Deep Reinforcement Learning-Based Multi-node Collaborative Task Offloading Optimization in 6G Space-Air-Ground Integrated Networks

Wang Li, Xin Chen, Libo Jiao(✉), Wangzhong Ning, and Wenwu Zhu

Beijing Information Science and Technology University, Beijing, China
{wangli,chenxin,jiaolibo,ningwangzhong,zhuwenwu}@bistu.edu.cn

Abstract. With the explosion of communication data volume, 6G communication has gradually entered the vision of academia and industry. In addition, in order to support the execution of computationally intensive applications, the study of 6G space-air-ground integration network (SAGIN) is becoming more and more extensive, in which satellites, drones, and base stations can provide arithmetic support for mobile users (MUs) through multi-access edge computing (MEC). However, the variety of offloading modes, the mobility of nodes and the stochastic state of wireless networks make that selecting the optimal base station and allocating the appropriate computational resources become more challenging. In this paper, we first describe the offloading process and establish the SAGIN model. Then the Deep Reinforcement Learning-based Task Offloading Optimization Algorithm (DTOOA) is proposed to jointly optimize the problem of minimizing latency and energy consumption. The numerical results show that the DTOOA scheme significantly outperforms benchmark schemes and markedly improves the quality of experience (QoE) of MUs.

Keywords: 6G · Space-Air-Ground Networks · Multi-access Edge Computing · Task Offloading · Deep reinforcement learning

1 Introduction

According to the June 2023 Mobility Report from Ericsson, the number of global 5G subscribers will reach 1.5 billion and average global monthly usage per smartphone will exceed 20 GB by the end of 2023. In the era of unprecedented information explosion, huge data traffic puts higher demands on communication technology. 5G, which has already entered the commercial deployment phase, still has various limitations. In order to meet services such as global coverage, ultra-high-rate transmission, ultra-low latency and the Internet of Everything, industry,

Supported by the National Natural Science Foundation of China under Grant 62202059.

A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 593, pp. 313–332, 2024.
https://doi.org/10.1007/978-3-031-63989-0_16

academia and standards organizations have begun exploring and researching 6G [10]. 6G space-air-ground integration network (SAGIN) is considered a promising network architecture to address the limitations of 5G.

Mobile users (MUs) usually have limited computing resources and cannot handle computationally intensive tasks locally. The multi-access edge computing (MEC) paradigm was born to better meet new applications such as autonomous driving, ultra-high-definition video and telemedicine [6]. Compared to traditional cloud computing [4], MEC brings computing resources closer to the MUs and reduces processing latency and energy consumption of the MUs by offloading the tasks to the edge nodes. However, the terrestrial radio access network suffers from limited coverage and rigid network structure, which cannot timely handle the task offloading requirements in remote areas. MEC-enabled 6G SAGIN has a three-layer network that includes ground-based, air-based and space-based networks, providing MUs with all-time, all-domain services by linking multiple nodes such as ground base stations, drones and satellites [7].

MEC-enabled 6G SAGIN is a dynamic, flexible multi-node network architecture. Therefore, optimizing the task offloading problem in 6G SAGIN networks has many challenges. Firstly, due to the limited computational resources of the edge nodes and temporal variability of the wireless network environment, the MUs need to select one of the multiple offloading methods and identify a particular edge node for task offloading. Yang et al. [13] utilized a multi-task learning approach in MEC to select appropriate edge nodes for computational offloading. Shen et al. [8] proposed a collaborative task offloading framework based on slicing for space-air-ground integrated vehicular networks and optimized the allocation of slicing resources through a deep reinforcement learning (DRL) algorithm. Secondly, when multiple MUs select the same edge node, the allocation of computational resources to each MUs needs to be optimized. Waqar et al. [12] proposed a MEC-enabled integrated aerial-terrestrial vehicle network and optimized the computational resource allocation to reduce the computational and communication overheads. Cao et al. [1] studied the problem of minimum energy consumption in low Earth orbit (LEO) satellite edge-assisted multilayer MEC systems. Tang et al. [9] proposed a reinforcement learning based traffic offloading scheme in SAGIN. Finally, edge nodes in 6G SAGIN are not stationary. When the mobility of MUs and mobile base stations (MBSs) makes the network topology change, new offloading methods need to be determined and computational resources need to be reallocated, Liu et al. [5] consider the effect of frequent movement of MUs on the selection of optimal edge nodes in mobile edge computing.

Many studies have discussed task offloading strategies in SAGIN, however, the mobility of MUs and MBSs, the problem for offloading method selection in the overlapping coverage area of edge nodes, and the complexity of multiple communication links in different edge nodes have not been fully considered. In this paper, we propose a task offloading framework in 6G SAGIN and solve the optimization problem involved using a DRL algorithm. Specifically, our contribution is as follows:

– A MEC-enabled 6G SAGIN task offloading framework is proposed. Considering the mobility of MUs and MBSs, the diversity of communication methods, and the heterogeneity of edge nodes, we described the task offloading process and model the system in 6G SAGIN.
– In order to improve the quality of experience (QoE) of MUs, we propose a problem of minimizing the average cost of MUs based on the system model, and a DRL-based task offloading optimization algorithm (DTOOA) incorporating Markov processes is proposed to jointly optimize the average cost of MUs weighted by latency and energy consumption.
– A number of simulation experiments were conducted to evaluate the performance of the DTOOA scheme. In many different simulation environments, the numerical results show that the DTOOA scheme is more effective than other offloading optimization schemes for reducing the average cost of MUs.

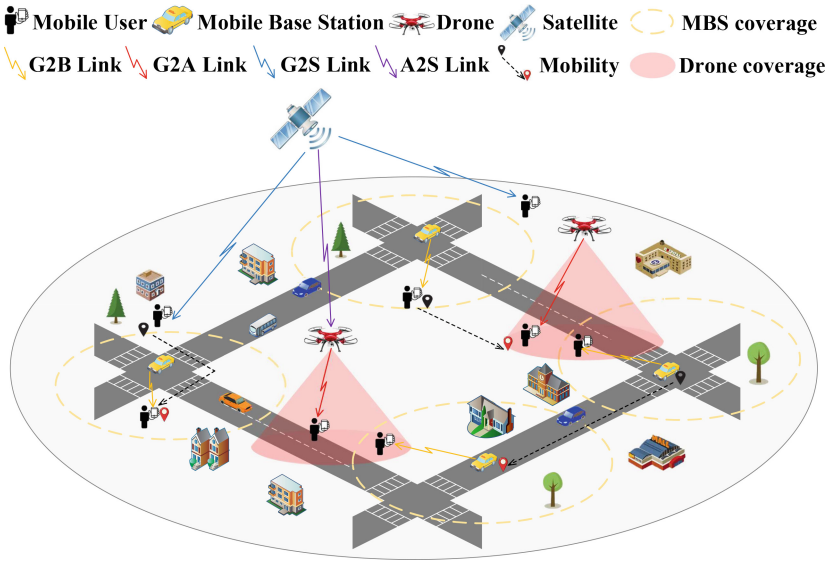

Fig. 1. MEC-enabled 6G Space-Air-Ground Integrated Networks.

2 System Model

Figure 1 shows the MEC-enabled 6G SAGIN scenario, which consists of LEO satellite, drones, and MBSs. In this paper, we optimize tasks offloading process within the satellite coverage. In addition to performing tasks locally, MUs can offload tasks to MBSs, drones or satellite via ground-to-MBS (G2B), ground-to-air (G2A) and ground-to-space (G2S) communication links. In particular, when

the path loss between the MU and the satellite is excessive due to the factors such as signal blocking and occlusion, the MU can first upload the task to the drone via G2A link, and then the drone offloads the task to the satellite via air-to-space (A2S) link.

In this section, we first present an overview of SAGIN, and then provide the detailed definitions of offloading design, mobility model, communication model, and computational model.

2.1 Overview

For Network Architecture. We deployed 6G SAGIN in remote areas without terrestrial fixed cellular base station coverage, thus utilize satellite, drones, and MBSs deployed in vehicles to support MUs for computationally intensive task offloading. In this scenario, we consider a LEO satellite, indexed as $\mathcal{O}$, multiple pre-deployed drones, indexed as $\mathcal{D} = \{1, 2, ..., d, ..., D\}$, multiple MBSs, indexed as $\mathcal{M} = \{1, 2, ..., m, ..., M\}$, and multiple MUs that may need to perform task offloading, indexed by $\mathcal{N} = \{1, 2, ..., n, ..., N\}$. LEO satellite provide seamless coverage of the entire network area, whereas both drones and MBSs have limited coverage. In addition, MBSs and MUs are moving within the network area, and task offloading for MUs can only be provided when the MUs enter the coverage area of the drones or MBSs.

For Time Slot and Task. We define the time slot model as $\mathbb{T} = \{1, 2, ..., t, ..., T\}$, in which the duration of each time slot is ψ, and assume that the network topology, node locations, and channel conditions remain constant over a time slot t. Each MU generates an atomic and non-redivisible computationally intensive task in each time slot t, and each computational task is described by a 3-tuple $\mathcal{K}_n(t) = (k_n^{\mathcal{I}}(t), k_n^{\mathcal{C}}(t), k_n^{\mathcal{L}}(t))$, where $k_n^{\mathcal{I}}(t)$[bits], $k_n^{\mathcal{C}}(t)$[cycles], and $k_n^{\mathcal{L}}(t)$[seconds] represent the input data size of the task, the amount of computation required for the task, and the maximum tolerable latency of the task, respectively.

2.2 Offloading Design

MUs are able to handle small scale tasks locally. However, in order to support compute-intensive applications such as AR/VR, autonomous driving, and ultra-high-definition video, MUs usually need to select edge nodes for task offloading. To minimize the long term average latency and energy consumption of MUs, we propose a three step offloading scheme including offloading request, offloading strategy, and task execution.

Offloading Request. At the beginning of each time slot, each MU sends an offload request to the edge node currently providing the service. If the current MUs is performing the task locally, the MUs will send an offloading request directly to the LEO satellite. The intelligent agent deployed in the satellite aggregate information such as offload requests and edge node computational resources and use the information to make intelligent decisions.

Offloading Strategy. The satellite is the center node in the 6G SAGIN topology, in which the deployed intelligent agent determine the offloading strategy for each time slot. The offloading strategy include determining how each MU task is to be performed and the computational resources allocated to the MUs by each edge node. There are five execution modes for each MU task, including local execution, offloading to a MBS, offloading to a drone, offloading to the LEO satellite, and offloading to the LEO satellite through a drone retransmission. Normally, the MU can offload the task to the satellite directly, however, when the ground-satellite communication signal fades excessively, drones with altitude advantage can be selected as a relay nodes to retransmit the MU task to the satellite for offloading. Of course, when the MU selects the edge node for offloading or retransmission, it needs to satisfy the precondition that the MU is within the coverage area of the edge node. In addition, the optimal choice of nodes needs to be considered when the MU is within the overlapping coverage of multiple edge nodes.

We define $\mathbf{U}_n(t) = \{u_n^{\mathcal{N}}(t), mu_n^{\mathcal{M}}(t), du_n^{\mathcal{D}}(t), u_n^{\mathcal{O}}(t), d'u_n^{\mathcal{R}}(t)\}, m \in \mathcal{M}, d \in \mathcal{D}, d' \in \mathcal{D}, \forall n \in \mathcal{N}$ as the 5-dimensional execution mode vector of the MU n task at time slot t, where $u_n^{\mathcal{N}}(t), u_n^{\mathcal{M}}(t), u_n^{\mathcal{D}}(t), u_n^{\mathcal{O}}(t), u_n^{\mathcal{R}}(t) \in \{0, 1\}$ denote whether the five execution modes are selected, in sequence corresponding to local execution, offloading to a MBS, offloading to a drone, offloading to satellite and offloading to satellite via a drone relay, respectively. The coefficients m and d indicate which specific MBS and drone the task is offloaded to, respectively. The coefficient d' indicates which specific drone is selected for task retransmission. MU n can only choose one task execution method, therefore the execution method needs to satisfy the constraint

$$u_n^{\mathcal{N}}(t) + u_n^{\mathcal{M}}(t) + u_n^{\mathcal{D}}(t) + u_n^{\mathcal{O}}(t) + u_n^{\mathcal{R}}(t) = 1, \forall n \in \mathcal{N}. \tag{1}$$

After the execution of the MU tasks has been determined, the intelligent agent need to determine the appropriate computational resources that edge nodes allocate to MUs. Let $\mathbf{C}_n(t) = \{c_n^{\mathcal{N}}(t), c_n^{\mathcal{M}}(t), c_n^{\mathcal{D}}(t), c_n^{\mathcal{O}}(t), c_n^{\mathcal{R}}(t)\}, \forall n \in \mathcal{N}$ denotes the 5-dimensional computational resource allocation vector, which represents the computational resources allocated to the MU n through local execution, offloading to a MBS, offloading to a drone, offloading to a satellite and offloading to satellite via a drone relay, respectively. To summarize, we define the offloading strategy of MU n at time slot t as

$$a_n(t) = (\mathbf{U}_n(t), \mathbf{C}_n(t)), \forall n \in \mathcal{N}. \tag{2}$$

Based on the definition above, the offloading strategy for all MUs at time slot t in 6G SAGIN scenario is given as

$$a(t) = \{\mathbf{U}(t), \mathbf{C}(t)\} = \{a_1(t), a_2(t), ..., a_n(t), ..., a_N(t)\}. \tag{3}$$

Task Execution. Based on the optimized offloading strategy made by the intelligent agent, the MUs execute the tasks according to the corresponding

execution method and the allocated computational resources. The task execution phase consists of three processes, including task upload, task computation and result return. When the task is executed locally, the task upload and result return processes are omitted.

2.3 Mobility Model

In the SAGIN scenario, the satellite provides global offloading strategy guidance and seamless coverage to support task offloading. Drones with smaller coverage areas are pre-deployed in areas of high computational demand to address the problem of localized computational intensity. MBSs with larger coverage range move randomly within the network range to avoid over-concentration of computing resources and extend the overall service range. The coordinates of MU n, MBS m, drone d, and satellite at time slot t are defined using 3-dimensional Cartesian coordinates $(x_n^{\mathcal{N}}(t), y_n^{\mathcal{N}}(t), 0)$, $(x_m^{\mathcal{M}}(t), y_m^{\mathcal{M}}(t), 0)$, $(x_d^{\mathcal{D}}(t), y_d^{\mathcal{D}}(t), z_d^{\mathcal{D}}(t))$ and $(x^{\mathcal{O}}(t), y^{\mathcal{O}}(t), z^{\mathcal{O}}(t))$ with units of meters, respectively. For simplicity, we assume that LEO satellites are semi-static over in a fixed time frame, and from a practical implementation perspective, it is necessary to federate multiple satellites to ensure the network range can be covered continuously and seamlessly. The drones fixed at pre-deployed locations, and the MUs and MBSs move randomly within the network range according to random vectors $\mathbf{v}_n^{\mathcal{N}}(t) = (\Delta x_n^{\mathcal{N}}(t), \Delta y_n^{\mathcal{N}}(t), 0)$ and $\mathbf{v}_m^{\mathcal{M}}(t) = (\Delta x_m^{\mathcal{M}}(t), \Delta y_m^{\mathcal{M}}(t), 0)$ following Gaussian distribution. Therefore, the coordinates of MU n and MBS m at time slot $t+1$ can be expressed as

$$\begin{aligned}
(x_n^{\mathcal{N}}(t+1), y_n^{\mathcal{N}}(t+1), 0) &= (x_n^{\mathcal{N}}(t), y_n^{\mathcal{N}}(t), 0) + \mathbf{v}_n^{\mathcal{N}}(t) \cdot \min(1, \frac{V_{max}^{\mathcal{N}}}{|\mathbf{v}_n^{\mathcal{N}}(t)|}), \\
(x_m^{\mathcal{M}}(t+1), y_m^{\mathcal{M}}(t+1), 0) &= (x_m^{\mathcal{M}}(t), y_m^{\mathcal{M}}(t), 0) + \mathbf{v}_m^{\mathcal{M}}(t) \cdot \min(1, \frac{V_{max}^{\mathcal{M}}}{|\mathbf{v}_m^{\mathcal{M}}(t)|}),
\end{aligned} \tag{4}$$

where $V_{max}^{\mathcal{N}}$ and $V_{max}^{\mathcal{M}}$ represent the maximum distance that MUs and MBSs can moving during a single time slot, respectively.

Based on the defined coordinate system, we can obtain the distances between MU n and MBS m, drone d, satellite at time slot t as $i_{n,m}^{\mathcal{M}}(t)$, $i_{n,d}^{\mathcal{D}}(t)$, $i_n^{\mathcal{O}}(t)$ respectively. The distance between drone d and satellite at time slot t is $i_m^{\mathcal{R}}(t)$. We define the topological relationship at time slot t between MU n and MBS m as $\alpha_{n,m}^{\mathcal{M}}(t)$ and denote as

$$\alpha_{n,m}^{\mathcal{M}}(t) = \begin{cases} 1, & i_{n,m}^{\mathcal{M}}(t) \le l_m^{\mathcal{M}} \\ 0, & else \end{cases}, \forall n \in \mathcal{N}, \forall m \in \mathcal{M}, \tag{5}$$

where $l_m^{\mathcal{M}}$ signifies the coverage radius of the MBS m. In a similar way, establish the topological association between MU n and drone d at time slot t as $\alpha_{n,d}^{\mathcal{D}}(t) \in \{0, 1\}$, then we have

$$\alpha_{n,d}^{\mathcal{D}}(t) = \begin{cases} 1, & i_{n,d}^{\mathcal{D}}(t) \le l_d^{\mathcal{D}} \\ 0, & else \end{cases}, \forall n \in \mathcal{N}, \forall d \in \mathcal{D}, \tag{6}$$

where $l_d^{\mathcal{D}}$ denotes the coverage radius of the drone d. Because of the mobility of MUs and MBSs, the set of edge nodes that each MU can request for task offloading may change at each time slot. We define the MBSs and drones vectors that can be requested by the MU n at time slot t as $\mathbf{M}_n(t)$ and $\mathbf{F}_n(t)$, respectively, then we have

$$\begin{aligned} \mathbf{M}_n(t) &= \{\alpha_{n,1}^{\mathcal{M}}(t), ..., \alpha_{n,M}^{\mathcal{M}}(t)\}, \forall n \in \mathcal{N}, \\ \mathbf{D}_n(t) &= \{\alpha_{n,1}^{\mathcal{D}}(t), ..., \alpha_{n,D}^{\mathcal{D}}(t)\}, \forall n \in \mathcal{N}. \end{aligned} \tag{7}$$

Based on the definition above, the optional task execution mode vector of MU n at time slot t can be defined as

$$\mathbf{O}_n(t) = \{o_n^{\mathcal{N}}(t), \mathbf{M}_n(t), \mathbf{D}_n(t), o_n^{\mathcal{O}}(t), \mathbf{D}_n(t)\}, \forall n \in \mathcal{N}, \tag{8}$$

where $o_n^{\mathcal{N}}, o_n^{\mathcal{O}} = 1$ means that the MU can always choose the two task execution modes of local execution or offloading to satellite. When MU n selects a task execution mode at time slot t, it must be satisfied that the selected execution mode is within the optional task execution mode vector. For a given execution mode vector $\mathbf{U}_n(t) = \{u_n^{\mathcal{N}}(t), mu_n^{\mathcal{M}}(t), du_n^{\mathcal{D}}(t), u_n^{\mathcal{O}}(t), d'u_n^{\mathcal{R}}(t)\}, m \in \mathcal{M}, d \in \mathcal{D}, d' \in \mathcal{D}, \forall n \in \mathcal{N}$, we define the execution mode index as $I_n^{\mathcal{U}}(t)$ and expressed as

$$\begin{aligned} I_n^{\mathcal{U}}(t) = u_n^{\mathcal{N}}(t) + (1+m)u_n^{\mathcal{M}}(t) + (1+M+d)u_n^{\mathcal{D}}(t) \\ + (2+M+D)u_n^{\mathcal{O}}(t) + (2+M+D+d')u_n^{\mathcal{R}}(t), \end{aligned} \tag{9}$$

where M, D denote the number of MBSs and drones, respectively. Considering the optional task execution mode vector is $(2+M+2D)$ dimensional and based on the execution mode index $I_n^{\mathcal{U}}(t)$, we can get the execution mode selection constraint $\mathbf{O}_n(t)[I_n^{\mathcal{U}}(t) - 1] = 1, \forall n \in \mathcal{N}$, and at this point, the index of the optional task execution mode vector starts from 0.

2.4 Communication Model

When MUs execute tasks by offloading to edge nodes, the task input data $k_n^{\mathcal{I}}(t)$ needs to be uploaded through the uplink. In SAGIN, we consider four types of communication links including G2B, G2A, G2S and A2S. Each MU is equipped with a single antenna and each edge node such as MBS, drone and satellite are equipped with multiple antennas. To better suppress the intra-cell interference, MUs use orthogonal frequency division multiple access (OFDMA) technology to access the edge nodes in the uplink.

G2B Communication Links. According to the OFDMA technology, the operating band $B^{\mathcal{M}}$ of each MBS is evenly partitioned into N sub-bands, and the size of each sub-bands is $W_n^{\mathcal{M}} = B^{\mathcal{M}}/N$ [MHz]. The non-overlapping sub-bands will be allocated to MUs connected to the same MBS, which avoids intra-cell interference of each MBS in the uplink. However, considering the random mobility of MBSs and MUs, it is essential to take into account the interference generated by different MUs communicating with different MBSs when coverage overlaps.

Based on the description above, the MU n to MBS m wireless uplink rate at time slot t in the G2B communication link can be expressed as

$$R_{n,m}^{\mathcal{M}}(t) = W_n^{\mathcal{M}} \log_2 \left(1 + \frac{p_n^{\mathcal{N}} |h_{n,m}^{\mathcal{M}}|^2 (i_{n,m}^{\mathcal{M}}(t))^{-\iota}}{\sigma_{G2B}^2 + \zeta_{n',m'}^{\mathcal{M}}(t)} \right), \tag{10}$$

where $p_n^{\mathcal{N}}$ is transmission power of MU n, $h_{n,m}^{\mathcal{M}}$ denotes the unity channel gain between MU n and MBS m at the reference distance of $i_{n,m}^{\mathcal{M}}(t) = 1$ m, ι represents the path loss exponent and σ_{G2B}^2 indicates the noise power in G2B communication link. $\zeta_{n',m'}^{\mathcal{M}}(t)$ represents the interference from other MUs employing the same sub-band as MU n within the same coverage area but connected to different MBSs. Then we have

$$\zeta_{n',m'}^{\mathcal{M}}(t) = \sum_{m' \in \mathcal{M}/\{m\}} \alpha_{n,m'}^{\mathcal{M}}(t) \sum_{n' \in \mathcal{N}/\{n\}} \alpha_{n',m'}^{\mathcal{M}}(t) \eta_{n,n'}(t) p_{n'}^{\mathcal{N}} |h_{n',m'}^{\mathcal{M}}|^2 (i_{n',m'}^{\mathcal{M}}(t))^{-\iota}, \tag{11}$$

where $\eta_{n,n'}(t) \in 0, 1$, $\eta_{n,n'}(t) = 0$ signifies that MU n and MU n' are operating on different sub-bands at time slot t, while $\eta_{n,n'}(t) = 1$ indicates that MU n and MU n' are utilizing the same sub-band at time slot t.

G2A Communication Links. In the SAGIN, drones are pre-deployed in hotspot areas and the coverage of the drones are non-overlapping, so there is no inter-cell interference between different drones in G2A communications. Utilizing the OFDMA technology, the transmission rate between MU n and drone d in G2A communication link at time slot t is calculated as

$$R_{n,d}^{\mathcal{D}}(t) = W_n^{\mathcal{D}} \log_2 \left(1 + \frac{p_n^{\mathcal{N}} |h_{n,d}^{\mathcal{D}}|^2 (i_{n,d}^{\mathcal{D}}(t))^{-\iota}}{\sigma_{G2A}^2} \right), \tag{12}$$

where $W_n^{\mathcal{D}} = B^{\mathcal{D}}/N$ is the channel sub-bandwidth of the drones, similar to the G2B communication link, $B^{\mathcal{D}}$ is the operating band of each drone and divided equally into N sub-bands. $h_{n,d}^{\mathcal{D}}$ denotes the unity channel gain between MU n and drone d at the reference distance of $i_{n,d}^{\mathcal{D}}(t) = 1$ m, σ_{G2A}^2 indicates the noise power in G2A communication link.

G2S and A2S Communication Links. In the G2S and A2S communication link, the distance from the satellite to MUs and drones is much larger than the moving distance of MUs and drones, therefore, we ignore the influence of mobility on the G2S and A2S communication rate. The communication rate from MU n and drone d to the satellite at time slot t can be calculated as

$$R_n^{\mathcal{O}}(t) = W_n^{\mathcal{O}} \log_2 \left(1 + \frac{p_n^{\mathcal{N}} 10^{\frac{-L_n^{\mathcal{O}}(t)}{10}}}{\sigma_{G2S}^2} \right), \tag{13}$$

$$R_d^{\mathcal{R}}(t) = W_d^{\mathcal{R}} \log_2 \left(1 + \frac{p_d^{\mathcal{D}} 10^{\frac{-L_d^{\mathcal{R}}(t)}{10}}}{\sigma_{A2S}^2} \right), \tag{14}$$

where $W_n^{\mathcal{O}}$ and $W_d^{\mathcal{R}}$ is the size of channel sub-bands allocated by the satellite to MU n and drone d, respectively. $p_d^{\mathcal{D}}$ is transmission power of drone d, $L_n^{\mathcal{O}}(t)$ and $L_d^{\mathcal{R}}(t)$ denotes the path loss from MUs and drones to the satellite at time slot t, σ_{G2S}^2 and σ_{A2S}^2 indicates the noise power in G2S and A2S communication link. It is worth noting that the rate of the G2S communication link is lower than that of the G2A and A2S communication links, due to the longer transmission distance and the higher path loss.

2.5 Computation Model

For the offloading strategy $a_n(t) = (\mathbf{U}_n(t), \mathbf{C}_n(t))$ of MU n at time slot t, task $\mathcal{K}_n(t) = (k_n^{\mathcal{I}}(t), k_n^{\mathcal{C}}(t), k_n^{\mathcal{L}}(t))$ has five execution modes, including local execution, offloading to a MBS, offloading to a drone, offloading to satellite and offloading to satellite via a drone relay.

Local Execution. We define the task latency of MU n local execution as $\tau_n^{\mathcal{N}}(t)$, and $\tau_n^{\mathcal{N}}(t)$ only consists of the task computation latency, which can be calculated as

$$\tau_n^{\mathcal{N}}(t) = k_n^{\mathcal{C}}(t) / c_n^{\mathcal{N}}(t), \tag{15}$$

where $c_n^{\mathcal{N}}(t)$ is the computational resource of MU n at time slot t. Following the widely used energy consumption model [13], the energy consumption for local execution by MU n at time slot t can be calculated as

$$e_n^{\mathcal{N}}(t) = \varsigma (c_n^{\mathcal{N}}(t))^2 k_n^{\mathcal{C}}(t), \tag{16}$$

where ς represent the energy efficiency parameter, which varies based on the chip architecture. According to (15) and (16), the weighted cost that combines latency and energy consumption for locally executed tasks $\mathcal{K}_n(t)$ can be expressed as

$$\Phi_n^{\mathcal{N}}(t) = \lambda \tau_n^{\mathcal{N}}(t) + (1 - \lambda) e_n^{\mathcal{N}}(t), \forall n \in \mathcal{N}, \tag{17}$$

where $\lambda \in [0, 1]$ denotes the preference of MU n for balancing latency and energy consumption. For instance, if an underpowered MU aims to conserve battery life, it may lower the value of λ. Conversely, the parameters are adjusted to prioritize the latency.

Offloading to MBS, Drone or Satellite. We assume that MBSs, drones, and satellite can concurrently provide task offloading for multiple MUs, and we quantify the computational resources as the number of CPU cycles per second. If MU n offloading the task $\mathcal{K}_n(t) = (k_n^{\mathcal{I}}(t), k_n^{\mathcal{C}}(t), k_n^{\mathcal{L}}(t))$ to MBSs, drones or satellite, there are normally three processes to consider, including task upload, task computation, and result return. In common, the size of the task results is much

smaller than the size of the task inputs $k_n^{\mathcal{I}}(t)$. Furthermore, the downlink rate significantly surpasses the uplink rate, so we can neglect the delay and energy consumption associated with the result return phase. Based on the description above, the total delay of MU n offloading task $\mathcal{K}_n(t)$ to MBS m, drone d and satellite at time slot t can be calculated as

$$\begin{aligned}
\tau_{n,m}^{\mathcal{M}}(t) &= k_n^{\mathcal{I}}(t)/R_{n,m}^{\mathcal{M}}(t) + k_n^{\mathcal{C}}(t)/c_n^{\mathcal{M}}(t), \\
\tau_{n,d}^{\mathcal{D}}(t) &= k_n^{\mathcal{I}}(t)/R_{n,d}^{\mathcal{D}}(t) + k_n^{\mathcal{C}}(t)/c_n^{\mathcal{D}}(t), \\
\tau_n^{\mathcal{O}}(t) &= k_n^{\mathcal{I}}(t)/R_n^{\mathcal{O}}(t) + k_n^{\mathcal{C}}(t)/c_n^{\mathcal{O}}(t),
\end{aligned} \tag{18}$$

where $c_n^{\mathcal{M}}(t)$, $c_n^{\mathcal{D}}(t)$ and $c_n^{\mathcal{O}}(t)$ are the computational resources allocated to the MU n by the MBS, drone and satellite based on the offloading strategy made by the intelligent agent. The total energy consumption of MU n offloading task $\mathcal{K}_n(t)$ to MBS m, drone d and satellite at time slot t is obtained as

$$\begin{aligned}
e_{n,m}^{\mathcal{M}}(t) &= p_n^{\mathcal{N}} k_n^{\mathcal{I}}(t)/R_{n,m}^{\mathcal{M}}(t) + p_n^{\mathcal{W}} k_n^{\mathcal{C}}(t)/c_n^{\mathcal{M}}(t), \\
e_{n,d}^{\mathcal{D}}(t) &= p_n^{\mathcal{N}} k_n^{\mathcal{I}}(t)/R_{n,d}^{\mathcal{D}}(t) + p_n^{\mathcal{W}} k_n^{\mathcal{C}}(t)/c_n^{\mathcal{D}}(t), \\
e_n^{\mathcal{O}}(t) &= p_n^{\mathcal{N}} k_n^{\mathcal{I}}(t)/R_n^{\mathcal{O}}(t) + p_n^{\mathcal{W}} k_n^{\mathcal{C}}(t)/c_n^{\mathcal{O}}(t),
\end{aligned} \tag{19}$$

where $p_n^{\mathcal{W}}$ represents the power of MU n during the waiting for result return phase. Based on (18) and (19), the delay and energy weighted-cost of the offloading task $\mathcal{K}_n(t)$ to the MBS m, drone f and satellite at time slot t can be calculated as

$$\begin{aligned}
\Phi_n^{\mathcal{M}}(t) &= \lambda \tau_{n,m}^{\mathcal{M}}(t) + (1-\lambda) e_{n,m}^{\mathcal{M}}(t), \forall n \in \mathcal{N}, \\
\Phi_n^{\mathcal{D}}(t) &= \lambda \tau_{n,d}^{\mathcal{D}}(t) + (1-\lambda) e_{n,d}^{\mathcal{D}}(t), \forall n \in \mathcal{N}, \\
\Phi_n^{\mathcal{O}}(t) &= \lambda \tau_n^{\mathcal{O}}(t) + (1-\lambda) e_n^{\mathcal{O}}(t), \forall n \in \mathcal{N},
\end{aligned} \tag{20}$$

in practical terms, when $\lambda = 1$, the objective is to minimize the delay for MU n, whereas $\lambda = 0$ is aimed at significantly enhancing the battery life of MU n.

Offloading to Satellite via Drone Relay. When the G2S communication link is obstructed or the path loss is excessive, MUs can offload the task to satellite via a drone relay. In this case, the MU needs to upload the task to the drone first, then the drone offloading the task to the satellite for execution. The result return phase is again omitted, and the total task execution latency for MU n is

$$\tau_{n,d}^{\mathcal{R}}(t) = k_n^{\mathcal{I}}(t)/R_{n,d}^{\mathcal{D}}(t) + k_n^{\mathcal{I}}(t)/R_d^{\mathcal{R}}(t) + k_n^{\mathcal{C}}(t)/c_n^{\mathcal{R}}(t), \tag{21}$$

where $c_n^{\mathcal{R}}(t)$ is the computational resources assigned to the MU n through offloading to satellite via a drone relay. In this case, the total energy consumption of MU n at time slot t can be obtained as

$$e_{n,d}^{\mathcal{R}}(t) = p_n^{\mathcal{N}} k_n^{\mathcal{I}}(t)/R_{n,d}^{\mathcal{D}}(t) + p_n^{\mathcal{W}} \left(k_n^{\mathcal{I}}(t)/R_d^{\mathcal{R}}(t) + k_n^{\mathcal{C}}(t)/c_n^{\mathcal{R}}(t) \right), \tag{22}$$

similar to the definitions previously, the weighted-cost of latency and energy consumption in this case can be calculated as

$$\Phi_n^{\mathcal{R}}(t) = \lambda \tau_{n,d}^{\mathcal{R}}(t) + (1-\lambda) e_{n,d}^{\mathcal{R}}(t), \forall n \in \mathcal{N}. \tag{23}$$

Let $C^{\mathcal{M}}$, $C^{\mathcal{D}}$ and $C^{\mathcal{O}}$ indicate the overall computational resources of MBSs, drones and the satellite. Based on the definition above, we can get the computational resource constraint at time slot t as

$$\sum_{n\in\mathcal{N}} c_n^{\mathcal{M}}(t) \le C^{\mathcal{M}}, \sum_{n\in\mathcal{N}} c_n^{\mathcal{D}}(t) \le C^{\mathcal{D}}, \sum_{n\in\mathcal{N}} \left(c_n^{\mathcal{O}}(t) + c_n^{\mathcal{R}}(t)\right) \le C^{\mathcal{O}}. \tag{24}$$

3 Problem Formulation

In this section, we formulate the multi-node collaborative task offloading optimization problem to jointly reduce the task computation latency and energy consumption of the MUs. Based on the models above and the offloading strategy $a(t)$ made by the intelligent agent for all MUs at time slot t, we can define the average cost of MUs weighted by delay and energy consumption at time slot t as

$$\overline{\Phi}(t) = \frac{1}{N}\sum_{n\in\mathcal{N}} \left(u_n^{\mathcal{N}}(t)\Phi_n^{\mathcal{N}}(t) + u_n^{\mathcal{M}}(t)\Phi_n^{\mathcal{M}}(t) + u_n^{\mathcal{D}}(t)\Phi_n^{\mathcal{D}}(t) + u_n^{\mathcal{O}}(t)\Phi_n^{\mathcal{O}}(t) + u_n^{\mathcal{R}}(t)\Phi_n^{\mathcal{R}}(t)\right). \tag{25}$$

Our goal is to minimize the average cost of MUs through optimizing the offloading strategies made by the intelligences at each time slot. Based on the description above, considering the constraints, including execution modes, tolerance delay, computational resources, coverage of edge nodes and mobility of MUs and MBSs. We formulate the problem of minimizing the average cost of MUs over a T-slot time horizon as

$$\begin{aligned}
&\mathbf{P1}: \min_{a(t)} \frac{1}{T}\sum_{t\in\mathbb{T}} \overline{\Phi}(t) \qquad (26)\\
\text{s.t. } &C1: u_n^{\mathcal{N}}(t), u_n^{\mathcal{M}}(t), u_n^{\mathcal{D}}(t), u_n^{\mathcal{O}}(t), u_n^{\mathcal{R}}(t) \in \{0,1\}, \forall n \in \mathcal{N},\\
&C2: u_n^{\mathcal{N}}(t) + u_n^{\mathcal{M}}(t) + u_n^{\mathcal{D}}(t) + u_n^{\mathcal{O}}(t) + u_n^{\mathcal{R}}(t) = 1, \forall n \in \mathcal{N},\\
&C3: |\mathbf{v}_n^{\mathcal{N}}(t)| \le V_{max}^{\mathcal{N}}, |\mathbf{v}_m^{\mathcal{M}}(t)| \le V_{max}^{\mathcal{M}}, \forall n \in \mathcal{N}, \forall m \in \mathcal{M},\\
&C4: \mathbf{O}_n(t)[I_n^{\mathcal{U}}(t) - 1] = 1, \forall n \in \mathcal{N},\\
&C5: u_n^{\mathcal{N}}(t)\tau_n^{\mathcal{N}}(t) + u_n^{\mathcal{M}}(t)\tau_{n,m}^{\mathcal{M}}(t) + u_n^{\mathcal{D}}(t)\tau_{n,d}^{\mathcal{D}}(t)\\
&\qquad\qquad + u_n^{\mathcal{O}}(t)\tau_n^{\mathcal{O}}(t) + u_n^{\mathcal{R}}(t)\tau_{n,d}^{\mathcal{R}}(t) \le k_n^{\mathcal{L}}(t), \forall n \in \mathcal{N},\\
&C6: 0 \le c_n^{\mathcal{M}}(t) \le C^{\mathcal{M}}, 0 \le c_n^{\mathcal{D}}(t) \le C^{\mathcal{D}}, 0 \le c_n^{\mathcal{O}}(t), c_n^{\mathcal{R}}(t) \le C^{\mathcal{O}}, \forall n \in \mathcal{N},\\
&C7: \sum_{n\in\mathcal{N}} c_n^{\mathcal{M}}(t) \le C^{\mathcal{M}}, \sum_{n\in\mathcal{N}} c_n^{\mathcal{D}}(t) \le C^{\mathcal{D}}, \sum_{n\in\mathcal{N}} \left(c_n^{\mathcal{O}}(t) + c_n^{\mathcal{R}}(t)\right) \le C^{\mathcal{O}}, \forall n \in \mathcal{N},
\end{aligned}$$

where $C1$ and $C2$ show that MUs can only choose a single execution mode at time slot t. $C3$ ensures that the distance moved by MUs and MBSs does not exceed the maximum limit at time slot t. $C4$ indicates that MUs can only select optional execution methods to perform tasks. $C5$ represents that the execution delay of tasks does not exceed the maximum tolerable delay. $C6$ and $C7$ mean that the computational resources allocated to the MUs and the total used computational resources are not exceeding the resource limits of the edge nodes, respectively.

4 DRL-Based Task Offloading Optimization Scheme

P1 is a mixed-integer nonlinear programming problem involving binary and continuous variables, so it is challenging to obtain an optimal or suboptimal solution. DRL has strong learning ability and adaptivity to handle highly complex problems, thus we use MDP-based Proximal Policy Optimization (PPO) algorithm to solve **P1**. In this section, we introduce the MDP model and the PPO framework, then we propose the DTOOA scheme to optimize the cost of MUs.

4.1 MDP Model

An MDP model can be characterized by a 4-tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R})$, where $\mathcal{S}$ represents the set of system states, and $s(t)$ indicates the system state at time slot t. $\mathcal{A}$ show the set of system actions, and $a(t)$ denotes the system action at time slot t. $\mathcal{P}$ signifies the probability of transitioning from state $s(t)$ to $s(t+1)$ when action $a(t)$ is taken, and $\mathcal{R}$ represents the immediate reward associated with action $a(t)$.

State Space. At the offloading request phase, the intelligent agent summarizes the state space consisting of offload requests, wireless network state and computational resources of edge nodes. Then state $s(t) \in \mathcal{S}$ at time slot t can be expressed as

$$s(t) = \{\mathbf{K}(t), \mathbf{O}(t), \mathbf{R}(t), \mathbf{C}(t)\}, \tag{27}$$

where $\mathbf{K}(t) = (\mathbf{k}^{\mathcal{I}}(t), \mathbf{k}^{\mathcal{C}}(t), \mathbf{k}^{\mathcal{L}}(t))$ denotes the set of task information for all MUs at time slot t, including input data size, computing amount, and maximum acceptable latency. $\mathbf{O}(t)$ represents the set of optional task execution mode vector for all MUs. $\mathbf{R}(t)$ is the set of all communication link rates, including the G2B, G2A and G2S communication rates for all MUs and the A2S communication rate for all drones. $\mathbf{C}(t)$ is the set of computational resources available for all the edge nodes. It is worth noting that the changes in the coordinates of all nodes are embodied in $\mathbf{O}(t)$ and $\mathbf{R}(t)$, so the state space can omit the coordinates of the nodes.

Action Space. At the offloading strategy phase, the intelligent agent aggregate system state information and takes action $a(t) \in \mathcal{A}$, $a(t)$ was defined previously as

$$a(t) = \{\mathbf{U}(t), \mathbf{C}(t)\}, \tag{28}$$

where $\mathbf{U}(t)$ is execution mode vector of all MUs at time slot t, and $\mathbf{C}(t)$ denotes the computational resource allocation vector of all MUs.

System Reword P1. strives to minimize the average cost for MUs, whereas MDP is commonly employed to maximize the long-term rewards of the system.

Consequently, we can leverage MDP to maximize the negative value of the average cost of MUs. To encourage the intelligent agent to select actions aligning with the delay constraints $C5$ of MUs, we configure the reward as

$$r(t) = \begin{cases} -\overline{\Phi}(t), & the\ C5\ is\ satisfied, \\ -N^*(t)\overline{\Phi}(t), & else, \end{cases} \tag{29}$$

where $N^*(t)$ denotes the count of MUs that fall short of meeting constraint $C5$ at time slot t. If constraint $C5$ is satisfied, the immediate reward corresponds to the negative value of the average user cost. Conversely, if constraint $C5$ is not met, the immediate reward takes the form of a penalty that is calculated as the negative of the average user cost multiplied by the number of MUs failing to satisfy condition $C5$. As a result, the intelligent agent exhibits a preference for actions that ensure compliance with constraint $C5$ at time slot t.

4.2 PPO Framework

In the MDP model above, the computational resource allocation vectors in the action space are continuous, so we propose the DTOOA scheme based on the PPO framework to find the optimal task execution mode vector and computational resource allocation vector for MUs. The PPO is an on-policy, model-free DRL algorithm which improves the stability by controlling the magnitude of policy updates during training. PPO framework belongs to the actor-crtic network structure, which normally has three neural networks, including the actor network π_θ, the old actor network $\pi_{\theta_{old}}$ and the crtic network Q_φ, where θ, θ_{old} and φ denote the parameters of actor network, old actor network and crtic network, respectively.

The actor network is the policy network, which is responsible for outputting the probability distribution of actions in a given state $s(t)$. The old actor network is used to compute the ratio of new to old policies, thus limiting the changing magnitude of the new policy. The actor network is updated by gradient ascent method $\theta \leftarrow \theta + \xi \cdot \nabla_\theta \mathcal{L}(\theta)$, where ξ is the learning rate and $\mathcal{L}(\theta)$ is the objective function. In order to ensure the stability of training, PPO defines a clipped surrogate objective function, which is expressed as

$$\mathcal{L}(\theta) = \mathbb{E}\left[\min\left(\frac{\pi_\theta(a(t)|s(t))}{\pi_{\theta_{old}}(a(t))|s(t))} \cdot \hat{A}(t), clip\left(\frac{\pi_\theta(a(t)|s(t))}{\pi_{\theta_{old}}(a(t))|s(t))}, 1-\epsilon, 1+\epsilon\right) \cdot \hat{A}(t)\right)\right], \tag{30}$$

where $\hat{A}(t)$ is the advantage function, which is used to evaluate whether the new strategy is better than the old one. Clip function is used to limit the ratio $\frac{\pi_\theta(a(t)|s(t))}{\pi_{\theta_{old}}(a(t))|s(t))}$ to the interval $[1-\epsilon, 1+\epsilon]$, and ϵ is a hyper-parameter that limits the ratios of the new to old policies. The advantage function is obtained from the critic network computation which defined as

$$\hat{A}(t) = \sum_{j=1}^{T-t} \gamma^j \cdot r(t+j) - Q_\varphi(s(t)), \tag{31}$$

where T denotes the maximum time slot, $r(t+j)$ represents the reward for future steps, and γ is the discount factor. $Q_\varphi(s(t))$ is the state-valued function output by the critic network. Critic network is updated by gradient descent method $\varphi \leftarrow \varphi - \xi \cdot \nabla_\varphi \mathcal{L}(\varphi)$ and loss function $\mathcal{L}(\varphi)$ can be expressed as

$$\mathcal{L}(\varphi) = \mathbb{E}\left[\left(\sum_{j=1}^{T-t} \gamma^j \cdot r(t+j) - Q_\varphi(s(t))\right)^2\right]. \tag{32}$$

4.3 DRL-Based Task Offloading Optimization Algorithm

The intelligent agent deployed in the satellite run the DTOOA algorithm, which collects environmental information from 6G SAGIN and makes offloading decisions. In the early stage of operation, the DTOOA algorithm needs to be trained for several iterations and reach convergence, after which the intelligent agent starts to make actual decisions.

Algorithm 1. DRL-Based Task Offloading Optimization Algorithm (DTOOA)

Initialize the actor network π_θ, the old actor network $\pi_{\theta_{old}}$ and the crtic network Q_φ with parameters θ, $\theta_{old} \leftarrow \theta$ and φ;
Initialize the replay memory buffer $\mathcal{B}$;
for *episode* $= 1$ *to* $MaxEpisode$ **do**
  Initialize the 6G SAGIN environment and obtain the initial state $s(1)$;
  for $t = 1$ *to* T **do**
    Execute action $a(t)$ generated by observing state $s(t)$ through the actor network π_θ;
    Calculate reward $r(t)$ based on (29) and get the next state $s(t+1)$;
    Save experience $(s(t), a(t), r(t), s(t+1))$ to the replay memory buffer $\mathcal{B}$;
  end for
  for each *experience* in replay memory buffer $\mathcal{B}$ **do**
    Update the actor network π_θ by computing the objective function (30) using a single experience;
    Update the critic network Q_φ by calculate the loss function (32) using a single experience;
  end for
  Update the old actor network $\pi_{\theta_{old}}$ with the parameter θ;
  Clear the experience replay buffer $\mathcal{B}$;
end for

At the start of time slot t, the intelligent agent makes an action $a(t)$ depending on the current state $s(t)$, gets the next state $s(t+1)$ and receives an immediate reward $r(t)$. Then, experience $(s(t), a(t), r(t), s(t+1))$ is saved to the replay memory buffer $\mathcal{B}$. After a episode, iterating through each experience in the replay memory buffer $\mathcal{B}$ to updates the actor and critic networks. Finally, the old actor network is updated and the replay memory buffer is cleared. For clarity, Algorithm 1 summarizes the details of DTOOA.

5 Numerical Results and Discussion

In this section, we utilize Python 3.9 and TensorFlow 2.0 to create a simulation platform and construct the 6G SAGIN model to evaluate the performance of our proposed DTOOA scheme. We first investigate the impact of internal parameters on the performance of the DTOOA scheme. Subsequently, we proceed to compare DTOOA against the other three schemes within various environments.

5.1 Simulation Setup

In the simulation scenario, we consider a 500 m radius circular network range with seamless satellite coverage, and the satellite $\mathcal{O}$ coordinate is $(0, 0, 300000)$ with units of meters. The coordinates of the $D = 2$ drones are $(-250, -250, 120)$ and $(250, 250, 120)$, and the coverage radius $l_d^{\mathcal{D}}$ is 100 m. The number of MUs in the range of $[10, 25]$ and $M \in \{2, 4, 6, 8, 10\}$ MBSs with $l_m^{\mathcal{M}} = 200$ m coverage radius stochastically distributed in the network range. The duration of each time slot ψ is 1 s, MBS and MU are moved within each time slot with maximum movement distances $V_{max}^{\mathcal{M}} = 10$ m and $V_{max}^{\mathcal{N}} = 3$ m, respectively. With reference to the communication and computational parameter settings in [1,2] and [3], Table 1 summarizes the default simulation parameters in our system model.

Table 1. Default Simulation Parameters.

Parameters	Values
Computational resources of each satellite/drone/MBS/MU	20/8/10/1 GHz
Operating band of each satellite/drone/MBS	24/5/10 MHz
Size of the task input data $k_n^{\mathcal{I}}(t)$	2–20 MB
Amount of computation required for the task $k_n^{\mathcal{C}}(t)$	0.5–3 Gigacycles
Maximum tolerable latency of the task $k_n^{\mathcal{L}}(t)$	0.8–1 s
Transmission power of each MU/drone $p_n^{\mathcal{N}}/p_d^{\mathcal{D}}$	0.2/1.6 W
Waiting power of each MU $p_n^{\mathcal{W}}$	0.05 W
Unity channel gain $h_{n,m}^{\mathcal{M}}/h_{n,d}^{\mathcal{D}}$	0/−50 dB
Path loss exponent ι	4
Background noise power $\sigma_{G2B}^2/\sigma_{G2A}^2$	−104/−104 dBm
Background noise power $\sigma_{G2S}^2/\sigma_{A2S}^2$	−114/−114 dBm
Path loss $L_n^{\mathcal{O}}(t)/L_d^{\mathcal{R}}(t)$	150–174/150 dB
Energy efficiency parameter ς	10^{-28}
Preference for latency and energy consumption λ	0.5

For DTOOA, both the actor and critic networks consist of four-layer neural architectures comprising an input layer, two hidden layers, and an output layer. The actor network features 512 neurons in both of its hidden layers, whereas

the critic network is composed of two hidden layers with 512 and 256 neurons, respectively. Besides, The $MaxEpisode$ is 1000 and the T is 20, the discount factor γ is 0.9, the ϵ used for clipping the surrogate objective is 0.2, and the mini-batch size is 64.

5.2 Parameter Analysis

In the simulated environment with $M = 6$ MBSs and $N = 10$ MUs, $k_n^{\mathcal{C}}(t)$, $k_n^{\mathcal{C}}(t)$ and $k_n^{\mathcal{L}}(t)$ obeying Gaussian distributions with mean values of 11 MB, 1.75 Gigacycles and 0.9 s, the path loss $L_n^{\mathcal{O}}(t)$ takes random values in the interval $[150, 174]$ dB, we evaluated the impact of learning rates on the convergence of the DTOOA scheme and conducted multiple experiments in the learning rate interval $[10^{-2}, 10^{-5}]$. We found that DTOOA converges only when the learning rate is near 1×10^{-4}. Figure 2 illustrates the convergence of the algorithm for learning rate is 5×10^{-4}, 1×10^{-4}, and 0.5×10^{-4}. With a learning rate set at 5×10^{-4}, the system learning rate becomes excessively close to 1, leading to an inability to learn the optimal solution and consequently resulting in a low system reward. When learning rate is 0.5×10^{-4}, the cumulative reward of the algorithm attains convergence after 700 iterations, exhibiting a slower convergence rate compared to the optimal scheme. When learning rate is 1×10^{-4}, the cumulative reward and convergence speed show notable improvements compared to the other cases, so we use 1×10^{-4} as the learning rate for subsequent simulation experiments.

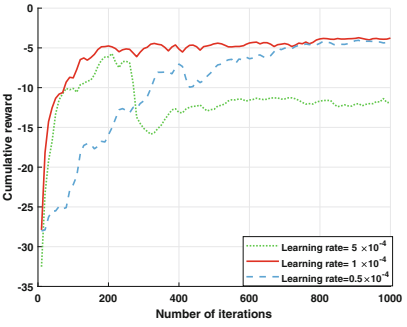

Fig. 2. Impact of different learning rate.

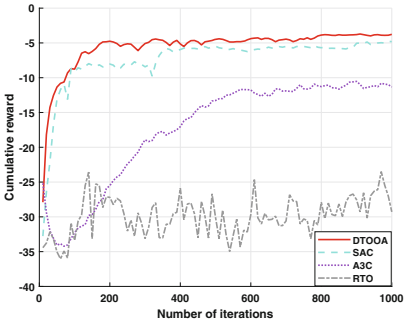

Fig. 3. Impact of various comparison algorithm.

5.3 Performance Analysis

To assess the algorithmic performance of DTOOA, we conduct a comparative evaluation against Soft Actor-Critic (SAC), Asynchronous Advantage Actor-Critic (A3C), and Random Task Offloading (RTO). Both SAC and A3C are

DRL algorithms based on actor-critic structure, SAC combines the features of off-policy and maximum entropy to improve the convergence [14] and A3C improves the training utility and training speed by asynchronous training [11]. In RTO, the actions are given randomly under the condition that the constraints are satisfied. In the performance analysis simulations, we set the default parameters as $M = 6$ MBSs and MUs, $N = 10$ MUs, $k_n^{\mathcal{C}}(t)$, $k_n^{\mathcal{C}}(t)$ and $k_n^{\mathcal{L}}(t)$ obeying Gaussian distributions with mean values of 11 MB, 1.75 Gigacycles and 0.9 s, the path loss $L_n^{\mathcal{O}}(t)$ takes random values in the interval $[150, 174]$ dB. In subsequent simulations, when analyzing the impact of a particular parameter on the long-term reward of the system, the default parameters were used for all other parameters.

System Reward Trend. With the simulation environment set as the default parameters, we evaluated the convergence trend of the cumulative rewards for the four schemes with the number of iterations. As shown in Fig. 3, our proposed DTOOA scheme has optimal convergence rewards and convergence speeds, and the cumulative rewards after convergence are 2.98% better than SAC and 21.62% better than A3C. Due to the clipping of the surrogate objective function in PPO, the DTOOA scheme has excellent stability and convergence, and can better cope with the situation that the state space changes abruptly, such as when the mobility of the MUs and MBSs leads to an abrupt change for the optional task execution mode vector $\mathbf{O}(t)$ of the MUs. The SAC scheme has higher cumulative rewards after convergence, but the convergence rate is inferior to the DTOOA scheme. The A3C scheme is inferior to the DTOOA and SAC schemes in terms of convergence speed and cumulative rewards, and the RTO scheme has no convergence trend.

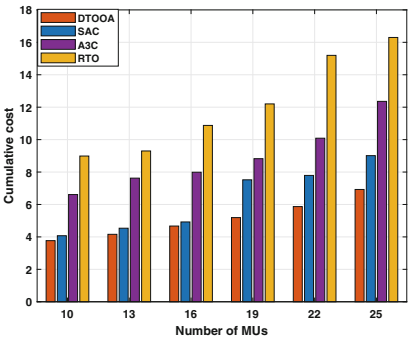

Fig. 4. Impact of different number of MUs.

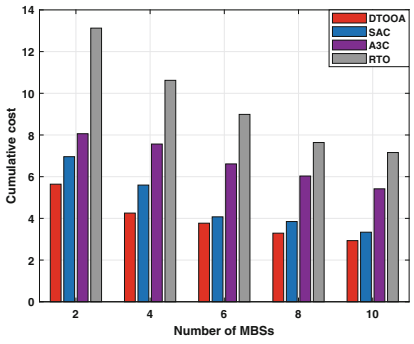

Fig. 5. Impact of different number of MBSs.

Number of MUs. In order to more visually represent the improvement in QoE of MUs by DTOOA, we compare the impact of the four schemes on the cumulative cost instead of the cumulative reward after convergence. As shown in Fig. 4, with the default parameter settings and setting the number of MUs to $[10, 25]$ with an increment of 3, we evaluate the impact of different numbers of MUs on the cumulative cost. The task computation requirements increase with the number of MUs, while the available wireless and computational resources are limited, thus the latency and energy cost for the MUs are increased. When the number of MUs increases to 25, the DTOOA scheme reduces the cumulative cost by 23.31% and 43.97% compared to the SAC and A3C schemes, respectively.

Number of MBSs. In the simulation environment with default parameter settings and with the numbers of MBSs set to $[2, 10]$ in increments of 2, Fig. 5 gives the trend of cumulative cost with different numbers of MBSs. The more the number of MBSs, the more edge nodes can be requested and the more computing resources can be allocated in the SAGIN system, thus reducing the cumulative cost and improving the QoE of MUs. When the number of MBSs is 2, compared to the SAC, A3C and RTO schemes, the DTOOA scheme reduces the cumulative cost by 18.97%, 30.02% and 56.97%, respectively.

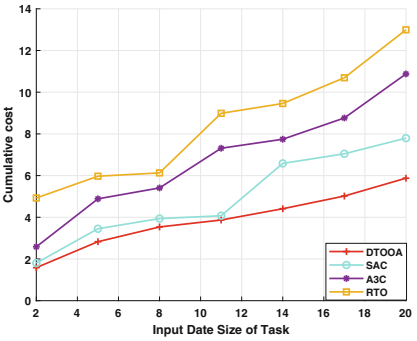

Fig. 6. Impact of different input data size of task.

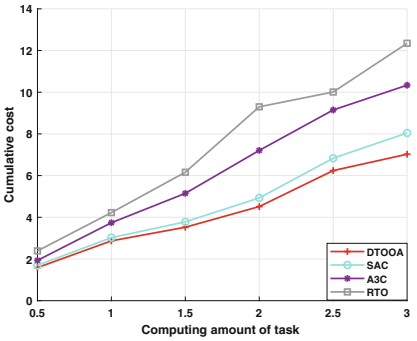

Fig. 7. Impact of different computing amount of task.

Input Data Size of Task. As shown in Fig. 6, in the simulation environment with default parameter settings and with the size of the task input data set to [2,20] in increments of 2, we analyze the impact of different task input data size on cumulative cost. The increase in the input data size of task leads to larger communication latency and energy consumption, which makes the cumulative cost increase. The curve of the DTOOA scheme is smooth, so the DTOOA scheme is insensitive to changes in the input data size of task and it is able to optimize task offloading in a stable manner. In addition, the DTOOA scheme reduces the

cumulative cost by 24.28% and 45.98% compared to the SAC and A3C schemes, respectively. Compared to the RTO scheme, it reduces the cumulative cost by 54.77%.

Computing Amount of Task. In the simulation environment where the computing amount of task is $[0.5, 3]$ Gigacycles in increments of 0.5 and other parameters are default parameters, The impact of different computing amount of task on the cumulative cost is shown in Fig. 7. Due to the restricted computing resources of the edge nodes, the larger the task computation volume, the higher the task computation latency and energy consumption, which makes the cumulative cost increase. The DTOOA scheme reduces cumulative costs by 12.66%, 32.02% and 41.21% over the SAC, A3C and RTO schemes, respectively.

6 Conclusion and Future Work

In this paper, we study the task offloading optimization problem in the 6G SAGIN scenario. In particular, we consider the modeling of multiple heterogeneous edge nodes and multiple offloading methods, meanwhile, the mobility of MUs and MBSs are considered. Temporally changing network environment and complex offloading scenario make the optimal task offloading problem challenging. First, we provide a detailed description of the task offloading process and establish various models in the SAGIN scenario. Then, we propose the multi-node collaborative task offloading optimization problem to jointly reduce the task computation latency and energy consumption of MUs. Finally, we propose the DTOOA scheme based on the Proximal Policy Optimization (PPO) framework to optimize the average cost of MUs, and conduct extensive simulation experiments. Numerical results show that the DTOOA scheme clearly outperforms the Soft Actor-Critic (SAC), Asynchronous Advantage Actor-Critic (A3C), and Random Task Offloading (RTO) schemes in a variety of different simulation environments, and can significantly reduce the average cost of MUs and improve the QoE of MUs.

In future work, considering a joint multi-satellite network to provide seamless assisted computation and caching for a region is a potential direction. In addition, modeling MBSs and drones for path planning to provide better service is also a viable option.

References

1. Cao, X., et al.: Edge-assisted multi-layer offloading optimization of LEO satellite-terrestrial integrated networks. IEEE J. Sel. Areas Commun. **41**(2), 381–398 (2023)
2. Gong, Y., Yao, H., Xiong, Z., Guo, S., Yu, F.R., Niyato, D.: Computation offloading and energy harvesting schemes for sum rate maximization in space-air-ground networks. In: GLOBECOM 2022 - 2022 IEEE Global Communications Conference, pp. 3941–3946 (2022)

3. He, L., Li, J., Wang, Y., Zheng, J., He, L.: Balancing total energy consumption and mean makespan in data offloading for space-air-ground integrated networks. IEEE Trans. Mob. Comput. **23**(1), 209–222 (2024)
4. Jin, X., Hua, W., Wang, Z., Chen, Y.: A survey of research on computation offloading in mobile cloud computing. Wireless Netw. **28**(4), 1563–1585 (2022)
5. Liu, H., Cao, G.: Deep reinforcement learning-based server selection for mobile edge computing. IEEE Trans. Veh. Technol. **70**(12), 13351–13363 (2021)
6. Ning, Z., Yang, Y., Wang, X., Guo, L., Gao, X.: Dynamic computation offloading and server deployment for UAV-enabled multi-access edge computing. IEEE Trans. Mob. Comput. **22**(5), 2628–2644 (2023)
7. Qin, P., Wang, M., Zhao, X., Geng, S.: Content service oriented resource allocation for space-air-ground integrated 6G networks: a three-sided cyclic matching approach. IEEE Internet Things J. **10**(1), 828–839 (2023)
8. Shen, H., Tian, Y., Wang, T., Bai, G.: Slicing-based task offloading in space-air-ground integrated vehicular networks. IEEE Trans. Mob. Comput. **23**(5), 4009–4024 (2024)
9. Tang, F., Hofner, H., Kato, N., Kaneko, K., Yamashita, Y., Hangai, M.: A deep reinforcement learning-based dynamic traffic offloading in space-air-ground integrated networks (SAGIN). IEEE J. Sel. Areas Commun. **40**(1), 276–289 (2022)
10. Wang, C.X., You, X., Gao, X., Zhu, X., Li, Z.: On the road to 6G: visions, requirements, key technologies, and testbeds. IEEE Commun. Surv. Tutor. **25**(2), 905–974 (2023)
11. Wang, Z., Li, M., Zhao, L., Zhou, H., Wang, N.: A3C-based computation offloading and service caching in cloud-edge computing networks. In: IEEE INFOCOM 2022 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), pp. 1–2 (2022)
12. Waqar, N., Hassan, S.A., Mahmood, A., Dev, K., Do, D.T.: Computation offloading and resource allocation in MEC-enabled integrated aerial-terrestrial vehicular networks: a reinforcement learning approach. IEEE Trans. Intell. Transp. Syst. **23**(11), 21478–21491 (2022)
13. Yang, B., Cao, X., Bassey, J., Li, X.: Computation offloading in multi-access edge computing: a multi-task learning approach. IEEE Trans. Mob. Comput. **20**(9), 2745–2762 (2021)
14. Zhou, X., Huang, L., Ye, T., Sun, W.: Computation bits maximization in UAV-assisted MEC networks with fairness constraint. IEEE Internet Things J. **9**(21), 20997–21009 (2022)