. Lazos and R. Poovendran, "Stochastic coverage in heterogeneous sensor networks," *ACM Trans. Sensor Netw. (TOSN)*, vol. 2, no. 3, pp. 325–358, Aug. 2006.

[13] V. Mhatre and C. Rosenberg, "Homogeneous vs heterogeneous clustered sensor networks: A comparative study," in *Proc. IEEE Int. Conf. Commun.*, Jun. 2004, pp. 3646–3651.

[14] I. S. Alshawi, Z. A. Abbood, and A. A. Alhijaj, "Extending lifetime of heterogeneous wireless sensor networks using spider monkey optimization routing protocol," *TELKOMNIKA Telecommun. Comput. Electron. Control*, vol. 20, no. 1, pp. 212–220, 2022.

[15] Z. Sun, S. Zou, R. Zhang, and Q. Li, "Quickest change detection in anonymous heterogeneous sensor networks," *IEEE Trans. Signal Process.*, vol. 70, pp. 1041–1055, 2022.

[16] T. Palanisamy, D. Alghazzawi, S. Bhatia, A. A. Malibari, P. Dadheech, and S. Sengan, "Improved energy based multi-sensor object detection in wireless sensor networks," *Intell. Autom. Soft Comput.*, vol. 33, no. 1, pp. 227–244, 2022.

[17] Z. Lin, H.-C. Keh, R. Wu, and D. S. Roy, "Joint data collection and fusion using mobile sink in heterogeneous wireless sensor networks," *IEEE Sensors J.*, vol. 21, no. 2, pp. 2364–2376, Jan. 2021.

[18] W. Osamy, A. Salim, A. M. Khedr, and A. A. El-Sawy, "IDCT: Intelligent data collection technique for IoT-enabled heterogeneous wireless sensor networks in smart environments," *IEEE Sensors J.*, vol. 21, no. 18, pp. 21099–21112, Sep. 2021.

[19] N. Kumar, P. Rani, V. Kumar, S. V. Athawale, and D. Koundal, "THWSN: Enhanced energy-efficient clustering approach for three-tier heterogeneous wireless sensor networks," *IEEE Sensors J.*, vol. 22, no. 20, pp. 20053–20062, Oct. 2022.

[20] R. Dasgupta and S. Yoon, "Energy-efficient deadline-aware data-gathering scheme using multiple mobile data collectors," *Sensors*, vol. 17, no. 4, p. 742, Apr. 2017.

[21] X. Lan, Y. Zhang, L. Cai, and Q. Chen, "Adaptive transmission design for rechargeable wireless sensor network with a mobile sink," *IEEE Internet Things J.*, vol. 7, no. 9, pp. 9011–9025, Sep. 2020.

[22] F. Tashtarian, K. Sohraby, and A. Varasteh, "Multihop data gathering in wireless sensor networks with a mobile sink," *Int. J. Commun. Syst.*, vol. 30, no. 12, p. e3264, Aug. 2017.

[23] A. Ojha and P. Chanak, "Multiobjective gray-wolf-optimization-based data routing scheme for wireless sensor networks," *IEEE Internet Things J.*, vol. 9, no. 6, pp. 4615–4623, Mar. 2022.

[24] A. E. Abdulla, Z. M. Fadlullah, H. Nishiyama, N. Kato, F. Ono, and R. Miura, "An optimal data collection technique for improved utility in UAS-aided networks," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr./May 2014, pp. 736–744.

[25] J. Yoon, S. Doh, O. Gnawali, and H. Lee, "Time-dependent ad-hoc routing structure for delivering delay-sensitive data using UAVs," *IEEE Access*, vol. 8, pp. 36322–36336, 2020.

[26] S. Roy, N. Mazumdar, and R. Pamula, "An energy and coverage sensitive approach to hierarchical data collection for mobile sink based wireless sensor networks," *J. Ambient Intell. Humanized Comput.*, vol. 12, no. 1, pp. 1267–1291, Jan. 2021.

[27] A. Mehto, S. Tapaswi, and K. K. Pattanaik, "PSO-based rendezvous point selection for delay efficient trajectory formation for mobile sink in wireless sensor networks," in *Proc. Int. Conf. Commun. Syst. Netw. (COMSNETS)*, Jan. 2020, pp. 252–258.

[28] Y. Zeng and J. Tang, "MEC-assisted real-time data acquisition and processing for UAV with general missions," *IEEE Trans. Veh. Technol.*, vol. 72, no. 1, pp. 1058–1072, Jan. 2023.

[29] Y. Zeng and J. Tang, "Real-time data acquisition and processing under mobile edge computing-assisted UAV system," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2022, pp. 5680–5685.

[30] R. I. D. Silva, J. D. C. V. Rezende, and M. J. F. Souza, "Collecting large volume data from wireless sensor network by drone," *Ad Hoc Netw.*, vol. 138, Jan. 2023, Art. no. 103017.

[31] J. Chen and J. Tang, "UAV-assisted data collection for dynamic and heterogeneous wireless sensor networks," *IEEE Wireless Commun. Lett.*, vol. 11, no. 6, pp. 1288–1292, Jun. 2022.

[32] J. Chen and J. Tang, "UAV-assisted data collection for wireless sensor networks with dynamic working modes," *Digit. Commun. Netw.*, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352864822002267, doi: 10.1016/j.dcan.2022.10.017.

[33] Y. Zeng and R. Zhang, "Energy-efficient UAV communication with trajectory optimization," *IEEE Trans. Wireless Commun.*, vol. 16, no. 6, pp. 3747–3760, Jun. 2017.

[34] D. D. Mrema and S. Shimamoto, "Performance of quadrifilar helix antenna on EAD channel model for UAV to LEO satellite link," in *Proc. Int. Conf. Collaboration Technol. Syst. (CTS)*, May 2012, pp. 170–175.

[35] V. Mnih et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, Feb. 2015.

[36] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," 2017, *arXiv:1710.10903*.

[37] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*.

[38] Y. Ran, H. Hu, X. Zhou, and Y. Wen, "DeepEE: Joint optimization of job scheduling and cooling control for data center energy efficiency using deep reinforcement learning," in *Proc. IEEE 39th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jul. 2019, pp. 645–655.

[39] D. Yi, X. Zhou, Y. Wen, and R. Tan, "Efficient compute-intensive job allocation in data centers via deep reinforcement learning," *IEEE Trans. Parallel Distrib. Syst.*, vol. 31, no. 6, pp. 1474–1485, Jun. 2020.

[40] Q. Cappart, T. Moisan, L.-M. Rousseau, I. Prémont-Schwarz, and A. Cire, "Combining reinforcement learning and constraint programming for combinatorial optimization," 2020, *arXiv:2006.01610*.

[41] Y. Zheng et al., "Forecasting fine-grained air quality based on big data," in *Proc. 21st ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2015, pp. 2267–2276.
[42] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "AirSim: High-fidelity visual and physical simulation for autonomous vehicles," in *Field and Service Robotics* (Springer Proceedings in Advanced Robotics), vol. 5, M. Hutter and R. Siegwart, Eds. Cham, Switzerland: Springer, 2018, pp. 621–635, doi: 10.1007/978-3-319-67361-5_40.
[43] M. Wang et al., "Deep graph library: A graph-centric, highly-performant package for graph neural networks," 2019, *arXiv:1909.01315*.
[44] Y. Wei and R. Zheng, "Informative path planning for mobile sensing with reinforcement learning," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Jul. 2020, pp. 864–873.

**Meiyi Yang** is currently pursuing the Ph.D. degree with the University of Electronic Science and Technology of China, Chengdu, China.

Her current research interests include artificial intelligence and wireless networks.

**Nianbo Liu** received the Ph.D. degree from the University of Electronic Science and Technology of China, Chengdu, China, in 2011.

He acts as a Research Assistant with The Hong Kong Polytechnic University, Hong Kong, in 2010. He is now a Research Associate with the School of Computer Science and Engineering, University of Electronic Science and Technology of China. His research interests include ad hoc networks and artificial intelligence.

**Yong Feng** (Member, IEEE) received the Ph.D. degree in information and communication engineering from the University of Electronic Science and Technology of China, Chengdu, China, in 2011.

He is currently a Professor with the Yunnan Key Laboratory of Computer Technology Applications, Kunming University of Science and Technology, Kunming, China. He has authored/coauthored three books and over 50 papers in peer-reviewed journals and conferences. His current research interests include the Internet of Things (Iot), blockchain, and deep bioadjust learning.

**Haigang Gong** received the M.S. degree from the University Electronic Science and Technology of China (UESTC), Chengdu, China, in 2003, and the Ph.D. degree from Nanjing University, Nanjing, China, in 2006.

He joined the Data Intensive Scalable Computing (DISCO) Laboratory, Nanyang Technological University, Singapore, from 2013 to 2014, as a Visiting Scholar. In 2006, he joined UESTC as a Teacher. He has published more than 80 articles and translated or edited two books as a partaker.

**Xiaoming Wang** received the B.S., M.S., and Ph.D. degrees from the University of Electronic Science and Technology of China, Chengdu, China, in 2001, 2004, and 2010, respectively.

Her main research directions are mobile big data, wireless sensor networks, and vehicular ad hoc networks.

**Ming Liu** received the Ph.D. degree from the Department of Computer Science and Technology, Nanjing University, Nanjing, China, in 2006.

In September 2006, he joined the College of Computer Science, University of Electronic Science and Technology of China (UESTC), Chengdu, China. He was promoted to Associate Professor and the Ph.D. Tutor in 2008 and December 2010, respectively.

Dr. Liu received the Excellent Doctoral Dissertation Award of Nanjing University Outstanding Ph.D. thesis in Jiangsu.