Timely Data Collection for UAV-based IoT networks: A Deep Reinforcement Learning Approach

Yingmeng Hu, Yan Liu, Aryan Kaushik, *Member, IEEE,*
Christos Masouros, *Senior Member, IEEE,* John Thompson, *Fellow, IEEE*

***Abstract*— In some real-time Internet of Things (IoT) applications, the timeliness of sensor data is very important for the performance of a system. How to collect the data of sensor nodes is a problem to be solved for an unmanned aerial vehicle (UAV) in a specified area, where different nodes have different timeliness priorities. To efficiently collect the data, a guided search deep reinforcement learning (GSDRL) algorithm is presented to help the UAV with different initial positions to independently complete the task of data collection and forwarding. First, the data collection process is modeled as a sequential decision problem for minimizing the average age of information or maximizing the number of collected nodes according to specific environment. Then, the data collection strategy is optimized by the GSDRL algorithm. After training the network using the GSDRL algorithm, the UAV has the ability to perform autonomous navigation and decision-making to complete the complexity task more efficiently and rapidly. Simulation experiments show that the GSDRL algorithm has strong adaptability to adverse environments, and obtains a good strategy for the UAV data collection and forwarding.**

***Index Terms*— Data collection, UAV trajectory optimization, age of information, deep reinforcement learning.**

I. Introduction

With the increasing development of the Internet of Things (IoT), the number of sensor nodes is growing explosively. The future application systems have stricter requirements on the timely delivery of the data collected from the sensor nodes [1]. For such applications, unmanned aerial vehicles (UAVs) can help to collect data from the sensor nodes (SNs) and then fly to the data center (DC) to deliver the data. UAVs have the advantages of rapid deployment, strong maneuverability and low cost [2]. Compared with the method of multi-hop data transmission [3], the UAV can flexibly adjust its position to improve communication environment. This helps to save energy and extend the battery lifetimes of the nodes. In addition, by constructing the communication systems between UAVs and ground terminals, and between UAVs, this helps to satisfy the needs of network services in various scenarios in the future. These include reliable and safe communication in public areas, network enhancement in hotspots, data collection in smart cities and improving network coverage in remote areas, etc., [4]–[8].

There are some real-time applications in the IoT systems, where the applications need to collect the surrounding environment information and monitor the status of the system periodically [9]. Those messages help the system to make intelligent decisions and control in a timely and effective way. For some applications with strict time constraints, outdated information will greatly reduce their effectiveness [10]. Due to the energy limitations of SNs and UAVs, these devices cannot carry out long-distance communication. According to different data timeliness of nodes, this paper focuses on how to plan the UAV's flight trajectory to efficiently complete the tasks of data collection and forwarding. In addition, the initial positions of an UAV are different every time. This causes a problem of great search complexity for the UAV trajectory optimization. Furthermore, the environment states may be changed at any time during the execution of the UAV's mission. However, it is difficult to solve these challenges using the traditional optimization algorithms.

With the development of artificial intelligence (AI) technology, machine learning algorithms have been applied to solve some more complex problems in practice. Among them, deep reinforcement learning (DRL) algorithms [11] have drawn lots of attention, which enable the UAV to operate autonomously and to deal with unpredictable conditions. However, due to large search space, classical DRL algorithms may only find inferior local solutions given the limited training time. To solve the issues, this paper proposes a guided search deep reinforcement learning (GSDRL) algorithm to find better suboptimal solutions efficiently. To satisfy the timeliness of high priority nodes given the limited UAV energy, the GSDRL algorithm helps the UAV to optimize the data collection

This work was supported in part by the National Science Fund for Distinguished Young Scholars under Grant 61901427 and in part by the New Generation of Information and Communication Technology Innovation under Grant Z211100004421019. The associate editor coordinating the review of this letter and approving it for publication was Q. Li. (Corresponding author: Yingmeng Hu, Yan Liu.)

Y. Hu and Y. Liu are with the China Satellite Network Innovation Co., Ltd.,Beijing, 100191, China (e-mail: huyingmeng@buaa.edu.cn, liuy15@tsinghua.org.cn). Aryan Kaushik is with the School of Engineering and Informatics, University of Sussex, Brighton BN1 9RH, UK (e-mail: aryan.kaushik@sussex.ac.uk). Christos Masouros is with Department of Electronic and Electrical Engineering, University College London, UK (e-mail: c.masouros@ucl.ac.uk). John Thompson is with the Institute for Digital Communications, School of Engineering, The University of Edinburgh, Edinburgh EH3 9JL, UK (e-mail: john.thompson@ed.ac.uk).

strategy and autonomously plan the flight trajectory according to the real-time environmental information.

A. Literature Review

Combined with terrestrial network communication systems, UAVs can help to solve the problem of the weak signal local coverage in IoT. It increases the capacity of the system, and also promotes the development of diversified IoT applications [12]. Thus, UAVs will play a vital important role in the next generation mobile communication systems. The applications of UAVs have also been widely investigated in academia and industry. The grant-free non-coherent index-modulation scheme combined with orthogonal frequency division multiplexing is investigated for applicant to unmanned aerial vehicle (UAV)-based massive IoT access in [13]. The performance evaluation of NOMA-enabled UAV networks is presented by adopting stochastic geometry to model the positions of UAVs and ground users in [14]. In addition, UAV's trajectory is optimized by deep reinforcement learning algorithm to collect data from IoT nodes in a practical 3D urban environment with imperfect channel state information in [15]. To minimize the interruption probability of relay network, a model is built by jointly optimizing the trajectory and the transmission power of the UAV in [16]. These works mainly focus on the energy consumption of the UAV and the SNs, but fail to consider the timeliness of sensor data in IoT.

Age of information (AoI) has been introduced to evaluate the freshness of information, which is the time it takes for the destination node to receive the packet produced by the source one. As a new important metric to measure the timeliness performance of the system, the AoI has attracted much attention about UAV-assisted data collection research. In order to minimize the system's AoI, a deep reinforcement learning algorithm is presented to optimize the UAV's trajectory in [17], where the data collection is forwarded to the data center via a link between the satellite and the UAV. By jointly optimizing the flight trajectory and the energy as well as service time allocation of packet transmission, a strategy is proposed to reduce the mean peak age in [18]. The UAV's flight trajectory is optimized by a deep reinforcement learning algorithm to minimize the SNs' average AoI in [19]. By optimizing the UAV trajectory, IoT devices scheduling, and channel allocation jointly, an AoI-energy-aware UAV-assisted data collection scheme is presented in [20] to reduce the expected AoI and energy consumption. The UAV acts as a mobile relay in [19] and [20]. However, when the communication distance between a BS and a sensor node (SN) is relatively far, the UAV needs to go back and forth between the BS and SNs, which causes a large amount of energy consumption and even a failed communication task. Different from [19] and [20], an energy-efficient data collection is presented to reduce the maximum energy consumption of all SNs in [21], where the UAV first collects the data of each SN one by one, and then forwards the data to the BS. Although this method is easy to be implemented, it is difficult to satisfy the timeliness requirements of high priority SNs in some applications.

B. Contributions

The timeliness of sensor data information is different in some IoT application scenarios. For example, very low latency is required in some virtual reality (VR) applications. However, for the forest environment monitoring, the system can tolerate longer delays relatively. The system may fail to meet the diversified requirements of the future services when the traditional data collection methods are adopted. Next, the initial position of the UAV is unfixed. For example, in the spatial crowdsourcing system [22]–[24], the initial position of an UAV undertaking a task is random. This brings some new challenges to optimize the UAV's trajectory. Besides, during the process of data collection, the BS may ask the UAV to repeatedly collect the message of the target nodes, or the energy consumed by the UAV increases suddenly due to extremely bad weather. These unpredictable factors may lead to some dynamic environment changes during the process of performing tasks. If the strategy of preplanned flight path is adopted for data collection, it will reduce the task execution efficiency and even result in a failed task due to the limited energy. Given the limited power of the UAV and SNs, the UAV needs to have the ability to make autonomous decisions and navigation in adverse conditions, i.e., when and where to collect and forward data. Using the deep reinforcement learning (DRL) algorithm [25], an agent can obtain this ability by continuously interactive learning from the environment.

An UAV data collection strategy based on DRL algorithm is proposed to minimize the average AoI or maximize the number of collected nodes. According to the environment and the state information on the UAV, the agent can make decisions by itself and complete the task in a specified area. Unlike the traditional data collection methods in [19], [20] and [21], the proposed scheme is more flexible, where the data of dynamic number of nodes can be collected together according to the specific environment, and then forwarded uniformly. The main contributions of this paper are summarized as follows:

1) This paper designs a dynamic data collection strategy based on deep reinforcement learning algorithm to efficiently collect the data of different priority nodes for an UAV with a random initial position in IoT. The UAV's flight trajectory is formulated autonomously to achieve the timeliness of high priority nodes and try its best to complete the data collection tasks of the ordinary priority nodes given the limited energy.
2) The method of action guidance is presented to assist the agent to find better actions quickly during the training process. It can not only accelerate the convergence speed of the network, also improves the system energy efficiency.
3) The proposed scheme achieves stronger autonomous decision-making ability to collect or forward the data of different priority nodes according to specific environment. Experimental results show that when compared with the fixed data collection strategy, the GSDRL algorithm can achieve better performance. It can ensure the timeliness of the Type II nodes, also reduces the average AoI of nodes.

TABLE I: Description of notations

Notations	Description
PL_L, PL_N	The path loss line-of-sight signal model and non-line-of-sight signal model
$PL_A^{U\to D}$	The average path loss between the UAV and the DC
P_R^D	The received power of the DC
P_T^{SN}	The transmission power of a SN
R_C, R_D	The UAV collection and forwarding rate
B_0	The system bandwidth
$A_{\pi(j),k}$	The AoI of a node $SN_{\pi(j),k}$
P_f, P_h^1, P_h^2	The power consumed by the UAV in flying data collecting and forwarding state
$\mathbf{G}_k$	Set of n_k $(1 \le n_k \le m_0)$ SNs
$\mathbf{\Omega}$	Set of UAV positions
$\mathbf{d}_0$, $\mathbf{d}_{DC}$	The UAV start and end positions
$\mathbf{d}_i^{(1)}, \mathbf{d}_i^{(2)}$	The locations of UAV to collect and forward the data of SN_i
$T_M\left(\mathbf{d}_l \to \mathbf{d}_{\pi(1),k}^{(1)}\right)$	The UAV transfer time from $\mathbf{d}_l$ to $\mathbf{d}_{\pi(1),k}^{(1)}$
$T_S^{sn}\left(SN_i, \mathbf{\Theta}\right)$	Data storage time in SN_i
$T_C\left(SN_{\pi(1),k}, \mathbf{\Theta}\right)$	The time to collect the data of $SN_{\pi(\cdot),k}$
$T_S^{uav}\left(SN_{\pi(\cdot),k}, \mathbf{\Theta}\right)$	The data storage time of $SN_{\pi(\cdot),k}$ on UAV
$T_S^{sn}\left(SN_{\pi(\cdot),k}, \mathbf{\Theta}\right)$	The data storage time of $SN_{\pi(\cdot),k}$ on SN
$T_U\left(SN_{\pi(\cdot),k}, \mathbf{\Theta}\right)$	The time to forward the data of $SN_{\pi(\cdot),k}$ to DC
m_0	Total number of SNs
d	Length of each grid
r_{DC}	Communication radius of the DC
r_{SN}	Communication radius of a SN
T_0	UAV maximum endurance time
$l_{\pi(i),k}$	Size of sensor data of $SN_{\pi(i),k}$
$\mathbf{t}_k$	UAV flight trajectory for $\mathbf{G}_k$
E_s	Total UAV energy consumed
E_u	Maximum capacity of the battery
$\mathbf{\Psi}_h$, $\mathbf{\Psi}_o$	High priority and ordinary nodes sets
$\lvert\mathbf{\Psi}_h\rvert$, $\lvert\mathbf{\Psi}_o\rvert$	Number of SNs in $\mathbf{\Psi}_h$ and $\mathbf{\Psi}_o$

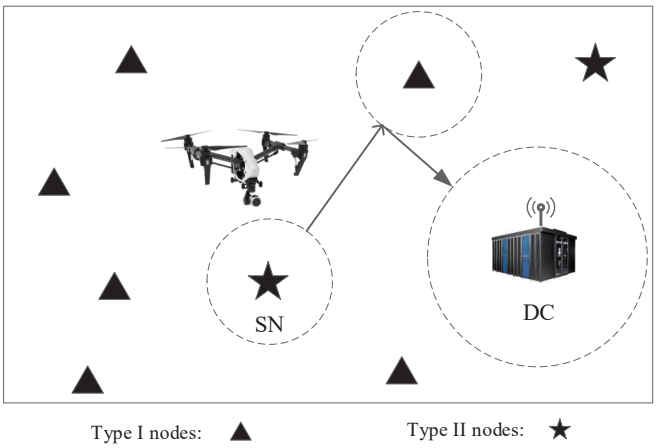

Fig. 1: A block diagram of UAV data collection and forwarding in IoT networks.

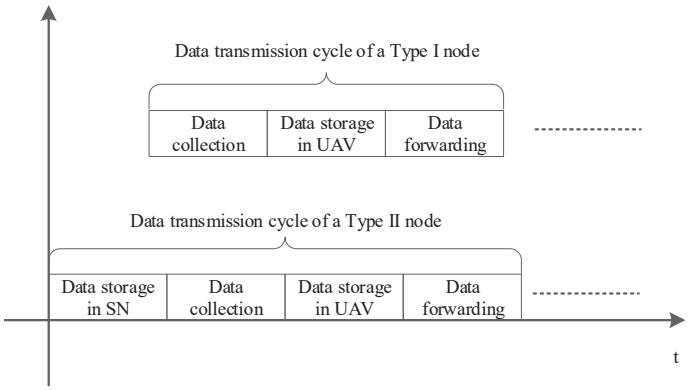

Fig. 2: The data transmission process for a Type I node and a Type II node.

The remainder of this paper is organized as follows: Section II introduces the channels between the UAV and the DC, and between the UAV and the SNs. The GSDRL algorithm is presented in Section III. Simulation results are discussed in Section IV. Finally, Section V summarizes the paper. The notations are illustrated in Table I in the paper.

II. System Model and Problem Formulation

The channel models are analyzed in this section. Then, we discuss the UAV's work mode and the corresponding power consumption. Finally, we formulate the data collection and forwarding model of the UAV.

A. System Model

As shown in Fig. 1, multiple SNs are randomly distributed in a certain area. According to the data generation method of nodes, we divide them into two types: Type I and Type II nodes[1]. The data of the Type I nodes is generated by a generate-at-will policy [18]. Only when the UAV is close to the SN, the SN will immediately generate updated data packets, and the data generation time is ignored. For the Type II nodes [26], their data has been generated and cached in their memories before the UAV departs, and the packet's timeliness is measured from the moment that the UAV takes off. Thus, the AoI of the Type II nodes will increase until the UAV forwards their data to the DC. Taking into account some actual factors, such as certain emergencies, the UAV is dispatched to preferentially collect the data of some special nodes and ensure the timeliness of the nodes. So we divide these nodes into two different sets, where the high priority nodes and the ordinary nodes are in Set Ψ_h and Ψ_o, respectively.

A data collection cycle of a SN is shown in Fig. 2. The UAV first flies to the vicinity of the SN, and collects its data. Then, the data are stored in the UAV's memory. When the UAV moves to the vicinity of the DC, it begins to forward the data. Different from Type I nodes, the AoI of Type II nodes also includes the time when the data is stored in the SN stack. We use the AoI as a metric to evaluate the freshness of information of a SN. When the data of a SN i has been forwarded by the UAV at time t, we can obtain the AoI of the SN as [26]

$$A_i(t) = t - u_i(t), \tag{1}$$

where $u_i(t)$ refers to the generation time of a packet at the SN i. If the UAV does not complete the data collection task due to insufficient energy, the AoI of an unloaded SN is considered to be the maximum endurance time of the UAV in the paper.

[1]We assume that Type II nodes all are high priority nodes in the paper.

1) The analysis of channel model: The path loss of line-of-sight (LoS) signal model and non-line-of-sight (NLoS) signal model are, respectively [27],

$$PL_L = 20\log\left(f_c\right)+20\log\left(\frac{4\pi}{v_c}\right)+20\log\left(d_{UD}\right)+\kappa_{LoS}, \quad (2)$$

and

$$PL_N = 20\log\left(f_c\right)+20\log\left(\frac{4\pi}{v_c}\right)+20\log\left(d_{UD}\right)+\kappa_{NLoS}, \quad (3)$$

where f_c is the carrier frequency, v_c is the speed of light, d_{UD} the distance between the UAV and the DC, κ_{LoS} is the LoS model's path loss, and κ_{NLoS} is NLoS model's path loss. The probability of the LoS model is given by,

$$P_L = \frac{1}{1+z_0\exp\left(-z_1\left(\frac{180}{\pi}\sin^{-1}\left(\frac{H_0}{d_{UD}}\right)-z_0\right)\right)}, \quad (4)$$

where H_0 is the altitude of the UAV above the ground, z_0 and z_1 are the parameters related to the specific environment [28]. Thus, the average path loss (dB) between the UAV and the DC can be written as

$$PL_A^{U\to D} = P_L \times PL_L + \left(1-P_L\right)PL_N. \quad (5)$$

The received power (dBW) of the DC can be obtained as follows:

$$P_R^D = 10\lg P_T^U - PL_A^{U\to D}, \quad (6)$$

where P_T^U is the transmission power of the UAV. As the UAV has a good maneuverability, the communication distance between the UAV and a SN is relatively shorter, and only the LoS model is considered between the UAV and the SN. We assume d_{SU} to be the distance from the SN to the UAV, and the path loss $PL_A^{S\to U}$ can be obtained by replacing d_{UD} with d_{SU} in (2). Thus, we can obtain the UAV receiving power (dBW) when it is in collection state as

$$P_R^U = 10\lg P_T^{SN} - PL_A^{S\to U}, \quad (7)$$

where P_T^{SN} is the transmission power of a SN. In addition, we also assume that the channel gain h follows the Rayleigh distribution, then the probability density function (PDF) and the cumulative distribution function (CDF) of $x = |h|^2$ are respectively,

$$f\left(x\right) = \frac{1}{\overline{P_r}}\exp\left(-\frac{x}{\overline{P_r}}\right), \quad (8)$$

and

$$F\left(x\right) = 1-\exp\left(-\frac{x}{\overline{P_r}}\right), \quad (9)$$

where $\overline{P_r}$ is the average received power of a signal. The collection and forwarding rate of the UAV at time t are, respectively,

$$R_C\left(t\right) = B_0\log_2\left(1+\frac{P_T^{SN}|h\left(t\right)|^2}{P_{\sigma^2}}\right), \quad (10)$$

and

$$R_D\left(t\right) = B_0\log_2\left(1+\frac{P_T^{U}|h\left(t\right)|^2}{P_{\sigma^2}}\right), \quad (11)$$

where B_0 is the signal bandwidth, P_{σ^2} is the noise power. From the Appendix, we can obtain that the average rate $E\left(R_C\right)$ as

$$E\left(R_C\right) = -\frac{B_0}{\ln 2}e^{\frac{P_{\sigma^2}}{P_T^{SN}P_R^U}}Ei\left(-\frac{P_{\sigma^2}}{P_T^{SN}P_R^U}\right), \quad (12)$$

where $Ei\left(\bullet\right)$ denotes the one-argument exponential integral function. Similarly, combining (6), (11) and (35), we can also achieve the average rate $E\left(R_D\right)$ as

$$E\left(R_D\right) = -\frac{B_0}{\ln 2}e^{\frac{P_{\sigma^2}}{P_T^{U}P_R^D}}Ei\left(-\frac{P_{\sigma^2}}{P_T^{U}P_R^D}\right). \quad (13)$$

2) UAV work mode: Given a random initial position, the UAV will fly to a SN to collect its data. When the data collection process of the SN is over, the UAV needs to move to another SN to continue collecting data, or to forward the data to the DC. The UAV will repeat the above process until the data of all SNs are collected and forwarded successfully, or the UAV is forced to finish its flight due to energy shortage. To simplify the analysis of the model, this paper only considers two modes of the UAV, namely the flight mode and the hovering mode. The hovering mode includes data collection and data forwarding states.

(1) Flight Mode

When the UAV is in flight mode, the power consumed by the UAV is [29]

$$P_f\left(V\right) = \frac{\delta}{8}\rho s A\Omega^3R^3\left(1+\frac{3V^2}{U_{tip}^2}\right)+ \left(1+k\right)\frac{W^{3/2}}{\sqrt{2\rho A}}\left(\sqrt{1+\frac{V^4}{4v_0^4}}-\frac{V^2}{4v_0^2}\right)^{0.5}+\frac{1}{2}d_0\rho s A V^3, \quad (14)$$

where V is the flying speed of the UAV, ρ is the air density, and the other parameters[2] are mostly related to the UAV, such as R and W are the rotor radius and the aircraft weight of the UAV, respectively.

(2) Hovering Mode

When the UAV collects the data of a SN, the power consumed by the UAV is [29]

$$P_h^1 \approx \frac{\delta}{8}\rho s A\Omega^3R^3+\left(1+k\right)\frac{W^{3/2}}{\sqrt{2\rho A}}. \quad (15)$$

If the UAV forwards the data to the DC, it not only needs to overcome its own gravity, wind and other factors, it also consumes certain power to transmit data, so the power consumed by the UAV is

$$P_h^2 = P_h^1 + P_c, \quad (16)$$

where P_c is the transmission power.

B. Problem Formulation

An UAV can select different SNs for data collection according to the specific environment, and forward the data at an appropriate time. In other words, it can dynamically select several nodes for data collection, and then forward them together. To illustrate the process of data collection and forwarding more clearly, we divide it into several groups and

[2]They can be referred to Table I in [29].

put the SNs, which are forwarded to the DC together, into the same group.[3]

There are m_0 SNs scattered randomly in a certain area, where the UAV needs to autonomously formulate its trajectory and make decisions by itself to collect or forward the data. We assume that given a strategy π, the UAV needs N_0 times of offloading to complete the task. These SNs are divided into N_0 groups, and it is $\mathbf{\Phi} = \{\mathbf{G}_1, \mathbf{G}_2, \cdots, \mathbf{G}_{N_0}\}$. A group $\mathbf{G}_i = \left\{SN_{\pi(1),i}, SN_{\pi(2),i}, \cdots, SN_{\pi(n_i),i}\right\}$[4] consists of $n_i\,(1 \le n_i \le m_0)$ SNs, where $SN_{\pi(j),i}, 1 \le j \le n_i$ denotes the *j*-th collected node in the *i*-th group. During the process of performing tasks, the UAV needs to choose some locations to execute two different actions, namely collecting or forwarding data. For m_0 SNs, there is a position set $\mathbf{\Omega} = \{\mathbf{d}_0, \mathbf{D}_1, \mathbf{D}_2, \cdots, \mathbf{D}_{m_0}, \mathbf{d}_{DC}\}$, where $\mathbf{d}_0$ and $\mathbf{d}_{DC}$ are the start and end point of the UAV, respectively. The parameter $\mathbf{D}_i = \left\{\mathbf{d}_i^{(1)}, \mathbf{d}_i^{(2)}\right\}, 1 \le i \le m_0$ is the position set for SN_i, where $\mathbf{d}_i^{(1)}$ and $\mathbf{d}_i^{(2)}$ are locations of the UAV when it collects data from SN_i and forwards SN_i's data to the DC, respectively. The UAV will forward the data only after the data of all nodes in the same group are all collected. When SN_i is not the last one to be collected in the group, we will set $\mathbf{d}_i^{(2)} = \emptyset$.

As shown in Fig. 3, we take SNs in the *k*-th group as an example to briefly explain the data collecting and forwarding process of the UAV. We assume that there are $n_k\,(1 \le n_k \le m_0)$ SNs in $\mathbf{G}_k$ to be collected and forwarded. It is $\mathbf{G}_k = \left\{SN_{\pi(1),k}, SN_{\pi(2),k}, \cdots, SN_{\pi(n_k),k}\right\}$. At τ_{k-1}, the UAV starts to fly from $\mathbf{d}_l$ to $\mathbf{d}_{\pi(1),k}^{(1)}$. The parameter $\mathbf{d}_l$ is the UAV position where it forwards the data of SNs in $\mathbf{G}_{k-1}$, it is $\mathbf{d}_l = \mathbf{d}_{\pi\left(n_{k-1}\right),k-1}^{(2)}$. It takes $T_M\left(\mathbf{d}_l \to \mathbf{d}_{\pi(1),k}^{(1)}\right)$ seconds for the UAV to transfer from $\mathbf{d}_l$ to $\mathbf{d}_{\pi(1),k}^{(1)}$, and then collects the data of $SN_{\pi(1),i}$. The UAV spends $T_C\left(SN_{\pi(1),k}, \mathbf{\Omega}\right) = \frac{l_{\pi(1),k}}{R_C}$ seconds to collect the data of $SN_{\pi(1),k}$, where R_C is the collection rate.

Next, the UAV moves to $\mathbf{d}_{\pi(2),k}^{(1)}$ to collect the data of $SN_{\pi(2),i}$. For the Type II nodes, as their data have been generated and stored in their memory, the AoI of the node $SN_{\pi(2),k}$ includes the time of data storage in SN. The UAV collects the data from the remaining nodes in $\mathbf{G}_k$ by the same way. The parameter $T_S^{uav}\left(SN_{\pi(1),k}, \mathbf{\Omega}\right)$ is the storage time of the data of $SN_{\pi(1),k}$ in the UAV, and $T_U\left(SN_{\pi(1),k}, \mathbf{\Omega}\right) = \frac{l_{\pi(1),k}}{R_D}$ is the time required to forward the data of the node to the DC, where R_D is the forwarding rate. When the data of $SN_{\pi(1),k}$ is collected, the UAV will fly from $\mathbf{d}_{\pi(1),k}^{(1)}$ to $\mathbf{d}_{\pi(2),k}^{(1)}$ and spends $T_C\left(SN_{\pi(2),k}, \mathbf{\Omega}\right)$ seconds to collect the data of $SN_{\pi(2),k}$. Similarly, the UAV continues to collecting the data until the data of all the SNs in $\mathbf{G}_k$ are collected. Finally, it flies from $\mathbf{d}_{\pi(n_k),k}^{(1)}$ to $\mathbf{d}_{\pi(n_k),k}^{(2)}$ and begins to forward the data. From Fig. 2 and Fig. 3, we can also obtain the AoI of a Type II node[5]

$$\begin{aligned} A_{SN_{\pi(j),k}}(\mathbf{\Theta}) = & T_S^{sn}\left(SN_{\pi(j),k}, \mathbf{\Theta}\right) + T_C\left(SN_{\pi(j),k}, \mathbf{\Theta}\right) \\ & + T_S^{\text{uav}}\left(SN_{\pi(j),k}, \mathbf{\Theta}\right) + T_U\left(SN_{\pi(j),k}, \mathbf{\Theta}\right) \end{aligned}, \tag{17}$$

where $T_S^{sn}\left(SN_{\pi(j),k}, \mathbf{\Omega}\right)$ is the data storage time in $SN_{\pi(j),k}$, which starts timing from the moment the UAV takes off. In other words, it is $T_S^{sn}\left(SN_{\pi(j),k}, \mathbf{\Omega}\right) = 0$ for a Type I node.

It can be observed from Fig. 3, the UAV's flight trajectory is $\mathbf{d}_l \to \mathbf{d}_{\pi(1),k}^{(1)} \to \mathbf{d}_{\pi(2),k}^{(1)} \to \cdots \to \mathbf{d}_{\pi(n_k),k}^{(1)} \to \mathbf{d}_{\pi(n_k),k}^{(2)}$. After the data of $\mathbf{G}_{k-1}$ is forwarded, the UAV will collect the data of $\mathbf{G}_k$. Thus, the total energy consumed E_s by the UAV is

$$E_s = \left(\begin{array}{l} P_f(V)\, T_b + \\ \sum\limits_{k=1}^{N_0} \left(\begin{array}{l} P_f(V)\, T_M\left(\mathbf{t}_k, \mathbf{\Omega}\right) + \\ P_h^1 \sum\limits_{j=1}^{n_k} T_C\left(SN_{\pi(j),k}, \mathbf{\Omega}\right) G_{\left(SN_{\pi(j),k}\right)}(\mathbf{\Omega}) + \\ P_h^2 \sum\limits_{j=1}^{n_k} T_U\left(SN_{\pi(j),k}, \mathbf{\Omega}\right) G_{\left(SN_{\pi(j),k}\right)}(\mathbf{\Omega}) \end{array} \right) \end{array} \right), \tag{18}$$

where T_b is the time required for the UAV to back to the DC. The parameter $T_M\left(\mathbf{t}_k, \mathbf{\Omega}\right)$ is the flight time required for the UAV to follow trajectory $\mathbf{t}_k$ in $\mathbf{G}_k$. The parameters $T_C\left(SN_i, \mathbf{\Omega}\right)$ and $T_U\left(SN_i, \mathbf{\Omega}\right)$ are the time for the UAV to collect and forward the data of $SN_{\pi(j),k}$, respectively. The parameters $P_f(V)$, P_h^1 and P_h^2 are the UAV consumed power in different states. They have been discussed in (14), (15) and (16).

Given the limited energy of an UAV with a random initial position, the UAV is dispatched to collect and forward the data of the SNs in the specified area. However, for the different priorities of nodes, the UAV will obtain two different optimization strategies due to the limited residual energy and the AoI requirement of Type II nodes in $\mathbf{\Psi}_h$. When the UAV is close to the SNs, it has the ability to complete the task of all the nodes in $\mathbf{\Psi}_h$ and $\mathbf{\Psi}_o$, simultaneously. It is $N = m_0$, so we aim to minimize the average AoI and reduce the energy consumption by optimizing the UAV data collection strategy. Besides, when the UAV is at the boundary of the area or far away from the SNs, it may fail to collect the data of all the nodes. At this time, the UAV has to give priority to completing the tasks of the high priority nodes in $\mathbf{\Psi}_h$ and tries its best to collect the ordinary nodes in $\mathbf{\Psi}_o$ as many as possible. So it is $N < m_0$. Thus, according to the environment, the objective function can be written by

$$\left\{ \begin{array}{l} \left(\begin{array}{l} \min\limits_{\mathbf{\Omega}} E_c = \left(\begin{array}{c} P_f T_f(\mathbf{\Omega}) + \\ P_c T_c(\mathbf{\Omega}) + P_u T_u(\mathbf{\Omega}) \end{array} \right), \\ min\limits_{\mathbf{\Omega}} \frac{\sum_{i=1}^{N} A_{(SN_i)}(\mathbf{\Omega})}{N} \end{array} \right), N = m_0 \\ \left(\begin{array}{l} \max\limits_{\mathbf{\Omega}} N = |\mathbf{\Psi}_h| + \sum\limits_{i=1}^{|\mathbf{\Psi}_o|} G_{(SN_i)}(\mathbf{\Omega}), \\ min\limits_{\mathbf{\Omega}} \frac{\sum_{i=1}^{N} A_{(SN_i)}(\mathbf{\Omega})}{|\mathbf{\Psi}_h| + |\mathbf{\Psi}_o|} \\ \sum\limits_{i=1} G_{(SN_i)}(\mathbf{\Omega}) \end{array} \right), N < m_0 \end{array} \right. \tag{19}$$

[3]In fact, the SNs do not need to be grouped before the UAV departures. According to the specific conditions, the UAV will make autonomous decisions to complete the task, including where to collect the data, where and when to forward the data, which SNs are forwarded together, which one should be picked up first and so on. This paper just takes the grouping method to describe the data collection and forwarding process for the simplification of the analysis.

[4]The subscript $\pi(\cdot)$ is the ordered index number of SNs.

[5]The AoI of the Type I nodes are set as T_0 when the nodes have been collected but cannot be forwarded to the DC. The AoI of the Type II nodes that have not been forwarded are also set as T_0.

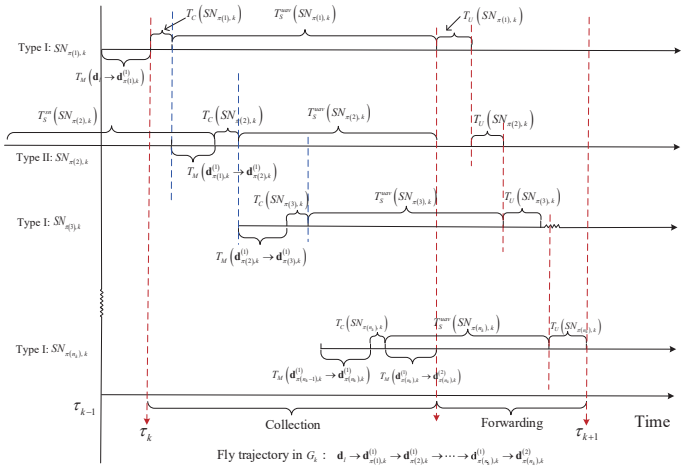

Fig. 3: The UAV data collecting and forwarding process of the SNs in $\mathbf{G}_k$.

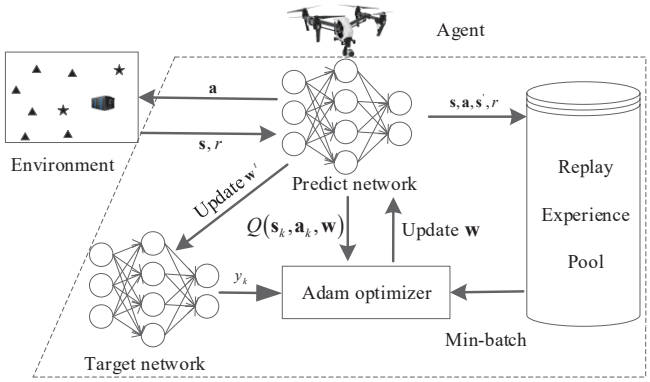

Fig. 4: The diagram of UAV data collection and forwarding based on the DRL algorithm.

$$
\begin{aligned}
&Subject \quad to: \\
&C1: A_{(SN_i)}(\mathbf{\Omega}) = \\
&\left(\begin{array}{c} T_S^{sn}(SN_i, \mathbf{\Omega}) + T_C(SN_i, \mathbf{\Omega}) \\ +T_S^{uav}(SN_i, \mathbf{\Omega}) + T_U(SN_i, \mathbf{\Omega}) \end{array} \right), SN_i \in \{\mathbf{\Psi}_h, \mathbf{\Psi}_o\} \\
&C2: 0 \le A_{(SN_j)}(\mathbf{\Omega}) \le T_{SN_j}^{th}, SN_j \in \mathbf{\Psi}_h, \\
&C3: E_s \le E_u, \\
&C4: \left\| \mathbf{d}_i^{(2)} - \mathbf{d}_{SN_i} \right\| \le r_{SN_i}, \left\| \mathbf{d}_i^{(2)} - \mathbf{d}_{DC} \right\| \le r_{DC}, 1 \le i \le m_0, \\
&C5: G_{(SN_i)}(\mathbf{\Omega}) = \{0, 1\}.
\end{aligned} \tag{20}
$$

where $|\mathbf{\Psi}_h|$ and $|\mathbf{\Psi}_o|$ are the number of SNs in $\mathbf{\Psi}_h$ and $\mathbf{\Psi}_o$, respectively. The parameters $T_f(\mathbf{\Omega})$ and $T_c(\mathbf{\Omega})$ are the total flight time and data collecting time of the UAV, respectively. The parameter $A_{(SN_i)}(\mathbf{\Omega})$ is the AoI of SN_i in (C1). Different from the nodes in $\mathbf{\Psi}_o$, the nodes in $\mathbf{\Psi}_h$ have stricter requirements for information timeliness, the AoI of the nodes in $\mathbf{\Psi}_h$ is less than or equal $T_{SN_j}^{th}$ in (C2). The constraint (C3) means that the total energy consumed of the UAV E_s should not exceed the maximum capacity E_u of the battery. Besides, the constraints (C4) set the communication range between the UAV and the DC, between SN_i and the UAV, where $\mathbf{d}_{SN,i}$ and $\mathbf{d}_{DC}$ are the coordinates of the SN_i and the DC, respectively. We also define an indicator function $G_{(SN_i)}(\mathbf{\Omega})$ in (C5). After the data of SN_i is collected and forwarded successfully, we set as $G_{(SN_i)}(\mathbf{\Omega}) = 1$. Otherwise, it is $G_{(SN_i)}(\mathbf{\Omega}) = 0$.

We assume that m_0 SNs are randomly scattered in a rectangle with an area of $L \times L$. The region is partitioned to grids, and the area of each grid is $d \times d$. It is assumed that the UAV only stays above the center of a grid when it collects data from the sensors. For a sensor SN_i, the locations of the UAV $\mathbf{d}_i^{(1)}$ and $\mathbf{d}_i^{(2)}$ all have $2(L/d)^2$ possibilities. Thus, the task complexity of the UAV is $\mathrm{O}\left(2^{l_0}(L/d)^{2l_0}\right)$, where $l_0 = m_0 + N_0$ is the number of actions the UAV takes. To achieve the timeliness of Type II nodes, the number of groups N_0 is also a random number due to the autonomous decision-making by the UAV according to the environment. Therefore, it is difficult to solve this problem using traditional trajectory optimization methods. The data collection and forwarding process of the UAV can be modeled as a sequential decision problem. Then, a guided search method based DRL algorithm is introduced to solve it. In the next section, we focus on the implementation of the proposed scheme.

III. The GSDRL Algorithm

This section first introduces the basic principle of the DRL algorithm and the process of network parameters to be updated. Then, a guiding fast search algorithm is proposed to improve the convergence speed of the network. Next, we present the implementation process of the proposed scheme in detail.

A. DRL Algorithm

An agent interacts with the environment in a trial-and-error way in the DRL algorithm continually. So it obtains the optimal strategy by maximizing the accumulation of rewards. The framework of UAV data collection and forwarding based on DRL algorithm is shown in Fig. 4. The basic framework of the DRL algorithms mainly consists of two parts, an agent (UAV) and the environment. The agent includes prediction network, target network, an optimizer, and a replay experience pool. The agent interacts with the environment with three parameters: state $\mathbf{S}$, action $\mathbf{A}$, and reward R. For the application scenario of UAV data collection and forwarding in the paper, we define as follows,

1) State space $\mathbf{S}$*:* The parameter $\mathbf{S} = \{\mathbf{s}\}$ represents the state space of the UAV, where $\mathbf{s} = \{\mathbf{B}, \mathbf{C}, \mathbf{D}, \mathbf{Y}, e, \eta\}$ includes the information about the UAV data collection, data forwarding, the AoI of a SN, the current position of the UAV, and the residual energy. The parameters $\mathbf{B} = \{b_i\}, \mathbf{C} = \{c_i\}$ are the states of the collection and forwarding of the SNs, respectively, where $b_i, c_i \in \{0, 1\}, 1 \le i \le m_0$. When it is b_i=1, the data of the i-th node has been collected. Otherwise, it means that the data of the node has not been collected. The variable c_i has a similar definition. The vector $\mathbf{D}$ represents the set of the average AoI of all the SNs. The dimensions of $\mathbf{B}$, $\mathbf{C}$ and $\mathbf{D}$ are all m_0. The parameter $\mathbf{Y}$ denotes the current position of the UAV, and e is the remaining power of the UAV. An indicator variable $\eta \in \{-1, 1\}$ describes the work model of the UAV. For example, if η=1, the UAV is in data forwarding state. Otherwise, the UAV is in data collection state.

2) Action space $\mathbf{A}$*:* The parameter $\mathbf{A} = \{\mathbf{a}\}$ is the action set of the UAV, where $\mathbf{a} = \{\mathbf{E}, \eta\}$ and $\mathbf{E}$ is the current coordinate of the UAV. According to the current environment, the UAV selects an appropriate location for data collection or forwarding. The agent selects an action using the greedy algorithm. Both exploration and exploitation methods are adopted by the agent to select an action. When the number of iterations is small, and the agent has limited information about the environment, it will randomly choose an action using exploration method. With the increase of the number of interactions with the environment, the agent continues to accumulate experience and autonomously rectifies its behavior to obtain a larger reward. We call this approach exploitation. As the agent's accumulated experience increases during the training process, the probability of exploration becomes smaller, the probability of exploitation for the agent gets larger, the convergence speed for the network becomes faster. However, it will get a poor local solution in most cases. Conversely, the network parameters converge slowly and the system needs to take a long time to find a good solution.

3) Reward R*:* The parameter R denotes the reward of the agent when it interacts with the environment. During the learning process, we need a reward to evaluate the action the agent selected. Different actions will usually get different rewards, which stimulates the agent to choose a better action. The reward function plays an important role in the convergence speed of the algorithm, and also directly affects the optimal solution. When the UAV is close to SNs to collect data, or the DC to forward the data, the agent will get a positive reward. Under the same conditions, if an action chosen by the agent can make the average AoI or the consumed energy become smaller, the agent will achieve a larger positive reward. If it choose an unreasonable action, such as the data collection and forwarding conditions are not satisfied, it will get a negative reward. So we define the reward as follows:

$$R = \begin{cases} \varpi_1\chi_1 + \varpi_2\chi_2 + \varpi_3\chi_3, flag = 1 \\ -\left(\varpi_1\chi_1 + \varpi_2\chi_2 + \varpi_3\chi_3\right), flag = 0 \end{cases}, \tag{21}$$

where ϖ_1, ϖ_2, and ϖ_3 are weight parameters. The functions χ_1, χ_2, and χ_3 are, respectively,

$$\chi_1 = 1 - \frac{\sum_{j=1}^{|\mathbf{\Psi}_h|} A_{(SN_j)}}{T_0 \sum_{j=1}^{|\mathbf{\Psi}_h|} G\left(SN_j\right)}, \tag{22}$$

$$\chi_2 = 1 - \frac{\sum_{j=1}^{|\mathbf{\Psi}_o|+|\mathbf{\Psi}_h|} A_{(SN_j)}}{T_0 \sum_{j=1}^{|\mathbf{\Psi}_o|+|\mathbf{\Psi}_h|} G\left(SN_j\right)}, \tag{23}$$

$$\chi_3 = (1 - \frac{E_s}{E_u}) \sum_{j=1}^{|\mathbf{\Psi}_o|+|\mathbf{\Psi}_h|} G\left(SN_j\right), \tag{24}$$

where T_0 represents the maximum endurance time of the UAV. The expression $\sum_{j=1}^{|\mathbf{\Psi}_h|} A_{(SN_j)} \Big/ \sum_{j=1}^{|\mathbf{\Psi}_h|} G\left(SN_j\right)$ denotes the average AoI of high priority nodes in χ_1. The expression $\sum_{j=1}^{|\mathbf{\Psi}_o|+|\mathbf{\Psi}_h|} G\left(SN_j\right)$ is the sum number of the nodes in $\mathbf{\Psi}_o$ and $\mathbf{\Psi}_h$, the data of which have been collected and forwarded successfully, $\sum_{j=1}^{|\mathbf{\Psi}_o|+|\mathbf{\Psi}_h|} A_{(SN_j)} \Big/ \sum_{j=1}^{|\mathbf{\Psi}_o|+|\mathbf{\Psi}_h|} G\left(SN_j\right)$ denotes the average AoI of all the nodes in χ_2. The expression $(1 - E_s/E_u)$ denotes the remaining power ratio in χ_3. If it is $flag = 1$, it means that the UAV is in collection or forwarding states and will obtain a positive reward. Otherwise, the action selected by the UAV cannot satisfy the communication conditions, so it will get a negative reward. Besides, if the UAV fails to back to the DC, it also will get a negative reward.

By adding the experience replay pool and the target network, the DRL algorithm tries to solve the convergence problem of the network parameters. The samples obtained are stored in the replay pool by the interaction between the agent and the environment. During the training process, a batch of samples are randomly selected from the stack for the parameters to be updated. This breaks the correlation between the samples. According to the state $\mathbf{s}$, the agent will select an action $\mathbf{a}$. After the UAV executes the action $\mathbf{a}$, we will get the next state $\mathbf{s}'$. In a similar way, the UAV will select different actions according to different states until the UAV completes the task or it runs out of the available energy. As shown in Fig. 4, we use an optimizer to update the parameters of the current network after each interaction. Next, we discuss the process of the network parameters to be updated.

B. Network Parameters Update

To maximize the long-term cumulative benefits, the agent interacts with the environment to update the state-action value $Q\left(\mathbf{s}, \mathbf{a}\right)$ by the method of trial-and-error. The process of the updated parameters in DRL algorithm is

$$Q\left(\mathbf{s}, \mathbf{a}\right) \leftarrow Q\left(\mathbf{s}, \mathbf{a}\right) + \beta\left(R + \lambda \max_{\mathbf{a}'} Q\left(\mathbf{s}', \mathbf{a}'\right) - Q\left(\mathbf{s}, \mathbf{a}\right)\right), \tag{25}$$

where $\beta \in (0, 1)$ is learning rate, $\lambda \in (0, 1)$ is discount factor. The parameter $\mathbf{s}'$ is the next state after the agent executes the action $\mathbf{a}$. From (25), we can get that the larger the discount factor λ, the more significantly the $Q\left(\mathbf{s}, \mathbf{a}\right)$ will be affected by the future returns. On the contrary, it is only affected by the immediate interests. Based on the principle of the DRL algorithms, the GSDRL algorithm uses deep neural networks $Q\left(\mathbf{s}, \mathbf{a}, \mathbf{w}\right)$ to approach $Q\left(\mathbf{s}, \mathbf{a}\right)$. To improve the performance stability of the system, the proposed scheme also has two networks including the target network and the current network. They have the same structure but different parameters. The state-action values from the target network and the current network are $Q^t\left(\mathbf{s}, \mathbf{a}, \mathbf{w}^t\right)$ and $Q\left(\mathbf{s}, \mathbf{a}, \mathbf{w}\right)$, where $\mathbf{w}^t$ and $\mathbf{w}$ are the parameters of the corresponding networks, respectively. Finally, a small batch stochastic gradient descent algorithm [30] is adopted to solve $\mathbf{w}$. We define the loss function as

$$\begin{aligned} L\left(\mathbf{w}\right) &= E\left(\left(g_k - Q\left(\mathbf{s}_k, \mathbf{a}_k, \mathbf{w}\right)\right)^2\right) \\ &= \frac{1}{N_1}\sum_{k=1}^{N_1}\left(g_k - Q\left(\mathbf{s}_k, \mathbf{a}_k, \mathbf{w}\right)\right)^2, \end{aligned} \tag{26}$$

where

$$g_k = R_k + \lambda Q^t\left(s'_k, \max_{a'} Q\left(s'_k, a'_k, \mathbf{w}\right), \mathbf{w}^t\right). \tag{27}$$

So the gradient of $\mathbf{w}$ can be obtained as follows

$$\nabla_{\mathbf{w}} L\left(\mathbf{w}\right) = E\left(\left(g_k - Q\left(s_k, a_k, \mathbf{w}\right)\right)\nabla_{\mathbf{w}} Q\left(s, a, \mathbf{w}\right)\right). \tag{28}$$

The parameter $\mathbf{w}$ will be updated by

$$\mathbf{w} \leftarrow \mathbf{w} + \beta\nabla_{\mathbf{w}} L\left(\mathbf{w}\right). \tag{29}$$

As the UAV adopts a dynamic data collection and forwarding strategy, it can collect one or more SNs before forwarding together independently. Due to the random starting positions for the UAV, the number of actions selected by the UAV is also different to complete the mission every time. Thus, the search space of the task is very large. We also discussed this issue in Section II-B. Taking the traditional DRL algorithms, the agent may get the inferior local solution within a limited search time. Next, in order to improve the search efficiency, we propose a fast search algorithm to guide the UAV's movements.

C. Guiding Action

As the number of iterations increases, the agent continuously learns the experience from the environment and makes corresponding decisions based on the real-time environment. To achieve the data timeliness of a SN, the agent usually forwards data as soon as possible at the signal coverage edge of the DC. The agent also flies to the signal coverage edge of a SN to collect the data. Next, we take the UAV data collection as an example to explain the process. First, we assume that the agent is in (x_0, y_0) at time t_0. It will fly to the point (x'_1, y'_1) at the next time t_1, where it just finds an uncollected SN in (x_s, y_s). In this case, the agent does not immediately perform the data collection action. It will rectify the action with a probability P_τ. We will further discuss it in Section IV. Next, we will explain the process of finding the revised action. We only adjust the UAV position, but do not modify the parameter η. In other words, we do not alter the state of the UAV such as data collection $(\eta = -1)$ or forwarding $(\eta = 1)$, and just revise the UAV position. By combining

Algorithm 1 The training process of the GSDRL algorithm.

Initialization: Randomly initialize all the parameters including the current network and the target network, and clear the replay experience pool $\mathbf{\Theta}$.
for episode $i = 1$ to $I_{\max}$ **do**
  Randomly initialize the UAV's location, state $\mathbf{s}$ and the energy $e = E_u$.
  for epoch $j = 1$ to $J_{\max}$ **do**
    Step 1: If $P_{\max} < \Delta \times i \times (j+1)$, it is P_ε=$P_{\max}$. Otherwise, it is P_ε=$\Delta \times i \times (j+1)$. The agent chooses an action $\mathbf{a} = \arg\max_{\mathbf{a}} Q(\mathbf{s}, \mathbf{a}, \mathbf{w})$ with the probability P_ε. Otherwise, it selects a random action $\mathbf{a}$.
    Step 2: Rectify the location of the UAV with the probability P_τ according to (30).
    Step 3: Execute $\mathbf{a}$ given the state $\mathbf{s}$ to get the next state $\mathbf{s}'$ and the reward R.
    Step 4: Restore $\{\mathbf{s}, \mathbf{a}, \mathbf{s}', R\}$ in $\mathbf{\Theta}$ and replace the current state $\mathbf{s}$ with $\mathbf{s}'$.
    Step 5: Randomly select m_0 samples from $\mathbf{\Theta}$, and calculate g_j according to (31).
    Step 6: Build a loss function $\frac{1}{N_1}\sum_{j=1}^{N_1}\left(g_j - Q(S_j, a_j, \mathbf{w})\right)^2$ and update $\mathbf{w}$ using the stochastic gradient descent method.
    Step 7: If it is $i\%\xi = 0$, the parameters of the target network is updated with $\mathbf{w}^t = \mathbf{w}$.
    Step 8: Terminate the episode if $\mathbf{s}'$ is the terminal state.
  end for
end for

the starting point (x_0, y_0) with the position of the target SN (x_s, y_s), the intersection point of the signal coverage boundary is obtained,

$$\begin{cases} x_1 = x_0 \pm r_{SN}\sqrt{\frac{(x_s-x_0)^2}{(x_s-x_0)^2+(y_s-y_0)^2}}, \\ y_1 = y_0 + \frac{y_s-y_0}{x_s-x_0}(x_1 - x_0). \end{cases} \tag{30}$$

The equation (30) has two different solutions, and we only need to choose the one with the shortest distance from (x_0, y_0). As the area is meshed, we need to further quantify the solution in (30). So multiple quantification solutions will be achieved. We can randomly select a solution which satisfies the condition of data collection. A similar way can also be used to revise the action in the data forwarding state, and an optimal location for data forwarding can also be found by replacing r_{SN} with r_{DC} in (30). The training and testing processes of the GSDRL algorithm are presented in Algorithm 1 and Algorithm 2, respectively.

For the training process in Algorithm 1, the agent first clears the experience pool $\mathbf{\Theta}$ and randomly initializes all the parameters including the location of the UAV, the original state $\mathbf{s}$ and the total energy of the UAV $e = E_u$. Given $\mathbf{s}$, the agent selects an action $\mathbf{a}$ to affect the environment. Then, it will further obtain the next state $\mathbf{s}'$ and the corresponding reward R. Next, the tuple $\{\mathbf{s}, \mathbf{a}, \mathbf{s}', R\}$ will be transferred to the pool $\mathbf{\Theta}$. The current state is updated with $\mathbf{s} = \mathbf{s}'$.

With an increase in the number of iterations, the parameters of the current network will gradually converge to local optimal solutions. To avoid this problem, the agent will select an action according to the current network with a probability P_ε, or choose a random action with a probability $1 - P_\varepsilon$ to explore the unknow environment. When $P_\varepsilon \geq P_{\max}$, we set P_ε=$P_{\max}$. Then, the agent will rectify the action with the probability P_τ to help in finding a better location to execute

Algorithm 2 The testing process of the GSDRL algorithm.

for episode $i = 1$ to $D_{\max}$ **do**
  Randomly initialize the UAV's location, the state $\mathbf{s}$ and the UAV energy $e = E_u$.
  while $c_{sn} \leq m_0$ and $P_r > 0$ **do**
    Choose an action $\mathbf{a}$ from the trained target network in Algorithm 1 according to $\mathbf{s}$.
    Execute $\mathbf{a}$ and output the next state $\mathbf{s}'$.
    if $F_{G_k} == 1$ **then**
      Count the number of the loaded SNs c_{sn}.
    end if
    Update the current state wtih $\mathbf{s} = \mathbf{s}'$ and the remaining power with P_r=$P_r - P_i$.
  end while
end for

the mission. With the increase of the number of interactions with the environment, the samples stored in the pool $\mathbf{\Theta}$ increase constantly. After the pool $\mathbf{\Theta}$ accumulates enough samples, the agent starts to take out N_1 random ones from the pool $\mathbf{\Theta}$ and trains the network. The parameter g_j is given by

$$g_j = \begin{cases} R_j, \varsigma = 1, \\ R_j + \lambda Q^{\mathrm{t}}\left(s, \arg\max_{\mathbf{a}'} Q(s', \mathbf{a}, \mathbf{w}), \mathbf{w}^t\right) & , \varsigma = 0, \end{cases} \tag{31}$$

where ς describes the work state of the agent. If $\varsigma = 1$, the agent will stop searching and the episode is over. Using the mini-batch stochastic gradient descent algorithm, the agent builds a loss function $\frac{1}{N_1}\sum_{j=1}^{N_1}\left(g_j - Q(\mathbf{s}_j, \mathbf{a}_j, \mathbf{w})\right)^2$ with N_1 samples to update the parameter $\mathbf{w}$. The current network parameter will be updated after every interaction, while the parameter $\mathbf{w}^t$ of the target network will be updated after ξ interactions. This helps to enhance the stability of network parameters. If $j < J_{\max}$, the process will re-back to the Step 1 and continues to collect or forward the data until the task is accomplished or the remaining energy of the UAV is exhausted.

The testing process is presented in Algorithm 2, the UAV initializes its position randomly and clears the data in the pool $\mathbf{\Theta}$. The energy is $e = E_u$. Then, according to its current state $\mathbf{s}$, the trained network obtained from Algorithm 1 outputs the next action $\mathbf{a}$ to be performed. Then, the UAV will execute the action $\mathbf{a}$ to collect or forward the data. For example, if it is $F_{G_k} = 1$, which denotes the data of the SNs in $\mathbf{G}_k$ all have been collected, the UAV will forward the data to the DC and counts the number of the loaded SNs. Next, it will update the current state $\mathbf{s} = \mathbf{s}'$ and the remaining energy P_r=P_r−P_i, where P_i is the energy consumed to perform the action $\mathbf{a}$. Using this iterative procedure, the UAV can complete the data collection and forwarding of the remaining SNs autonomously.

IV. Simulation Results

In this section, we demonstrate the effectiveness of the proposed scheme through simulation experiments. The experimental setup are windows system 64 bits, CPU Core i7-6700 HQ, pycharm 2020, Torch 1.13.1, numpy 1.24.1. The number of SNs is 8, which is presented in Table II. In the paper, the positions of SNs are fixed, where the Type I nodes are [1, 2], [1.8, 2.5], [1,-2], [0.5, -1], [-2, -3], [-1, 1], [-2, 2], and the Type II nodes are [1.5, 1.8], [-1, -1.5]. The location of the DC is [-2, 2]. The GSDRL algorithm achieves the best performance when compared with the fixed collection data strategy, Deep Q-Network (DQN) [11] and Double DQN (DDQN) [31]. In this paper, the parameters[6] related to the channel model and the training

[6]UAV related parameters are selected based on Table 1 in [29].

TABLE II: Channel and UAV related simulation parameters

Parameters	value
Transmission power of a SN P_{sn}	0.1 W
Transmission power of an UAV P_c	5 W
The total energy of the UAV E_{u}	360000 J
Communication radius of the DC r_{DC}	2000 m
Communication radius of a SN r_{SN}	900 m
N-LOSS model's path loss κ_{NLOSS}	21 dB
LOSS model's path loss κ_{LOSS}	0.1 dB
Maximum endurance time of the UAV T_0	2118 s
Number of the SNs m_0	8
Length of the side of the rectangle L	6 km
Length of each grid d	300 m
Area of an UAV search region	36 km^2
Speed of light v_c	3×10^8
Flying altitude of the UAV H_0	50 m
Environment parameters z_0	4.88
Environment parameters z_1	0.43
Carrier frequency f_c	2.5×10^9 Hz
System bandwidth B_0	1 MHz
Noise power spectrum density	-174 dBm/Hz
Maximum tolerance AoI of the nodes $T_{SN_j}^{th}$ in $\mathbf{\Psi}_h$	600 s

TABLE III: Simulation parameters in the GSDRL algorithm

Parameters	value
Maximum number of episode $I_{\max}$	50000
Maximum number of epoch $J_{\max}$	30
Number of testing times $D_{\max}$	100
Size of the pool $\mathbf{\Theta}$	10000
Weight parameters ϖ_1, ϖ_2 and ϖ_3	10, 5, 10
Number of samples selected N_1	200
Probability increment Δ	0.000001
Steps of parameters to be updated ξ	200
Learning rate β	0.0001
Decay rate γ	0.9
Optimizer	Adam
Activation functions	ReLU/ Leaky-relu/ ReLU
Input layer	$(3m_0 + 4) \times 128$
Hidden layers	$128 \times 256, 256 \times 512$
Output layer	$512 \times 2(L/d + 1)^2$
Maximum probability of selected action from the network $P_{\max}$	0.9

network model are shown in Table II and Table III, respectively. To demonstrate the training process more clearly, the original training data is filtered by the movmean function in Matlab.

A. Channel Analysis

First, we perform an experimental simulations to verify the channel models between the UAV and a SN, and between the UAV and the DC. As shown in Fig. 5, as the communication distance increases, the transmission rate decreases constantly. The transmission power of the UAV is larger than that of the SNs, so the forwarding information rate from the UAV to the DC is larger than that from the SN to the UAV at the same distance. Although the UAV can fly closely to the SN to improve the rate, the farther the UAV flies, the more energy it consumes. Similarly, the UAV has to deal with a similar problem when it forwards the data. Thus, the UAV needs to balance the rate, the AoI and the consumed energy when choosing its location. The GSDRL algorithm can easily solve this problem. Under the constraints of the energy, it can obtain a better location according to the environment. In addition, the simulation results match with the theoretical results derived in (12) and (13).

B. Parameters' Analysis

It can be observed from Fig. 6 that as the number of iterations increases, the reward converges to be stable. From Algorithm 1, the

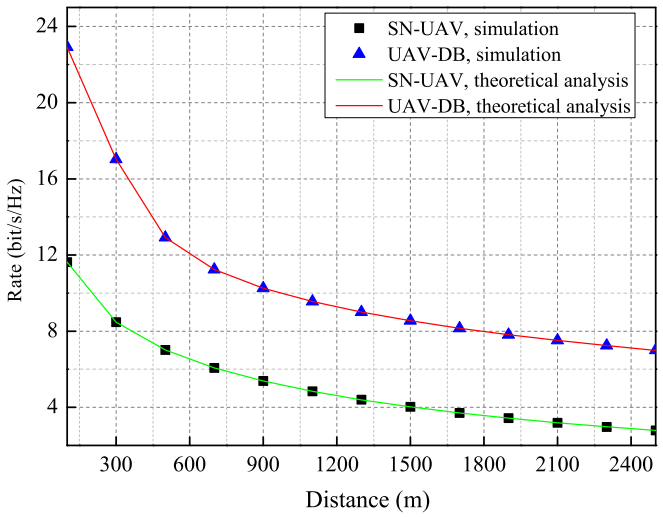

Fig. 5: Rate versus distance for SN-UAV and UAV-DC with V = 20.

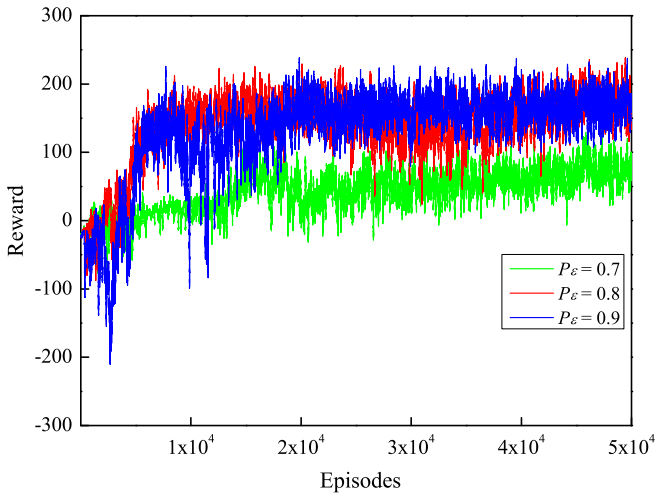

Fig. 6: The training process for different probabilities of action selected in the GSDRL algorithm.

agent selects the actions by the network with exploitation probability P_ε, while randomly choosing an action with the exploration probability $(1 - P_\varepsilon)$. It continues to learn more experience from the environment as the number of iterations increases. It also gradually grasps the ability to take corresponding actions based on real-time environmental information, and improves the rewards. As P_ε increases, the agent tends to choose the actions from the network more frequently, so the accumulated rewards also increase. But this does not mean that the larger P_ε is, the better the reward of the trained network. When P_ε becomes small, the agent will randomly choose an action with a large probability to explore the unknown environment. At this time, as the agent fails to make rational decisions based on environmental information, it may result in frequent failed tasks. We know that the agent stores the selected actions, the current states, the rewards, and the next states into the experience pool $\mathbf{\Theta}$. If the agent suffers too many failures during the training, there will be more inferior samples stored in the pool, which causes the agent to learn a poor strategy.

As shown in Fig. 7, when P_ε increases from 0.8 to 0.9, the average AoI of the Type II nodes does not decrease, but increases. In other words, the system performance does not get better. The parameter P_ε becomes larger, the agent is more inclined to choose the actions from

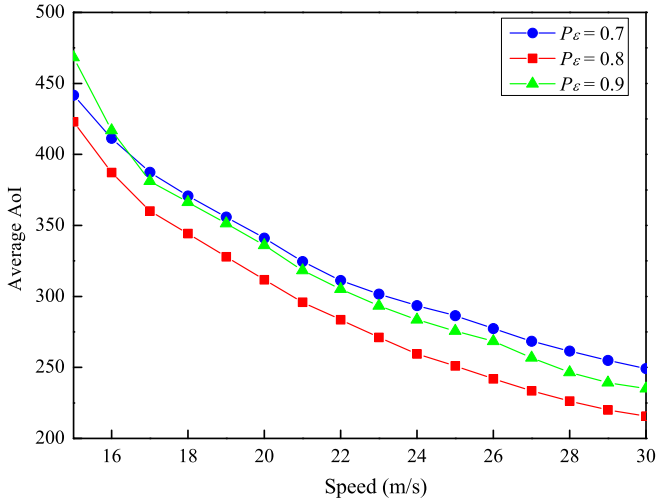

Fig. 7: Average AoI of the Type II nodes versus Speed for different probabilities of action selected in the GSDRL algorithm.

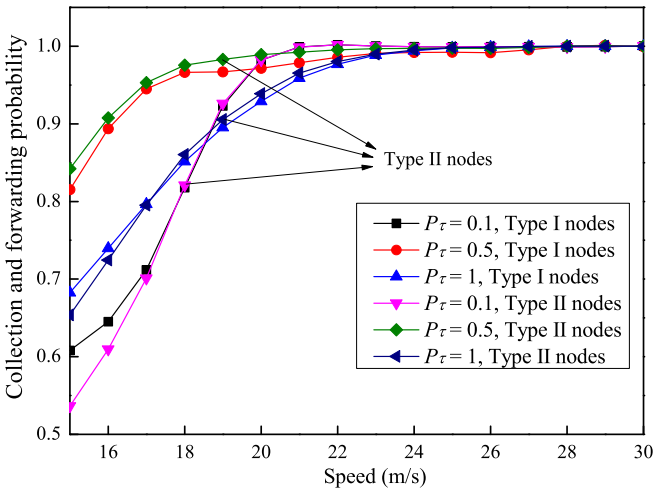

Fig. 8: The collection and forwarding probability versus speed with different P_τ in the GSDRL algorithm.

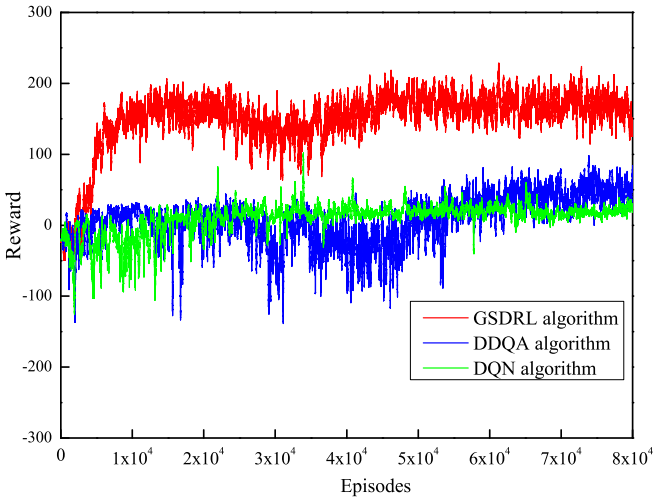

Fig. 9: Training process comparison of the GSDRL, DDQN and DQN algorithms.

the network. Although by increasing P_ε can improve the convergence of network parameters, it also results in a local inferior solution. Based on the above factors, we choose P_ε=0.8 in this paper.

We can see from Section III.C that the agent prefers staying at the signal edge of the target for data collection or forwarding by taking guiding actions strategy. This helps to reduce the flight distance and save more energy. However, there are some overlapping signal coverage zones between multiple SNs when they are located closer together. If we choose to revise the actions given by the network every time, the agent may learn a relatively poor strategy. When the UAV is at the boundary of the area or far away from the SNs, it may fail to collect the data of all nodes due to limited residual energy.

The collection and forwarding probabilities of the nodes with different P_τ are presented in Fig. 8. As the speed increases, the probability curves also increase and approach 1 finally. From (19), we can get that the UAV will prefer to collect the data of Type II nodes to ensure their timeliness. Besides, the 0.5 curves is higher than that of the 0.1 and 1 curves. As P_τ increases from 0.1 to 0.5, the probability increases. However, the probability curves will reduce when P_τ increases from 0.5 to 1. If we adjust the UAV's action rarely, the convergence speed is relatively slow. Conversely, if we revise the UAV's action very frequently, this will not be conducive to help the UAV to find a better solution. For example, the UAV will achieve a better result when it chooses a certain point in the signal overlap area instead of the boundary. Thus, very frequently rectifying actions may miss some opportunities to learn a good strategy. As a result, it will take longer time to complete the data collection and forwarding task. With these various factors taken into consideration, a suitable alternative is P_τ=0.5.

C. Performance Comparison

Next, we demonstrate the superior performance of the proposed algorithm by comparing with the DDQN algorithm, the DQN algorithm, and the fixed collection data strategy. We know that for the deep reinforcement learning algorithms, when more factors in state $\mathbf{s}$ are considered, the more features of the sample are collected, the results of network training will be better. However, there are always too many unpredictable factors in the practical applications. Thus, the UAV's flight speed act as an external dynamic factor to test the robustness of the different algorithms in this paper.

From Section III, we can get that when the conditions of receiving or sending data for the UAV are satisfied, the agent will get a positive reward. Otherwise, it will be punished. In other words, it will achieve a negative reward. As the number of interactions between the agent and the environment increases, the agent continues to accumulate more experience, and it will choose appropriate actions to obtain more rewards based on the current environment state. The three reward curves with different colors are obtained as shown in Fig. 9. As the number of iterations increases, the rewards will initially increase and then become stable. In the classic DRL algorithm, the action with the largest Q value is selected as the next action each time. However, this approach results in an overestimation of the actual reward function due to noise in the Q values, so the system may achieve a suboptimal local solution.

Using g_k in (31), we can find that the choice and the evaluation of the actions are achieved by two different networks in both the DDQN algorithm and the GSDRL algorithm, respectively. As shown in Fig. 9, the GSDRL algorithm has gradually learned the ability to make corresponding decisions when the number of episodes reaches about 3000 times. However, the reward curve of the DDQN algorithm gets stable after about 5500 times. Thus, by rectifying the action method, the GSDRL algorithm can further accelerate the convergence speed and obtain more reward.

Fig.10 presents the collection and forwarding probability of Type I and Type II nodes of the three algorithms. To ensure the timeliness of high-priority nodes, the UAV prefers to collect the Type II nodes

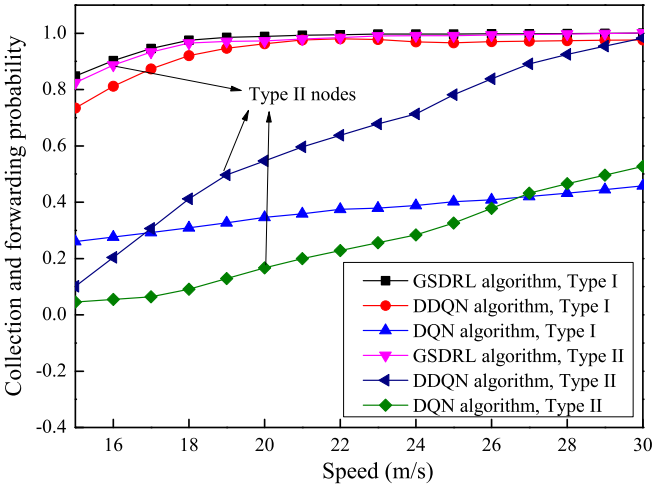

Fig. 10: The collection and forwarding probability versus the speed for the Type I and Type II nodes.

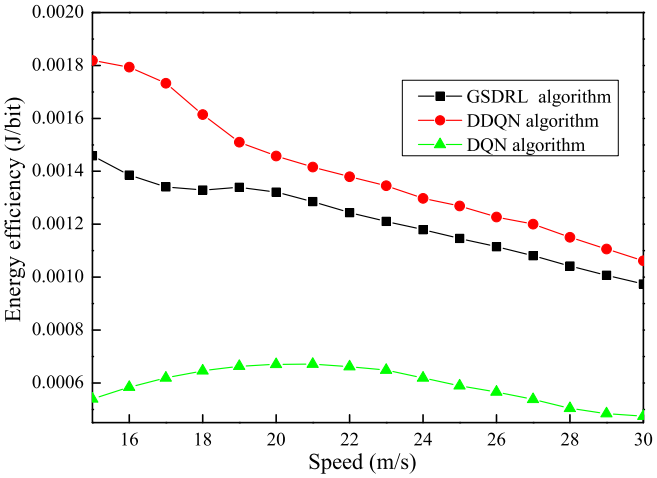

Fig. 11: The energy efficiency of the algorithms versus the speed.

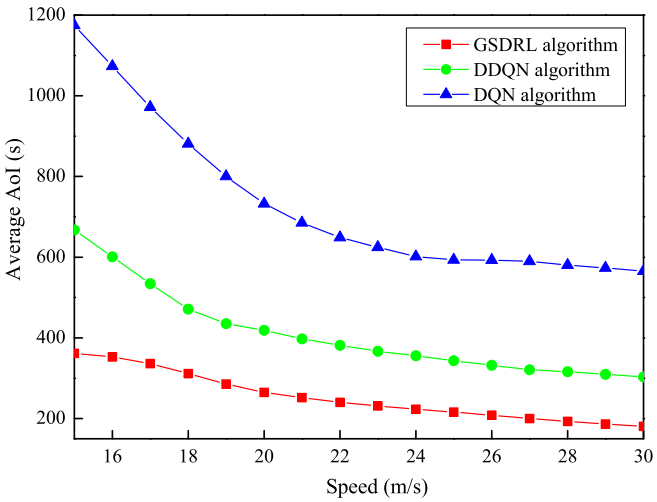

Fig. 12: AoI versus speed for the GSDRL, DDQN and DQN algorithms.

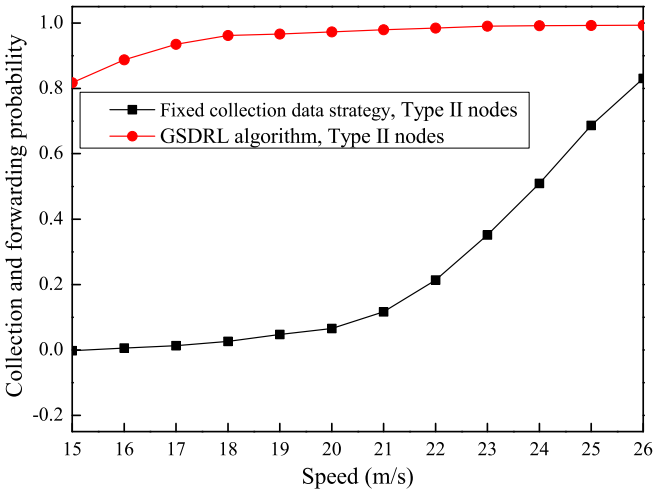

Fig. 13: The collection and forwarding probability versus the speed for the Type II nodes.

and forwards them to the DC as soon as possible. As the fly speed of the UAV increases, it can approach the nodes more quickly, and completes the data collection and forwarding task in less time, so the collection and forwarding probability also increases. The GSDRL algorithm adopts an action guidance strategy to accelerates the convergence speed of the network. When compared with the other algorithms, regardless of Type I and Type II nodes, the GSDRL algorithm can complete data collection and forwarding task with a probability close to 1 when the speed is greater than 20 m/s.

As can be seen from Fig.11, the energy efficiency of the algorithms decreases as the speed increases. Among them, the DQN algorithm has the lowest energy efficiency. Although the DDQN algorithm achieves more efficient than the GSDRL algorithm. However, we can find it from Fig.10 that the DDQN algorithm fails to ensure the timeliness of the Type II nodes. To reduce the AoI of the Type II nodes, the GSDRL algorithm flies to a suitable position to forward the data to the DC as soon as possible. Although this results in a larger flight distance and more energy consumption, the GSDRL algorithm can simultaneously take into account the timeliness of the two different types nodes.

As shown in Fig. 12 that with the increase of UAV flight speed, the UAV can quickly get closer to the near vicinity of the SNs and the DC to reduce the average AoI. When compared with the existing baseline algorithms, the proposed algorithm can find a better location for the UAV to perform the task. It also has achieved the best strategy to complete the task for data collection and forwarding according to the real-time environment. When compared with the DQN and DDQN algorithms, if the speed is 20 m/s, the AoI of the GSDRL algorithm can be reduced by about 63.85% and 36.76%, respectively.

Performance comparisons for two different data collection strategies are presented in Fig.13 and Fig.14. For a fixed data collection strategy, the UAV first collects the data of all nodes, then sends the data to the DC together. Then, the GSDRL algorithm adopts a dynamic data collection strategy. As can be seen from Fig.13, as the UAV's speed increases, the collection and forwarding probability of Type II nodes also increases. This is because the UAV can approach the Type II nodes more quickly, and completes the data collection and forwarding task in less time. By adopting the dynamic collection strategy, the GSDRL algorithm can optimize fly trajectory according to the specific environment, so the probability of collecting and forwarding for the Type II nodes of the GSDRL algorithm is higher than that of the fixed data collection strategy. Besides, the GSDRL algorithm tries to collect and forward the data of the Type II nodes as soon as possible, finds a short trajectory to collect the Type I nodes. Thus, we can also find that the average AoI of the GSDRL algorithm is lower than that of the fixed collection strategy in Fig.14.

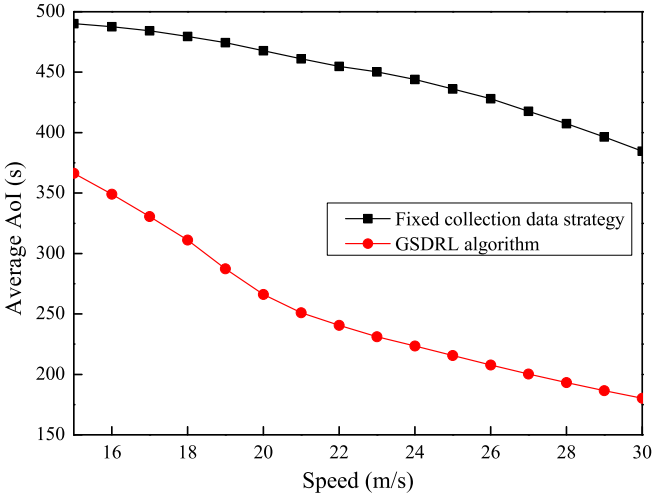

Fig. 14: The average AoI versus the speed for the fixed collection data strategy and the GSDRL algorithm.

V. Conclusion

For an UAV to perform the task of data collection and forwarding, we model it as a sequential decision-making problem with the goal of minimizing the average AoI of the system. A guided search deep reinforcement learning algorithm is presented to solve the problem in this paper. The proposed scheme optimizes data collection and forwarding strategy to meet the timeliness requirements of different users. A fixed mode is taken by the traditional data collection method generally, where the UAV given an initial location executes the task according to a preplanned flight trajectory. Different from the traditional data collection methods, the proposed scheme is more flexible. The agent selects an optimal strategy to cope with the diverse conditions. For the larger search space, we design an action guided search method to speed up the convergence of the network parameters and find a better flight trajectory. Simulation experiments show that the GSDRL algorithm has strong adaptability to the adverse environment. It can choose reasonable actions based on the real-time environment. When compared with the fixed data collection strategy, the GSDRL algorithm finds better locations for the UAV to collect and forward the data. It can ensure the timeliness of the Type II nodes, also reduces the average AoI of nodes. Except for a single UAV data collection, multiple drones assisted each other is a new issue to be discussed to perform data collection task.

Appendix

We assume that $\Psi = P_T^{SN} \Big/ P_{\sigma^2}$ and $y = B_0 \times \log_2(1+\Psi x) \geq 0$, we can get

$$dy = \frac{B_0}{\ln 2} \times \frac{\Psi}{1+\Psi x} dx, \tag{32}$$

where B_0 is the signal bandwidth. According to (8) and (9), we can obtain that the CDF of y is

$$\begin{aligned} F_y(y) &= P(Y \leq y) \\ &= P(B_0 \log_2(1+\Psi X) \leq y) \\ &= P\left(X \leq \frac{1}{\Psi}\left(2^{y/B_0}-1\right)\right) \\ &= F_x\left(X \leq \frac{1}{\Psi}\left(2^{y/B_0}-1\right)\right). \end{aligned} \tag{33}$$

Combining (9) and (33), we can get the PDF of y is given by

$$\begin{aligned} f_y(y) &= f_x\left(\frac{1}{\Psi}\left(2^{y/B_0}-1\right)\right) \frac{2^{y/B_0}}{\Psi} \ln 2 \\ &= \frac{1}{\overline{P_r}} \exp\left(-\frac{1}{\overline{P_r}}\left(\frac{1}{\Psi}\left(2^{y/B_0}-1\right)\right)\right) \frac{2^{y/B_0}}{\Psi} \ln 2. \end{aligned} \tag{34}$$

According to (32), so the expected value of y is

$$\begin{aligned} E(y) &= \int y f(y)\, dy \\ &= \int \left(\begin{array}{l} B_0 \times \log_2(1+\Psi x) \frac{1}{\overline{P_r}} \\ \times \exp\left(-\frac{1}{\overline{P_r}}\left(\frac{1}{\Psi}\left(2^{y/B_0}-1\right)\right)\right) \\ \times \frac{2^{y/B_0}}{\Psi} \ln 2 \times \frac{B_0}{\ln 2} \times \frac{\Psi}{1+\Psi x} \end{array} \right) dx \\ &= \frac{B_0}{\overline{P_r}} \int \ln 2 (1+\Psi x) \exp\left(-\frac{1}{\overline{P_r}} x\right) dx \\ &= \frac{B_0}{\ln 2 \times \overline{P_r}} \int \ln(1+\Psi x) \times \exp\left(-\frac{x}{\overline{P_r}}\right) dx \\ &= -\frac{B_0}{\ln 2} e^{\frac{1}{\Psi \overline{P_r}}} Ei\left(-\frac{1}{\Psi \overline{P_r}}\right), \end{aligned} \tag{35}$$

where $Ei(\bullet)$ denotes the one-argument exponential integral function. Combining (10) and (35), we can get

$$\begin{aligned} E(R_C) &= -\frac{B_0}{\ln 2} e^{\frac{1}{\Psi \overline{P_r}}} Ei\left(-\frac{1}{\Psi \overline{P_r}}\right) \\ &= -\frac{B_0}{\ln 2} e^{\frac{P_{\sigma^2}}{P_T^{SN} P_R^U}} Ei\left(-\frac{P_{\sigma^2}}{P_T^{SN} P_R^U}\right). \end{aligned} \tag{36}$$

References

[1] M. R. Palattella *et al.*, "Internet of things in the 5G era: Enablers, architecture, and business models," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 3, pp. 510–527, Mar. 2016.
[2] Y. Zeng, R. Zhang, and T. J. Lim, "Wireless communications with unmanned aerial vehicles: opportunities and challenges," *IEEE Commun. Mag.*, vol. 54, no. 5, pp. 36–42, May 2016.
[3] P. Agrawal and N. Patwari, "Correlated link shadow fading in multi-hop wireless networks," *IEEE Trans. Wireless Commun.*, vol. 8, no. 8, pp. 4024–4036, Aug. 2009.
[4] A. Fotouhi *et al.*, "Survey on UAV cellular communications: Practical aspects, standardization advancements, regulation, and security challenges," *IEEE Commun. Surveys Tutorials*, vol. 21, no. 4, pp. 3417–3442, Fourthquarter 2019.
[5] F. Mohammed *et al.*, "UAVs for smart cities: Opportunities and challenges," in *Proc. 2014 International Conference on Unmanned Aircraft Systems (ICUAS)*, Orlando, FL, USA, 2014, pp. 267–273.
[6] S. Hayat, E. Yanmaz, and R. Muzaffar, "Survey on unmanned aerial vehicle networks for civil applications: A communications viewpoint," *IEEE Commun. Surveys Tutorials*, vol. 18, no. 4, pp. 2624–2661, Fourthquarter 2016.
[7] B. Galkin, J. Kibiłda, and L. A. DaSilva, "A stochastic model for UAV networks positioned above demand hotspots in urban environments," *IEEE Trans. Veh. Technol.*, vol. 68, no. 7, pp. 6985–6996, Jul. 2019.
[8] N. H. Motlagh, M. Bagaa, and T. Taleb, "UAV-based IoT platform: A crowd surveillance use case," *IEEE Commun. Mag.*, vol. 55, no. 2, pp. 128–134, Feb. 2017.
[9] M. Emara, H. Elsawy, and G. Bauch, "A spatiotemporal model for peak AoI in uplink iot networks: Time versus event-triggered traffic," *IEEE Internet Things J.*, vol. 7, no. 8, pp. 6762–6777, Aug. 2020.
[10] S. Kaul, R. Yates, and M. Gruteser, "Real-time status: How often should one update?" in *Proc. 2012 Proceedings IEEE INFOCOM*, Orlando, FL, USA, 2012, pp. 2731–2735.
[11] M. Volodymyr *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015.
[12] Y. Zeng, R. Zhang, and T. J. Lim, "Wireless communications with unmanned aerial vehicles: opportunities and challenges," *IEEE Commun. Mag.*, vol. 54, no. 5, pp. 36–42, May 2016.

[13] Q. Li *et al.*, "Joint activity and blind information detection for UAV-assisted massive IoT access," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 5, pp. 1489 –1508, May 2022.
[14] Y. Liu *et al.*, "UAV communications based on Non-Orthogonal multiple access," *IEEE Wireless Commun.*, vol. 26, no. 1, pp. 52–57, Feb. 2019.
[15] Y. Wang *et al.*, "Trajectory design for UAV-based internet of things data collection: A deep reinforcement learning approach," *IEEE Internet Things J.*, vol. 9, no. 5, pp. 3899 – 3912, Mar. 2022.
[16] S. Zhang *et al.*, "Joint trajectory and power optimization for UAV relay networks," *IEEE Commun. Lett.*, vol. 22, no. 1, pp. 161–164, Jan. 2018.
[17] C. Zhou *et al.*, "Deep RL-based trajectory planning for AoI minimization in UAV-assisted IoT," in *Proc. 2019 11th International Conference on Wireless Communications and Signal Processing (WCSP)*, Xi'an, China, 2019, pp. 1–6.
[18] M. A. Abd-Elmagid and H. S. Dhillon, "Average peak Age-of-information minimization in uav-assisted IoT networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 2, pp. 2003–2008, Feb. 2019.
[19] T. Peng *et al.*, "Deep reinforcement learning for efficient data collection in UAV-Aided internet of things," in *Proc. 2020 IEEE International Conference on Communications Workshops (ICC Workshops)*, Dublin, Ireland, 2020, pp. 1–6.
[20] M. Sun *et al.*, "AoI-energy-aware UAV-assisted data collection for IoT networks: A deep reinforcement learning method," *IEEE Internet Things J.*, vol. 8, no. 24, pp. 17 275–17 289, Dec. 2021.
[21] C. Zhan, Y. Zeng, and R. Zhang, "Energy-efficient data collection in UAV enabled wireless sensor network," *IEEE Trans. Wireless Commun. lett.*, vol. 7, no. 3, pp. 328–331, Nov. 2018.
[22] G. Chatzimilioudis *et al.*, "Crowdsourcing with smartphones," *IEEE Internet Comput.*, vol. 16, no. 5, pp. 36–44, Oct. 2012.
[23] L. Wang *et al.*, "Sparse mobile crowdsensing: challenges and opportunities," *IEEE Commun. Mag.*, vol. 54, no. 7, pp. 161–167, Jul. 2016.
[24] M. Alwateer, S. W. Loke, and N. Fernando, "Enabling drone services: Drone crowdsourcing and drone scripting," *IEEE Access*, vol. 7, pp. 110 035–110 049, Aug. 2019.
[25] K. Arulkumaran *et al.*, "Deep reinforcement learning: A brief survey," *IEEE Signal Process. Mag.*, vol. 34, no. 6, pp. 26–38, Nov. 2017.
[26] M. Costa, M. Codreanu, and A. Ephremides, "On the age of information in status update systems with packet management," *IEEE Trans. Inf. Theory*, vol. 62, no. 4, pp. 1897–1910, Apr. 2016.
[27] D. Athukoralage *et al.*, "Regret based learning for UAV assisted LTE-U/WiFi public safety networks," in *Proc. 2016 IEEE Global Communications Conference (GLOBECOM)*, Washington, DC, USA, 2016, pp. 1–7.
[28] A. Al-Hourani, S. Kandeepan, and A. Jamalipour, "Modeling air-to-ground path loss for low altitude platforms in urban environments," in *Proc. 2014 IEEE Global Communications Conference*, Austin, TX, USA, 2014, pp. 2898–2904.
[29] Y. Zeng, J. Xu, and R. Zhang, "Energy minimization for wireless communication with rotary-wing UAV," *IEEE Trans. Wireless Commun.*, vol. 18, no. 4, pp. 2329–2345, Apr. 2019.
[30] J. Konečnỳ *et al.*, "Mini-batch semi-stochastic gradient descent in the proximal setting," *IEEE J. Sel. Topics Signal Process.*, vol. 10, no. 2, pp. 242–255, Mar. 2016.
[31] H. van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in *Proc. Association for the Advancement of Artificial Intelligence*, 2016, pp. 2094–2100.