Reducing Latency in NOMA-aided MEC Networks: A Deep Reinforcement Learning Approach

Shuang Zhang, Pingkang Guo, and Huilong Jin
College of Engineering, Hebei Normal University, Shijiazhuang 050024, China
Emails: {zshuang, JHL981}@hebtu.edu.cn, gpkang@stu.hebtu.edu.cn

***Abstract*—This paper delves into the latency reduction in mobile edge computing (MEC) networks augmented by non-orthogonal multiple access (NOMA) under flexible timing conditions. We introduce an iterative mechanism for task offloading between two users in a NOMA setup, prioritizing the optimization of phase shifts in a Reconfigurable Intelligent Surface (RIS) to curtail uplink latency. Drawing inspiration from deep reinforcement learning's (DRL) adeptness in handling complex control challenges, we innovatively crafted a deep deterministic policy gradient (DDPG)-based time-delay minimization (DDPR) algorithm, to adeptly navigate the non-convex optimization inherent in latency minimization. Numerical results suggest that our proposed system markedly reduces latency in a brief period, outperforming current methods.**

***Index Terms*—Non-Orthogonal Multiple Access (NOMA), Mobile Edge Computing (MEC), Reconfigurable Intelligent Surface (RIS), Deep Reinforcement Learning (DRL), Latency Reduction**

I. Introduction

Recent years have witnessed a surging interest in Reconfigurable Intelligent Surfaces (RIS), with significant endeavors directed towards their assimilation into traditional wireless networks. Such integration is intended to augment communication efficiency and foster the emergence of innovative applications. The advent of RIS has precipitated a plethora of groundbreaking technological uses, encompassing RIS-augmented non-orthogonal multiple access (NOMA) [1], RIS-enhanced physical layer security [2], and RIS-supported mobile edge computing (MEC) networks [3]. The RIS-supplemented wireless transmission framework signifies the vanguard of wireless communication technology, hinged on the meticulous modulation of phase to markedly enhance communicative performance. It is engineered to satisfy the demands of extensive connectivity, amplify the efficacy of spectral utilization and curtail latency [4]. In scenarios where orthogonal multiple access (OMA) delegates tasks to peripheral highly-mobile nodes, the scarcity of spectral resources may precipitate escalated transmission latency [5]. NOMA adeptly navigates this challenge by permitting multiple terminals to concurrently inhabit identical frequency resources. Consequently, within the purview of NOMA communication paradigms, RIS is projected to evolve into a ubiquitously acknowledged pivotal communication technology, bolstering edge cloud computing in the quest for formidable capacity and velocity.

Research has demonstrated that the distinctive attributes of RIS can notably enhance the spectral efficiency of wireless communication systems [6], garnering substantial research interest. Despite its significant technical potential, the integration of RIS into MEC under NOMA remains in the nascent stage of exploration due to the intricacies involved in joint resource allocation. The optimal scheduling with delay considerations and the access schemes for RIS-aided MEC systems are yet to be fully understood.

Existing studies on phase configuration have implemented various methodologies, including Software-Defined Radio (SDR) and Full-Phase Iteration (FPI) algorithms [7], [8]. However, these approaches encounter issues such as high complexity and performance degradation. More recent research [9] has explored greedy cell iterative optimization techniques, which prove inefficient in network systems. The authors in [10] have demonstrated that strategic RIS deployment and beamforming design can markedly enhance communication rates between users and cloud infrastructure, consequently reducing computation offload latency. Yet, these studies often oversimplify scenarios where data transfer occurs either simultaneously or sequentially, neglecting the influence of RIS on user compute offload scheduling across different multiple access contexts. With the advancements in AI, DRL-based methods are devoid of this prerequisite and boast the advantages of online learning and dynamic sample generation, offering more efficient storage utilization [11]. In [11], the authors introduce a novel DRL framework to address resource allocation challenges in NOMA, comprising two distinct modules: one for discrete variable optimization and the other for continuous variable resolution. Nonetheless, the framework structure is associated with a proportional increase in training complexity.

In this work, we introduce a DRL-based algorithm known as DDPR, aimed at reducing the latency within MEC networks. The key contributions of our research are outlined as follows:

- The system is configured under a flexible time-division NOMA framework, where users transmit data to the MEC cloud computing server following a scheduled offloading sequence. Subsequently, edge cloud computing is executed under Time Division Multiple Access (TDMA) communication. We propose an iterative two-

This work was supported by Science and Technology Project of Hebei Education Department (Grant No. QN2023233), Shijiazhuang Science and Technology Plan Project (241791277A).

Corresponding author: Huilong Jin.

ISBN 978-3-903176-65-2

user NOMA scheme for MEC offloading designed to minimize the delay associated with offloading processes.
- To address the RIS phase adjustment challenge within the NOMA-MEC network as a DRL task, we introduce an innovative DDPR algorithm that is based on the Deep Deterministic Policy Gradient (DDPG) framework. This algorithm employs an Actor network to devise distinct actions and a Critic network to estimate the Q-valued action function. This approach is intended to diminish the exploration time required by the Agent to traverse the solution space, thereby hastening the network's convergence.
- To facilitate the Agent's learning of the optimal actions We have developed a min-max normalization technique, which is based on the DDPR component of the DDPG algorithm, to pre-process the observed states. Concurrently, we have enhanced the activation function. The configuration of the neural network parameters is expected to markedly enhance the algorithm's performance and rate of convergence, rendering the network training process more efficacious.

The rest of this manuscript is delineated as follows. Section II presents the system model, encompassing the channel model, the task offloading model, and a detailed exposition of the optimization problem at hand. Section III addresses the resolution of the primary research issue, proposing a solution algorithmic framework grounded in the DDPR approach. Section IV encompasses system simulations, analytical discussions, and the presentation of findings. Section V synthesizes the research and concludes the paper.

II. SYSTEM MODEL & PROBLEM FORMULATION

A. Communication Model

In this work, we examine MEC systems augmented by a RIS, as depicted in Fig. 1. The RIS is strategically positioned to facilitate the computation offloading process from two users, each equipped with a single antenna, to a single-antenna Access Point (AP) that acts as an entry point to the edge cloud. The RIS incorporates a reflective element and is controlled by a micro-controller.

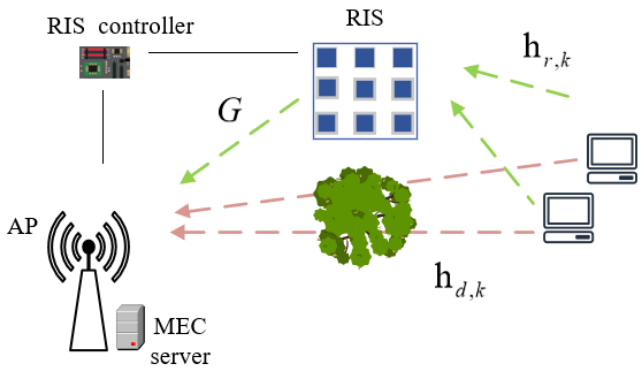

Fig. 1: Example of RIS-aided MEC system.

The channel model is comprised of path loss and small-scale fading components. It is postulated that the direct channel from the Access Point (AP) to the user undergoes Rayleigh fading. Consequently, for RIS-aided wireless communication systems, the channel model between the AP and user k can be articulated as follows:

$$\mathrm{h}_k = \mathrm{h}_{d,k} + G\Theta\mathrm{h}_{r,k}, \tag{1}$$

where $\mathrm{h}_{r,k} \in \mathbb{R}^{N\times 1}$ represents the channel response vector between the RIS and the user k, $G \in \mathbb{R}^{1\times N}$ represents the channel response vector between the RIS and the AP, $\mathrm{h}_{d,k}$ represents the channel response vector between user k and the AP, $\Theta = \mathrm{diag}\{\emptyset_1, \ldots, \emptyset_N\}$, $\emptyset_n = e^{j\theta_n}$, with θ_n representing the n-th element phase shift in the RIS. Here, we assume an ideal reflection of RIS such that the signal power is lossless from each reflecting element.

B. Computing Offloading Model

The computational workload generated by the k-th ground user during the t-th time interval is defined as $I_k(t) = \{D_k(t), F_k(t)\}$, where $D_k(t)$ signifies the data volume of the computation task, and $F_k(t)$ indicates the CPU cycles necessary to accomplish the task. Regarding data transmission, it is presumed that two users, π_1 and π_2, initially send their data concurrently during a shared transmission period by employing NOMA. Subsequently, π_1 and π_2 proceed with task offloading in accordance with the predetermined scheduling sequence. Users upload their computational tasks to the designated AP for execution, with the MEC server possessing restricted computational capabilities. It is hypothesized that each MEC server handles one task sequentially, causing other tasks to queue in the waiting list. The latency incurred by users due to waiting must be taken into account.

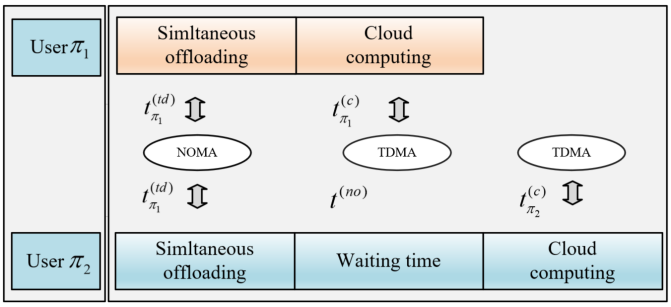

Fig. 2: Flexible time-sharing NOMA scheme for computing task offloading.

As shown in Fig. 2, utilizing $\pi_k \triangleq \{\pi_1, \pi_2\}$ to represent the task offloading sequence for two users. Within this sequencing scheme, users π_1 and π_2 simultaneously transmit data over the allocated period, and it should be noted that the proposed flexible time-sharing NOMA framework collapses to traditional NOMA and TDMA setups. We posit that the data size of the computational outcomes for both users is minimal (for instance, in applications such as facial recognition, object detection in video streams, etc.), rendering the feedback delay

from the results inconsequential. Consequently, based on the aforementioned assumptions, the computation offloading latency for each user is given by

$$\begin{aligned} T_{\pi_1} &= t_{\pi_1}^{(td)} + t_{\pi_1}^{(c)}, \\ T_{\pi_2} &= t_{\pi_1}^{(td)} + t^{(no)} + t_{\pi_2}^{(c)}, \end{aligned} \tag{2}$$

where $t_{\pi_k}^{(td)}, k = \{1, 2\}$ signifies the latency required for user k to offload the task, $t^{(no)}$ signifies the waiting time of π_2 on the MEC server, $t_{\pi_k}^{(c)}, k = \{1, 2\}$ denotes the MEC computing delays for user k. It is crucial to note that the time overhead for the ground user to offload the computational task to the MEC is given by $T_{\pi_k} = \frac{D_{\pi_k}(t)}{r_{\pi_k}(t)}$. The MEC server takes $T_{\pi_k}^C(t) = \frac{F_{\pi_k}(t)}{f^C(t)}$ to process the computational task in bits, where $f^C(t)$ represents the number of CPU cycles required to compute 1 bit of data.

For each discrete transmission by user π_k, it is straightforward to identify the optimal RIS reflection that maximizes its attainable rate. Consequently, the upload delay for a task to the edge server is contingent upon the task's size and the transfer rate allocated to the end user. The transmission rate for the k-th user is given by

$$R_k = \beta_k B \log_2(1 + \frac{p_{\pi_k}|h_r \Theta G + h_d|^2}{\beta_k B N_0}), \tag{3}$$

where β_k represents the uplink bandwidth factor for the k-th user, B signifies the total uplink bandwidth, N_0 indicates the power spectral density of additive Gaussian noise, p_{π_k} denotes the transmission power of user k. For NOMA-based transmissions, as the effective channel gain for the user is dictated by the RIS's passive reflection, there are $2! = 2$ potential decoding orders at the AP for the users. For instance, if π_1 is decoded prior to π_2, then the achievable rates for the two users are, respectively, given by

$$\begin{aligned} R_1 &= B \log_2(1 + \frac{p_{\pi_1}|h_r \Theta G + h_d|^2}{p_{\pi_2}|h_r \Theta G + h_d|^2 + B N_0}), \\ R_2 &= B \log_2(1 + \frac{p_{\pi_2}|h_r \Theta G + h_d|^2}{B N_0}). \end{aligned} \tag{4}$$

Similarly, if π_2 is decoded first, the approach is consistent with the prior case.

C. Problem Formulation

Under the constraint of the maximum transmit power of the user, the delay minimization issue within the RIS-assisted dual-user NOMA-MEC network can be formulated as

$$\begin{aligned} \mathbf{P:} \quad & \min_{H, \Theta, p_{\pi_k}} \max\{T_{\pi_1}, T_{\pi_2}\}, \\ \mathbf{s.t.} \quad & t_{\pi_1}^{(td)} r_{\pi_1}^{(td)} + t^{(no)} r_{\pi_1}^{(no)} = L_{\pi_1}, \\ & t_{\pi_2}^{(td)} r_{\pi_2}^{(td)} + t^{(no)} r_{\pi_2}^{(no)} = L_{\pi_2}, \\ & t_{\pi_1}^{(td)} \geq 0, t_{\pi_2}^{(td)} \geq 0, t^{(no)} \geq 0, \\ & 0 \leq p_{\pi_1} \leq P_{\pi_1}, 0 \leq p_{\pi_2} \leq P_{\pi_2}, \\ & 0 < \theta_n < 2\pi, \forall n \in \{1, 2, ..., N\}, \end{aligned} \tag{5}$$

where $\mathbf{H} = \{G, \mathrm{h}_{r,k}, \mathrm{h}_{d,k}\}$, $\mathbf{p} = \{p_{\pi_1}, p_{\pi_2}\}$, L_{π_k} denotes the task quantity for π_k, and the non-convexity and constraints of problem render a globally optimal solution infeasible. Conventional convex optimization methods struggle with such intricate non-convex challenges. Hence, we address this problem utilizing DRL, which is elaborated subsequently.

III. Methodology

To address this issue, we formulate the optimization problem as a Markov Decision Process (MDP), subsequently addressing it through DRL which encompasses five principal elements: the environment, the agent, the state space, the action space and the reward function [12]. Fundamentally, an agent learns decision-making by engaging with the environment. The agent's objective is to maximize the aggregate reward accumulated over time. This process involves the agent perceiving the environment's state, selecting actions in accordance with a policy, and receiving rewards or penalties that direct its learning trajectory. This framework is commonly represented as an MDP, wherein the environment's subsequent state is contingent solely on the present state and the action taken, rather than the sequence of actions that occurred previously.

In the context of a Deep Q-Network (DQN), a commonly-used DRL aglorithm, the Q-function, denoted as $Q(s, a|\theta)$, is utilized to estimate the expected utility of taking action a in state s under policy π, which can be defined as

$$Q_{\pi'}(s, a; \theta) = \mathbb{E}_{\pi'}\{G_t | s_t = s, a_t = a\}, \tag{6}$$

where the expectation $\mathbb{E}$ encapsulates the anticipated cumulative reward, represented as $G_t = \sum_{k=0}^{\infty} \lambda^k r_{t+k+1}$, λ is a discount factor within the interval $(0, 1]$. The parameters θ pertain to the deep neural network (DNN) that approximates the Q-function, with θ signifying the network's weights and biases. The training data, comprising tuples $\{s_t, a_t, r_t, s_{t+1}\}$, are extracted from a replay buffer, which serves as a reservoir of past experiences, facilitating the stochastic sampling of batches for the training process.

Although DQNs have been successful in handling tasks with discrete action spaces, there are still some limitations: i) DQNs are primarily designed for environments with discrete action spaces. They struggle with continuous action spaces where the number of possible actions is large or infinite. ii) DQNs can suffer from overestimation bias because the same network is used to select and evaluate actions. iii) Balancing exploration and exploitation is challenging in DQNs, as random actions may be suboptimal and lead to inefficient learning.

To address these limitations, we propose a Deep Deterministic Policy Gradient (DDPG)-based latency reduction (DDPR) algorithm. Specifically, DDPG is an Actor-Critic algorithm that is suitable for environments with continuous action spaces. It uses two neural networks: an Actor network that selects actions, and a Critic network that evaluates the actions. The Critic network is similar to the DQN, estimating the Q-value of state-action pairs. The Actor network, however, is deterministic and outputs actions directly based on the current state. DDPG uses the deterministic policy gradient theorem to

update the Actor network, allowing it to learn a policy without the need for a separate exploration strategy. This makes DDPG more efficient in environments where actions are continuous and the state space is high-dimensional. Additionally, DDPG employs techniques like target networks and replay buffers to stabilize training and improve sample efficiency. The Q-value in the main Critic network $Q(s,a|\theta^Q)$ and the main Actor network $\mu(s|\theta^\mu)$ can be, respectively, expressed as

$$\begin{aligned} Q(s,a) &= \mathbb{E}[r_1^\gamma | S_1 = s, A_1 = a; \pi'], \\ \mu(s) &= \mathbb{E}[r_1^\gamma | S_1 = s; \pi']. \end{aligned} \tag{7}$$

A. MDP Description

The main components in an MDP, namely, the state space, action space and the reward, are elaborated below.

- **State space $\mathcal{S}$:** At time step t, the state is defined as $s_t = [H, p_{\pi_1}, p_{\pi_2}]$. The state space $\mathcal{S}$ encompasses the sum of all $s_t, t = 1, ..., T$. The agent perceives the system's state by interacting with the environment. It is worth mentioning that, given that DNNs process real numbers rather than complex numbers, the construction of state s necessitates the dissection of complex-number components into their real and imaginary constituents, which are then presented as separate inputs to the network in this context.
- **Action space $\mathcal{A}$:** At time step t, the agent employs the state s_t as input to adjust the phase shift induced by the RIS. Upon completion of the update, a novel phase shift is achieved. The action is constituted by an H matrix and a phase shift matrix Θ, where $H = \mathrm{Re}\{H\} + \mathrm{Im}\{H\}$, $\Theta = \mathrm{Re}\{\Theta\} + \mathrm{Im}\{\Theta\}$. Consequently, the action space of the system comprises these matrices, with actions being delineated within a continuous action space. Similarly, the action space $\mathcal{A}$ encompasses the sum of all $a_t, t = 1, ..., T$.
- **Reward r:** The performance of a_t should be measured by a scalar reward r_t to help the agent make better decisions, and in step t of DRL, specifically in the given state space and action space, if the scheme can minimize problem p, while satisfying the constraints, then a_t should be rewarded. At time step t, considering the constraints, the internal reward is defined, and the reward is determined as the sum rate capacity. Meanwhile, in the simulation experiment, the average reward is calculated as $r_T = \frac{1}{T}\sum_t^T r(t)$.

B. Proposed DDPR Algorithm

To solve this problem, a significant challenge stems from the continuous nature of both the state and action spaces, which could be tackled by DDPG. As illustrated in Fig. 3, the DDPG architecture comprises two DNNs: the Actor network and the Critic network. The Actor network ingests the state as input and generates a continuous action, which then gets forwarded to the Critic network alongside the state. This network configuration is employed to approximate the action value, effectively bypassing the requirement to identify the action

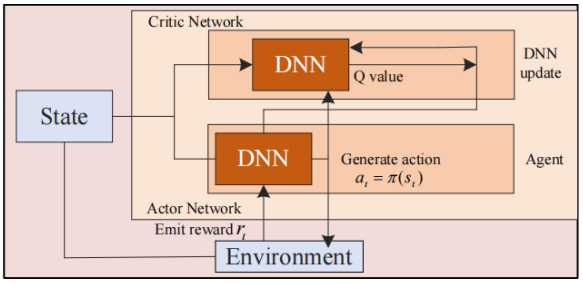

Fig. 3: Architecture of the proposed DDPR algorithm.

that optimizes the value function for the subsequent state, a task that would otherwise involve non-convex optimization [11]. The subsequent updates for the trained Critic network and the trained Actor network are detailed as follows

$$\begin{aligned} &\theta_c^{(t+1)} = \theta_c^{(t)} - \mu_c \Delta_{\theta_c^{(train)}} \ell(\theta_c^{(train)}), \\ &\theta_a^{(t+1)} \\ =&\theta_a^{(t)} - \mu_a \Delta_a q(\theta_c^{(t\arg et)} | s^{(t)}, a) \Delta_{\theta_a^{(train)}} \pi'(\theta_a^{(train)} | s^{(t)}), \end{aligned} \tag{8}$$

where μ_c and μ_a denote the learning rate for training the Critic network and the Actor network, respectively, $\pi'(\theta_a^{(train)}|s^{(t)}$ denotes the parameters of the trained Actor network given $s^{(t)}$, $\Delta_a q(\theta_c^{(t\arg et)}|s^{(t)}, a)$ is the gradient of the target Critic network with respect to the action, $\Delta_{\theta_a^{(train)}}\pi'(\theta_a^{(train)}|s^{(t)})$ is the gradient $\theta_a^{(train)}$ of the trained Actor network.

The target Critic network influences the training Actor network's updates by leveraging the gradient of the reward function with respect to the selected action. This mechanism ensures that subsequent action selection aligns with the desired action direction, thereby enhancing the value function. The subsequent updates for the target Critic and Actor networks are detailed as follows

$$\begin{aligned} \theta_c^{(\text{target})} &\leftarrow \tau_c \theta_c^{(\text{train})} + (1 - \tau_c)\, \theta_c^{(\text{target})}, \\ \theta_a^{(\text{target})} &\leftarrow \tau_a \theta_a^{(\text{train})} + (1 - \tau_a)\, \theta_a^{(\text{target})}, \end{aligned} \tag{9}$$

where τ_c and τ_a are the updated learning of the target Critic network and the target Actor network, respectively.

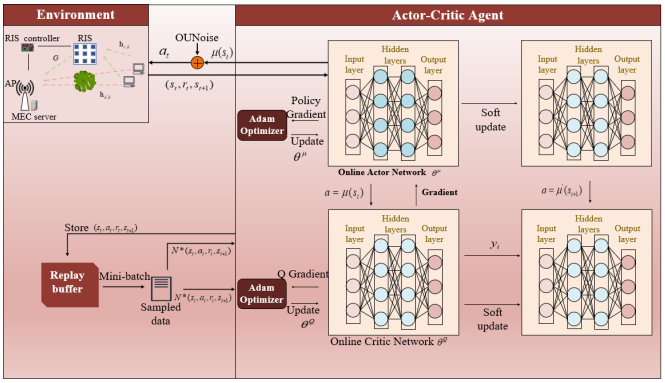

Fig. 4: Learning graph of the proposed DDPR-based algorithm

As depicted in Fig. 4, a concise overview of the DDPR-based learning graph is presented. In DDPR, the inclusion

of a nonlinear activation function aims to mitigate the issue of vanishing or exploding gradients during neural network training. The ReLU function, a variant of the standard ReLU [8], is employed for this purpose. While ReLU is known for its effectiveness in addressing gradient-related issues, it offers the added benefit of accelerated learning when dealing with sparse features. The specifics of the proposed algorithm are encapsulated in Algorithm 1.

Algorithm 1: Algorithm 1: Deep Deterministic Policy Gradient-based latency reduction (DDPR) algorithm

Input: T, τ_a, τ_c, N_0
Output: a $= \{H, \Theta\}$
Initialize: Experience replay memory $\mathcal{M}$ with size $\mathcal{D}$, $Q(s, a|\theta^Q)$ and $\mu(s|\theta^\mu)$ with weights θ^Q and θ^μ, $\theta_a^{(\text{train})} = \theta_a^{(\text{target})}$, $\theta_c^{(\text{train})} = \theta_c^{(\text{target})}$
for episode = 0,1,2..., $\mathcal{N}$-1 **do**
  Collect and preprocess state s_t for the n^{th} episode to obtain the first state $s^{(0)}$
  for t=0, 1, 2, $\cdots$, T-1 **do**
    Obtain action $s^{(t+1)} = \{H^{(t)}, \Theta^{(t)}\} = \pi'(\theta_a^{(train)})$
    Observe new state $s^{(t+1)}$ given action $a^{(t)}$
    Observe instant reward $r^{(t+1)}$,
    Store the experience $(s^{(t)}, a^{(t)}, r^{(t+1)}, s^{(t+1)})$ in the replay memory
    Independent exploration reward Update the training Critic network $\theta_c^{(train)}$
    Update the training Actor network $\theta_a^{(train)}$
    Update the target Critic network $\theta_c^{(target)}$ after every U steps
    Update the target Actor network $\theta_a^{(target)}$ after every U steps
  end for
  Update the target networks using the soft update algorithm.
end for

IV. SIMULATION RESULTS

In this section, we utilize PyTorch to assess the efficacy of the proposed solution. We presume an edge server with a default CPU clock frequency of 50×10^9 cycles per second. The total uplink bandwidth is configured to $B = 3$ MHz, with a default standard deviation of 0.1 MHz. The initial transmit power for users is set at 30dB, ensuring robust signal reception for dual users. Furthermore, we consider the default number of reflected beamforming elements to be $N = 24$. The simulation outcomes were computed by averaging data from a minimum of 10,000 distinct trials.

In Fig. 5, during the simultaneous offloading phase, NOMA demonstrates a slight advantage over TDMA in achieving the minimization of total delay. Under constraints of limited cloud capacity, the proposed flexible time-sharing NOMA scheme exhibits a comparable sum delay to the NOMA benchmark scheme, which is lower than that of the TDMA benchmark scheme, for the MEC communication system with RIS deployment. Here, the baseline scheme TDMA is defined as the exclusive use of TDMA communication within the MEC process. Furthermore, given that the NOMA priority could be positive, the proposed scheme can be simplified to a pure NOMA approach. At the same time, we also consider the

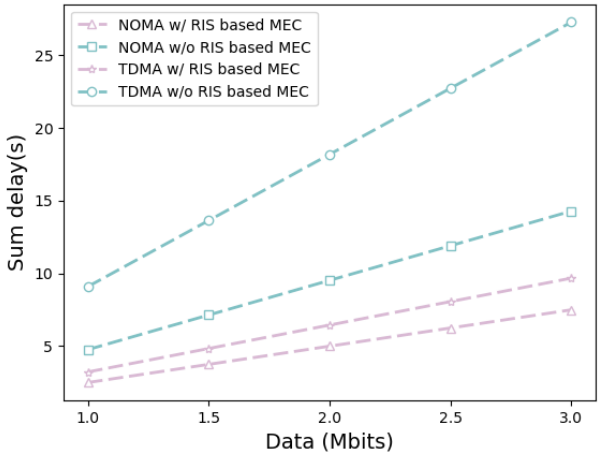

Fig. 5: Performance comparison of the proposed NOMA and standard TDMA schemes with flexible time-sharing.

performance pros and cons of the two schemes in the case of no IRS deployment. Additionally, the task amount for the user is set as $[1, 1.5, 2, 2.5, 3]$, which more intuitively highlights the superiority of the NOMA computation offloading scheme under flexible time-sharing.

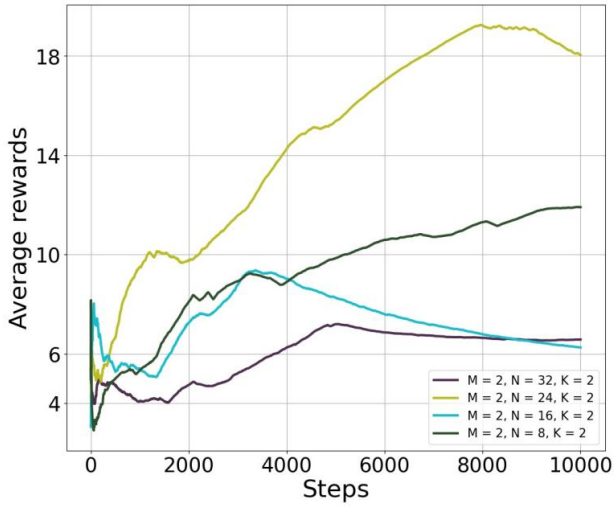

Fig. 6: Comparison of average rewards across time steps for various system parameter configurations.

As illustrated in Fig. 6, we further elucidate the correlation between the average reward and the quantity of RIS elements. Set AP to have M antennas to communicate with the user. In accordance with the transmission rate stipulated by our reward function, the rate value in the proposed scheme progressively ascends with the augmentation of RIS elements. However, the optimal convergence is achieved when the RIS element count is 24, and the system configuration with $N = \{8, 16, 24, 32\}$ attains the most favorable training outcome. The reward is plotted against the time step. In comparison to transmit power, the DDPR algorithm exhibits greater resilience to fluctuations in system parameters. Specifically, as the element

count N escalates, the average reward incrementally rises as anticipated, yet this does not prolong the convergence period of the DDPR method. As depicted in Fig. 7, we employ the

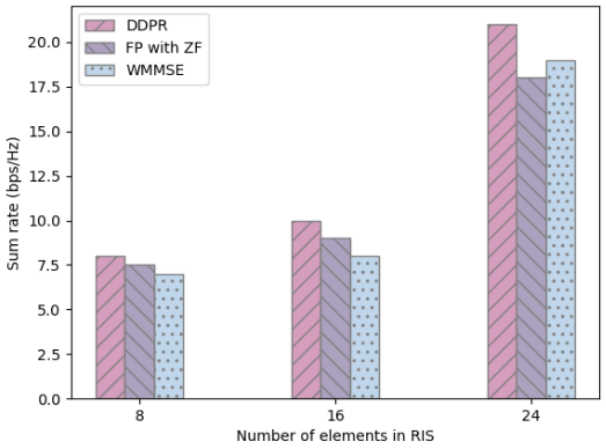

Fig. 7: Performance comparison of sum rate under different schemes, where WMMSE stands for Weighted Minimum Mean Square Error (WMMSE) scheme, and FP with ZF stands for Fractional Programming (FP) with Zero-Forcing (ZF) beamforming scheme.

Weighted Minimum Mean Square Error (WMMSE) algorithm [13] and an iterative algorithm grounded in Fractional Programming (FP) with Zero-Forcing (ZF) beamforming [14]. In their standard configurations, both methodologies necessitate comprehensive and current cross-cell Channel State Information (CSI). They are inherently centralized and iterative. These algorithms serve as benchmark solutions against which our proposed DDPR approach is evaluated. The comparative analysis reveals that our algorithm marginally outperforms these benchmarks, significantly enhancing the transmission rate and thereby facilitating the reduction of latency.

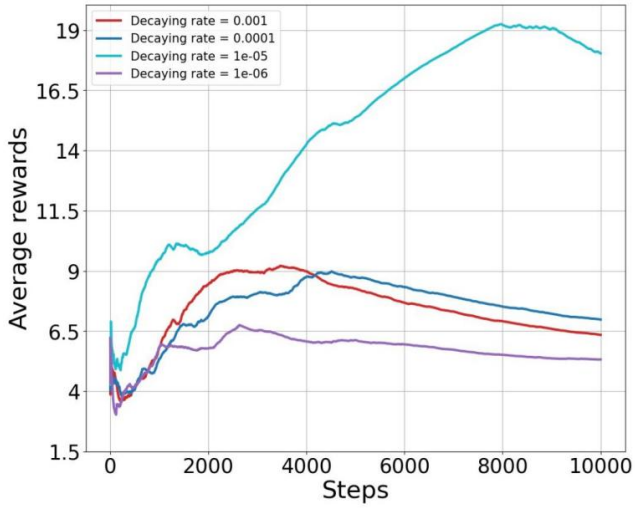

Fig. 8: Average reward trend analysis with different settings.

Fig. 8 illustrates the mean time step of reward pairs across various learning rates. The results indicate that the performance is notably superior when the learning rate is set at 0.00001. This is attributed to the fact that an excessively high learning rate tends to amplify oscillations, thereby significantly degrading the performance.

V. CONCLUSION

This work delves into the problem of minimizing delay in a RIS-assisted MEC framework under the flexible time-division NOMA scheme. We introduce a DDPG-based Latency Reduction (DDPR) algorithm as a solution to this challenge. The simulation outcomes demonstrate that our proposed algorithm outperforms existing methodologies with respect to both latency and transmission rate. Looking ahead, our future research endeavors will encompass integrating reliability into the task data offloading process within RIS-assisted edge computing systems.

REFERENCES

[1] Y. Wang, Z. Yang, J. Cui, P. Xu, G. Chen, T. Q. S. Quek, and R. Tafazolli, "Optimizing the Fairness of STAR-RIS and NOMA Assisted Integrated Sensing and Communication Systems," in *IEEE Trans. Wireless Commun.*, vol. 23, no. 6, pp. 5895-5907, June 2024.

[2] Y. Sun *et al.*, "RIS-Assisted Robust Hybrid Beamforming Against Simultaneous Jamming and Eavesdropping Attacks," in *IEEE Trans. Wireless Commun.*, vol. 21, no. 11, pp. 9212-9231, Nov. 2022.

[3] S. Guo *et al.*, "Differential Reflecting Modulation for Reconfigurable Intelligent Surface-Based Communications," in *IEEE Commun. Lett.*, vol. 25, no. 3, pp. 907-910, March 2021.

[4] C. Feng, H. H. Yang *et al.*, "Hybrid Learning: When Centralized Learning Meets Federated Learning in the Mobile Edge Computing Systems," in in *IEEE Trans. Commun.*, vol. 71, no. 12, pp. 7008-7022, Dec. 2023.

[5] C. Feng, H. H. Yang *et al.*, "Mobility-Aware Cluster Federated Learning in Hierarchical Wireless Networks," in *IEEE Trans. Wireless Commun.*, vol. 21, no. 10, pp. 8441-8458, Oct. 2022.

[6] X. Ma *et al.*, "Joint Beamforming and Reflecting Design in Reconfigurable Intelligent Surface-Aided Multi-User Communication Systems," in *IEEE Tran. Wireless Commun.*, vol. 20, no. 5, pp. 3269-3283, May 2021.

[7] S. Guo, S. Lv, H. Zhang, J. Ye and P. Zhang, "Reflecting Modulation," in *IEEE J. Sel. Areas Commun.*, vol. 38, no. 11, pp. 2548-2561, Nov. 2020.

[8] X. Yu, D. Xu and R. Schober, "Optimal Beamforming for MISO Communications via Intelligent Reflecting Surfaces," in *IEEE Int. Workshop Signal Process. Advances Wireless Commun. (SPAWC)*, Atlanta, GA, USA, 2020, pp. 1-5,

[9] Y. Han, W. Tang, S. Jin, C.-K. Wen, and X. Ma, "Large intelligent surface-assisted wireless communication exploiting statistical CSI," in *IEEE Trans. Veh. Technol.*, vol. 68, no. 8, pp. 8238–8242, 2019.

[10] H. Lu *et al.*, "Aerial Intelligent Reflecting Surface: Joint Placement and Passive Beamforming Design With 3D Beam Flattening," in *IEEE Trans. Wireless Commun.*, vol. 20, no. 7, pp. 4128–4143, Jul. 2021.

[11] Z. Mlika, S. Cherkaoui, Network slicing with MEC and deep reinforcement learning for the internet of vehicles, 2022, http://dx.doi.org/10.48550/arXiv. 2201.11295.

[12] V. Mnih, K. Kavukcuoglu, D. Silver et al., "Human-level control through deep reinforcement learning," Nature, vol. 518, no. 7540, pp. 529–533, 2015.

[13] H. Sun, X. Chen, Q. Shi, M. Hong, X. Fu, and N. D. Sidiropoulos, "Learning to optimize: Training deep neural networks for interference management," in *IEEE Trans. Signal Process.*, vol. 66, no. 20, pp. 5438-5453, Oct. 2018.

[14] C. Huang, A. Zappone, G. C. Alexandropoulos, M. Debbah and C. Yuen, "Reconfigurable intelligent surfaces for energy efficiency in wireless communication," in *IEEE Trans. Wireless Commun.*, vol. 18, no. 8, pp. 4157-4170, Aug. 2019.