Latency-aware task offloading in UAV-assisted edge computing: Leveraging deep reinforcement learning

Nguyen Tri Hai[1*]

[1]Seoul National University of Science and Technology, Republic of Korea

[*]Corresponding author: haint93@seoultech.ac.kr

ARTICLE INFO

DOI:10.46223/HCMCOUJS.tech.en.13.2.2910.2023

Received: August 17th, 2023

Revised: September 06th, 2023

Accepted: September 07th, 2023

Keywords:

deep reinforcement learning; latency; optimization; task offloading; unmanned aerial vehicle

ABSTRACT

Due to the exponential expansion of the Internet of Things (IoT) technology, the increasing amount of data generated by numerous IoT devices has become a significant challenge for communication and computing networks. On the other hand, the concept of Mobile Edge Computing (MEC) presents a viable solution as it brings computing, storage, and networking capabilities in close proximity to the user, enabling the hosting of applications that require significant computation power and low latency right at the network's edge. In addition, the maneuverability, cost-effectiveness, and ease of deployment make Unmanned Aerial Vehicles (UAVs) highly versatile wireless platforms. Capitalizing on the benefits of MEC and UAVs, this paper proposes a UAV-assisted MEC system that offloads services to reduce the computation burden in IoT. The objective is to maximize the task completion rate, considering factors such as user transmit power, task offloading rate, and UAV trajectory variables. To tackle this optimization problem, this paper devises a method based on the Deep Deterministic Policy Gradient (DDPG), an algorithm for continuous action spaces in deep reinforcement learning. The numerical simulations show the effectiveness of the proposed approach in comparison to other benchmarks, i.e., deep Q-network (DQN) and full offloading methods. In a simulation, the proposed DDPG method can achieve a fully optimized system with a 100% task success rate, while the DQN-based method achieves a lower task success rate of 95.93% and the full offloading method only obtains a task success rate of 82.73%.

1. Introduction

The advent of 5G and beyond networks has been made possible by advancements in networking technologies and novel computing-communication frameworks (Liu et al., 2017). Among these technologies, Mobile Edge Computing (MEC) is crucial in enhancing network performance by facilitating distributed computation (Huda & Moh, 2022; Wang, Zhang, Liu, Long, & Nallanathan, 2022). MEC brings computational power and energy resources to the network edge, allowing decentralized computation capabilities. Despite its numerous advantages, the deployment of MEC servers is limited by the positioning of stationary towers, making it challenging to establish them at any location or time. Moreover, the infrastructure can be vulnerable to natural disasters, posing a risk to MEC functionality. Additionally, installing infrastructure in remote areas like hotspots and mountains presents significant difficulties. Consequently, IoT devices may struggle to adequately serve their users in such scenarios.

MEC servers on Unmanned Aerial Vehicles (UAVs) offer vital support to communication networks due to their versatility and quick deployment capabilities (Dao et al., 2021; Huda & Moh, 2022). This support becomes crucial in situations where mobile users in hotspot areas or emergency situations need temporary offloading. By augmenting MEC servers with extra computational power, UAV-aided MEC expedites computations and enhances the battery life of mobile devices. In addition, artificial intelligence, particularly Deep Reinforcement Learning (DRL), is utilized in conjunction with MEC to enable intelligent decision-making (Cheng et al., 2019; Nguyen & Park, 2022; Wang et al., 2022). A notable algorithm utilized in this context is the deep Q-network (DQN) (Mnih et al., 2015), which integrates Deep Neural Networks (DNNs) with reinforcement learning to autonomously tackle complex decision-making issues. Despite its ability to resolve problems including high-dimensional state spaces, DQN models still face challenges with continuous action spaces. To overcome this limitation, the Deep Deterministic Policy Gradient (DDPG) algorithm was introduced (Lillicrap et al., 2016). DDPG is an off-policy actor-critic model-free method capable of learning policies in continuous action spaces. The actor-critic approach is composed of a policy function, which acts as an actor and generates actions, and a Q-value function, which acts as a critic and evaluates the performance of the actor, guiding its subsequent actions.

This paper incorporates the benefits of MEC, UAV, and DRL. Firstly, a system model is presented for a UAV-MEC that serves users in a communication network. Secondly, the problem of task offloading is formulated with the goal of maximizing the completion of tasks. The problem considers the UAV trajectory, offloading ratio, and transmit power. To address the non-convex nature of the issue, it is transformed into a Markov Decision Process (MDP), and the DDPG algorithm is employed. Lastly, the efficiency of the proposed DDPG-based algorithm is illustrated through simulations.

The structure of this article is as follows: Section 2 provides an introduction to the system model and problem formulation. Section 3 outlines the DDPG-based solution. Section 4 showcases the simulation results. Finally, Section 5 concludes the study.

2. System model and problem formulation

2.1. System model

A network environment including K User Devices (UEs) and an MEC server positioned on a UAV (UAV-MEC) is examined, as illustrated in Figure 1 (Yang, Bi, & Zhan, 2021). The UAV-MEC serves the UEs in computation offloading. With the rise in compute-intensive and time-sensitive services, UEs face limitations in computing resources, battery life, and other constraints. Consequently, UEs must wirelessly offload a portion of their tasks to the UAV-MEC server for processing. The time is divided into T intervals. The set of time slots is represented by $\mathcal{T} = \{1,2,\dots,T\}$. It is assumed that the UAV has enough energy to complete the flight and wireless communications within a flight period of T. At $t \in \mathcal{T}$, the position of UE k is defined by $\mathbf{q}_k(t) = (x_k(t), y_k(t)), \forall k \in \mathcal{K}$, where $\mathcal{K} = \{1,2,\dots,K\}$ is the set of UEs. The UAV is positioned at a fixed height of H_U and its horizontal position is given by $\mathbf{q}_U(t) = (x_U(t), y_U(t))$. The current velocity of the UAV is defined as $v_U(t) \leq v_{max}$, where v_{max} is the maximum velocity that the UAV can travel. Hence, $\mathbf{q}_U(t+1) = \mathbf{q}_U(t) + v_U(t) \times t$. In addition, the distance $d_k(t)$ between UE k and the UAV-MEC is given by:

$$d_k(t) = \sqrt{(x_U(t) - x_k(t))^2 + (y_U(t) - y_k(t))^2 + H_U^2} \tag{1}$$

During time slot t, the UE k has a computational task, denoted by $Task_k(t) = (w_k, c_k, t_k)$, where w_k, c_k, t_k refer to the data size (bits), the required computational resources (CPU cycles/bit), and the maximum acceptable latency, respectively. The UE has the option to execute its task independently or transfer it to the UAV-MEC for parallel processing. Hence, an offloading rate variable $o_k(t) \in [0,1]$ is defined, representing the percentage of the task transferring to the UAV for computation. The remaining percentage of the task $1 - o_k(t)$ is processed at the UE k.

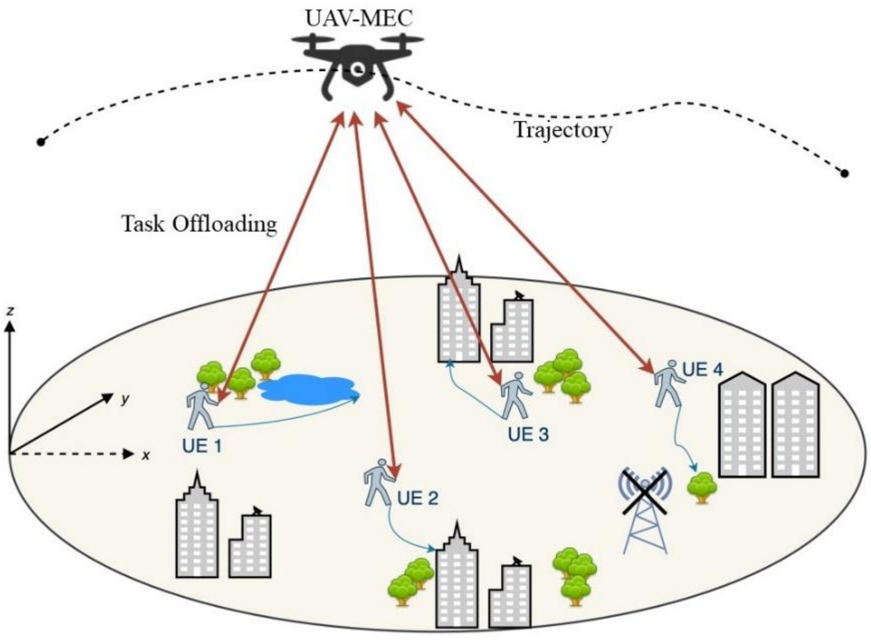

Figure 1. System model

Source: Yang et al. (2021)

Communication model: Based on previous studies (Nguyen et al., 2022; Truong, Dao, & Cho, 2022) it is assumed that the communication links between the UAV and UEs are primarily wireless with Line-of-Sight (LoS) characteristics. In this scenario, the channel quality is solely determined by the distance of communication.

Non-orthogonal multiple access (NOMA) is utilized in the transmission links between UEs and the UAV, enabling the UAV to efficiently serve multiple UEs (Liu et al., 2017). The UAV can obtain perfect Channel State Information (CSI). With this information, the UAV decodes each signal sequentially using the Successive Interference Cancellation (SIC) method, selecting the decoding order of UEs based on their channel gain with the UAV. At timestamp t, the channel gain g_k of UE k is estimated by:

$$g_k(t) = g_0 d_k^{-2}(t) \tag{2}$$

where g_0 is the channel gain at a unit distance of 1m. Based on the Shannon formula, the transmission rate of UE k is expressed:

$$r_k(t) = W \, log_2\left(1 + \frac{p_k(t) g_k(t)}{\sum_{j=k+1}^{K} p_j(t) g_j(t) + \sigma^2}\right) \tag{3}$$

where W is the communication bandwidth, $p_k(t)$ denotes the transmit power of UE k, and σ^2 represents the environment noise.

Computation model: The delay is measured for the tasks of all UEs. The communication and computation models in the study neglect the size of the results transmitted from the UAV to a UE, as it is considerably smaller compared to the offloaded data (Sun, Ni, & Wang, 2021; Wang, Fang, Ding, & Xiong, 2021; Zhu et al., 2021).

For the task $Task_k(t)$, the UE k utilizes the allocated bandwidth to offload a portion $o_k(t)$ of its computational data to the UAV. Consequently, for each UE k, the delay for the UAV to process this task includes the duration of offloading and the duration of execution as:

$$T_k^{offload}(t) = \frac{o_k(t)w_k(t)}{r_k(t)} + \frac{o_k(t)w_k(t)c_k(t)}{F_k(t)} \tag{4}$$

where $F_k(t)$ represents the allocated computational resources of the UAV-MEC for the UE k at timestamp t, assuming that UAV-MEC computational resources F_U are allocated evenly for all UEs. Additionally, the local processing time of the $1 - o_k(t)$ part of the task at the UE k is calculated by:

$$T_k^{local}(t) = \frac{(1-o_k(t))w_k(t)c_k(t)}{f_k} \tag{5}$$

where f_k (CPU cycles/s) represents the computational power of the UE k. According to the partial offloading method, the latenct for executing the task is defined by the higher value between local execution and offloading execution (Nguyen & Park, 2023; Truong et al., 2022). Hence, the delay of the UE k for processing $Task_k(t)$ is given by:

$$T_k(t) = max\{T_k^{local}(t), T_k^{offload}(t)\} \tag{6}$$

2.2. Problem formulation

The objective is to maximize the task completion rate in a UAV-assisted edge network by jointly optimizing transmit power, offloading rate, and UAV trajectory in all time slots. Consequently, the optimization issue can be expressed by:

$$\max_{p_k(t), o_k(t), \mathbf{q}_U(t), \forall k} \sum_{k \in \mathcal{K}} H\big(t_k(t) - T_k(t)\big)$$

$$\begin{aligned}
&\text{subject to:}\\
&\text{(C1)}: o_k(t) \in [0,1], \forall k, t\\
&\text{(C2)}: 0 \le p_k(t) \le P_{max}, \forall k, t\\
&\text{(C3)}: 0 \le x_U(t) \le x_{max}, \forall t\\
&\text{(C4)}: 0 \le y_U(t) \le y_{max}, \forall t\\
&\text{(C5)}: v_U(t) \le v_{max}, \forall t
\end{aligned} \tag{7}$$

where $H(\cdot)$ is a unit step function that outputs 1 if the input value is greater than zero, and 0 otherwise. Constraint (C1) defines the range of values for the offloading rate, while constraint (C2) specifies the range of values for UE transmit power. Constraints (C3), (C4), and (C5) guarantee that the UAV operates within the designated region of interest at a limited speed. However, due to the vast number of real-time observations generated in the dynamic environment, traditional methods may struggle to solve the problem effectively. Therefore, the DDPG algorithm is introduced to address this challenge.

3. Deep reinforcement learning-based approach

3.1. Markov decision process

MDP utilizes a mathematical approach to represent the decision-making process of a dynamic system. This study considers the UAV as the agent and the communication network as the environment. The agent's decision-making process relies on observations and learning from its interaction with the environment. The MDP is presented as a collection of state, action, and reward elements, defined in the following.

- The state in timestamp t is defined by $s(t) = \{\mathbf{q}_U(t), \mathbf{q}_k(t), Task_k(t)\}$, which includes the coordinate of UAV, coordinate of UEs, and the tasks to be computed.
- The action in timestamp t is defined by $a(t) = \{v_U(t), p_k(t), o_k(t)\}$, which includes the current speed of the UAV, the transmit powers of UEs, and the offloading rates of the tasks.
- The reward in timestamp t is defined by $r(t) = \sum_{k\in\mathcal{K}} H\big(t_k(t) - T_k(t)\big)$, which is the objective of the optimization problem.

3.2. DDPG-based approach

The DDPG algorithm is employed to address the defined MDP model. DDPG is a technique that utilizes four DNNs, namely actor network $\mu(\cdot\,|\theta_\mu)$, critic network $Q(\cdot\,|\theta_Q)$, target actor network $\mu'(\cdot\,|\theta_{\mu'})$, and target critic network $Q'(\cdot\,|\theta_{Q'})$. The actor network $\mu(\cdot\,|\theta_\mu)$ is responsible for mapping the state and action, while the critic network $Q(\cdot\,|\theta_Q)$ estimates the Q value using state-action pairs. The algorithm consists of observation process and learning process. During the observation phase, the agent interacts with the environment, and the experience replay records tuples of the form$(s(t), a(t), s(t+1), r(t))$. During the learning phase, a small batch of tuples is randomly sampled from the experience replay buffer. The actor network and critic network play crucial roles as they serve as the primary objects for learning and updating in DDPG. The framework of the DDPG-based algorithm is depicted in Figure 2.

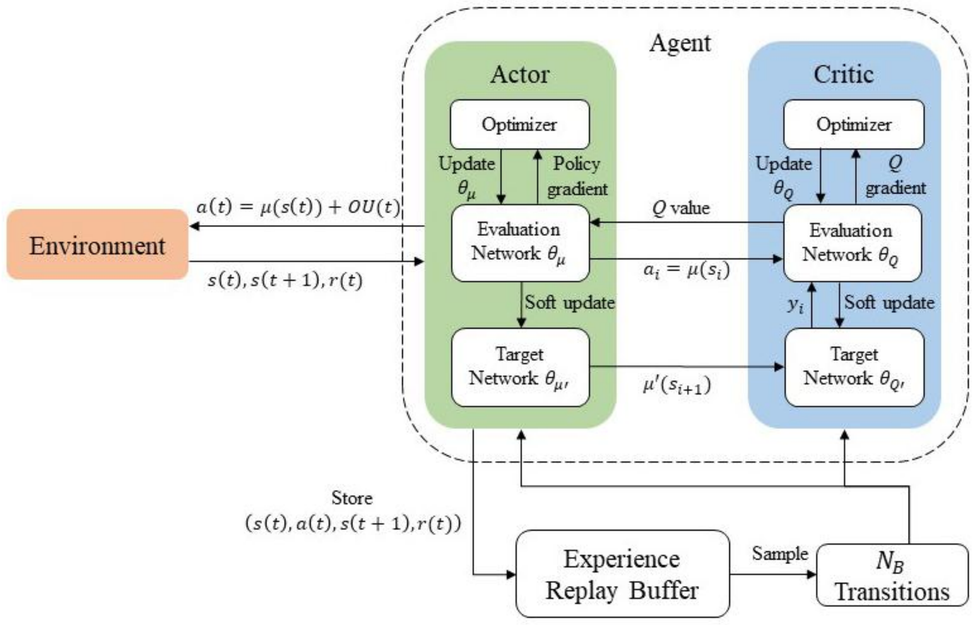

Figure 2. DDPG framework

The action at timestamp t is obtained by incorporating a noise into the actor policy for the observation process, as $a(t) = \mu\big(s(t)\big|\theta_\mu\big) + OU(t)$, where $OU(t)$ is the Ornstein-Uhlenbeck noise. The actor network $\mu(\cdot\,|\theta_\mu)$ is modified by maximizing the total expected reward. Once the action $a(t) = \mu(s(t)|\theta_\mu)$ is determined from the actor network, the critic network calculates the evaluated Q-value $Q(s(t), a(t)|\theta_Q)$. The actor network's policy gradient is then updated via:

$$\nabla_{\theta_\mu} J = \frac{1}{N_B}\sum_i \nabla_a Q(s, a|\theta_Q)|_{s=s_i, a=\mu(s_i)} \nabla_{\theta_\mu}\mu(s|\theta_\mu)|_{s_i} \tag{8}$$

where N_B represents the size of the mini-batch taken from the experience buffer. The critic network $Q(\cdot\,|\theta_Q)$ is updated by minimizing the discrepancy between the evaluated and target values. The action $a(t+1)$ is obtained by referring to the target network given the state $s(t+1)$ as $a(t+1) = \mu'(s(t+1)|\theta_{\mu'})$. Then, the target value of $Q'(\cdot\,|\theta_{Q'})$ is computed by:

$$y = r(t) + \gamma Q'(s(t+1), a(t+1)|\theta_{Q'}) \tag{9}$$

where $\gamma \in [0, 1]$ is the discount factor. The evaluated Q-value $Q(s(t), a(t)|\theta_Q)$ is estimated in the critic network. The loss is expressed by:

$$L = \frac{1}{N_B}\sum_i \Big(y_i - Q\big(s_i, a_i|\theta_Q\big)\Big)^2 \tag{10}$$

The parameters of the critic network are modified through the utilization of gradient descent, resulting in the reduction of the loss function. The parameters of the target networks are updated through:

$$\begin{aligned} \theta_{Q'} &= \tau\theta_Q + (1-\tau)\theta_{Q'} \\ \theta_{\mu'} &= \tau\theta_\mu + (1-\tau)\theta_{\mu'} \end{aligned} \tag{11}$$

where $\tau \in (0, 1)$ is the soft update rate. In the end, the DDPG returns an optimal policy or well-trained actor network, which is used in the execution phase.

4. Performance evaluation

4.1. Simulation settings

In the simulations, K UEs are distributed in a 200m × 200m area randomly. A UAV-MEC serves the UEs in the considered area with the starting coordinate of (0, 0, 50) and the maximum velocity of $v_{max} = 15$ m/s. The computing power of all UEs is set as 1GHz while the computing power of the UAV-MEC is set as 10GHz. The task size for the UEs varies from 0.5 to 1.0 Megabits (Mb), and the required computational resources to compute one bit of the task data are randomly distributed between 900 to 1200CPU cycles/bit. Additionally, the maximum allowable delay is set at 0.5s. The maximum UE transmit power, noise power, bandwidth, and channel gain of unit distance 1m are set as $P_{max} = 20\text{dBm}$, $\sigma^2 = -170\,\text{dBm/Hz}$, $B = 1\,\text{MHz}$, $g_0 = 1.42e^{-4}$, respectively. The operation time is 300s. The number of training episodes is set to 1,000. DNNs include two hidden layers of 256 neurons each and use ReLU as an activation function. The learning rate is set to $1e^{-4}$, the discount factor γ is 0.99, the mini-batch size is 32, the experience replay buffer size is $1e^{-6}$, and the soft update rate is $1e^{-2}$. The simulations are implemented by the Spyder IDE 5.4.3 with Python programming language 3.10.9 and PyTorch 1.12.1. Furthermore, two benchmark schemes are used for comparison purposes to evaluate the performance of the proposed DDPG-based algorithm (DDPG) as follows.

- DQN-based offloading computation algorithm (DQN) (Mnih et al., 2015; Nguyen & Park, 2023): DQN is a well-known DRL algorithm. The action space is discretized into multiple discrete levels, which can be applied to the DQN algorithm to solve the problem.
- Full offloading computation (Full Offloading) (Nguyen & Park, 2023; Truong et al., 2022): All tasks are offloaded to UAV-MEC.

To evaluate the performance of the methods, the task success rate metric is used. The task success rate represents the percentage or ratio of successfully executed tasks out of the total tasks, ranging from 0 to 100%. A higher task success rate indicates better performance and successful completion of the offloaded tasks.

4.2. Simulation results

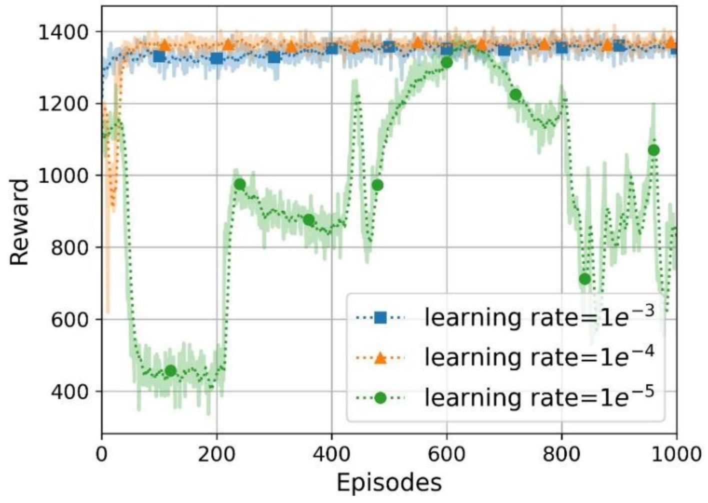

Figure 3. The DDPG-based algorithm's convergence under different learning rates

To begin with, the convergence behavior of the proposed DDPG-based algorithm is assessed. One of the most influential hyperparameters in this algorithm is the learning rate, which profoundly impacts convergence behavior. As 5 UEs are considered, Figure 3 illustrates the convergence patterns for various learning rates in the proposed algorithm. Notably, if the learning rate is too small, i.e., $1e^{-5}$, it makes very tiny updates to the weights in networks, leading to the slow training process and low performance. If the learning rate is set too high, it can cause undesirable divergent behavior in the loss function. As a result, the learning rate of $1e^{-4}$ outperforms others by attaining the highest reward and achieving the fastest convergence. Consequently, the learning rate of $1e^{-4}$ is designated as the default choice for subsequent evaluations.

In the following, the influence of the number of UEs in the network on offloading performance is examined. The range of UEs considered varied from 5 to 10 UEs while keeping other parameters at their default values. The results presented in Figure 4 indicate the performance of three methods with different numbers of UEs. Due to the limited computing power of UAV-MEC, the performance of the system declines as the number of UEs increases. This decline is particularly significant in the case of Full Offloading, where the task success rate drops to 82.73% and 5.48% for 5 UEs and 10 UEs, respectively. A similar trend can be observed with DDPG and DQN, but they still manage to achieve satisfactory results, with task completion rates of 74.63% and 51.20% in the scenario with 10 UEs, respectively.

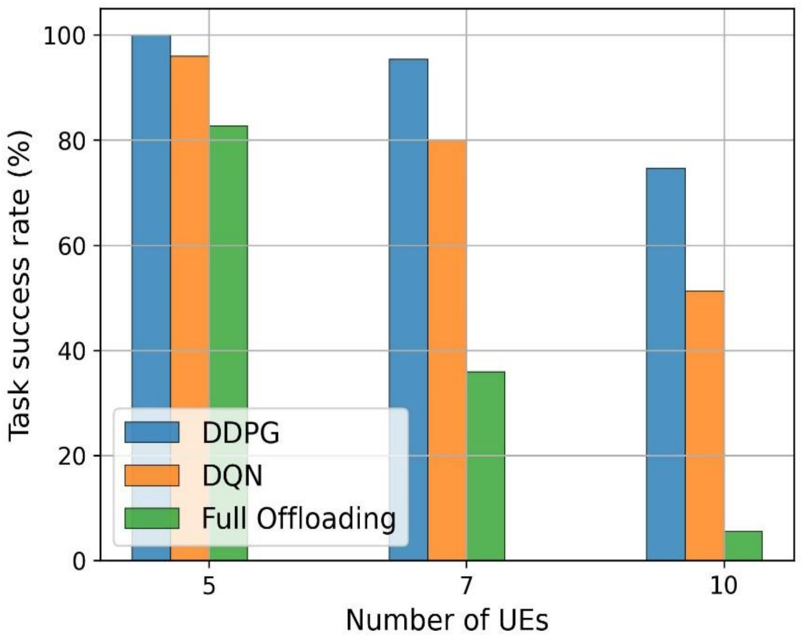

Figure 4. Task success rate versus the number of UEs

Next, the efficiency of offloading methods affected by the maximum delay is analyzed. The range of the maximum delay is set from 0.3 to 0.5s, while the number of UEs is set at 5, and all other parameters are kept at their default values. The results can be observed in Figure 5. As the maximum allowable delay becomes stricter, the performance decreases. For example, when the maximum allowable delay is set at 0.3s, Full Offloading can only complete 15.80% of the tasks, while DDPG and DQN complete 55.40% and 36.00% respectively. However, as the allowable task delay increases to 0.4s, DDPG improves significantly and can process 92.47% of the tasks. Furthermore, at a task delay of 0.5s, DDPG achieves a 100% completion rate. The performance of Full Offloading and DQN also improves as the task delay increases, but still remains lower than that of DDPG.

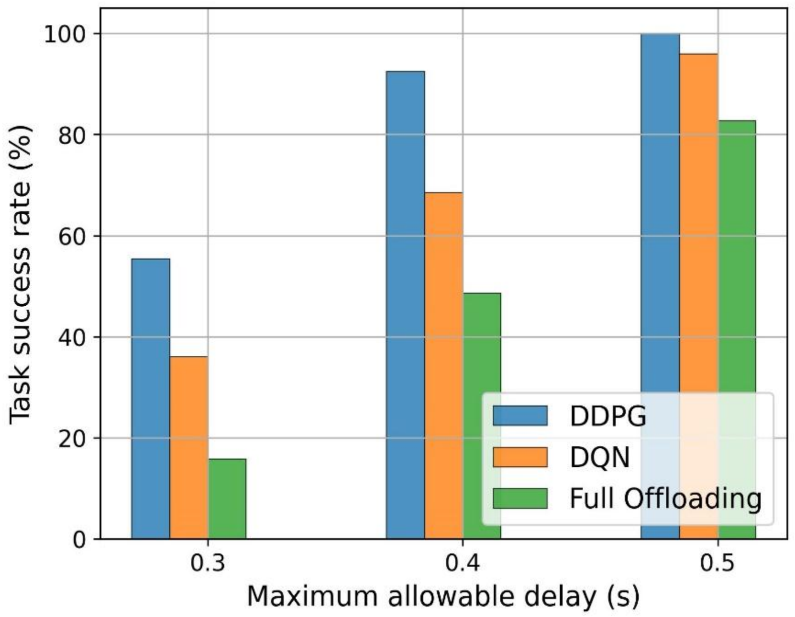

Figure 5. Task success rate versus the maximum allowable delay of the tasks

Finally, the impact of the computational power of the UAV-MEC on the effectiveness of the methods is evaluted. The computing power of the UAV-MEC ranges from 8 to 12GHz, while the number of UEs is fixed at 5, and all other parameters are set to their default values. As shown in Figure 6, the task success rate increases with the computing power of the UAV-MEC. Full Offloading heavily relies on the computing capacity of the edge server, resulting in a significant difference in task success rates as the computing capacity increases. Even at the lowest computing capacity of 8 GHz, DDPG and DQN demonstrate good performance with task success rates of 92.13% and 84.67% respectively. However, with the same scenario, Full Offloading only achieves a 50.60% task success rate. When the UAV-MEC computing power increases to 10GHz, DDPG achieves a fully optimized system with a 100% task success rate, while DQN achieves a lower task success rate of 95.93% and Full Offloading only obtains a task success rate of 82.73%. With a high level of computing capacity for the UAV (i.e., 12GHz), both DDPG and DQN algorithms achieve the maximum task success rate of 100%. However, Full Offloading falls slightly short of the optimal rate, achieving a task success rate of 96.20%.

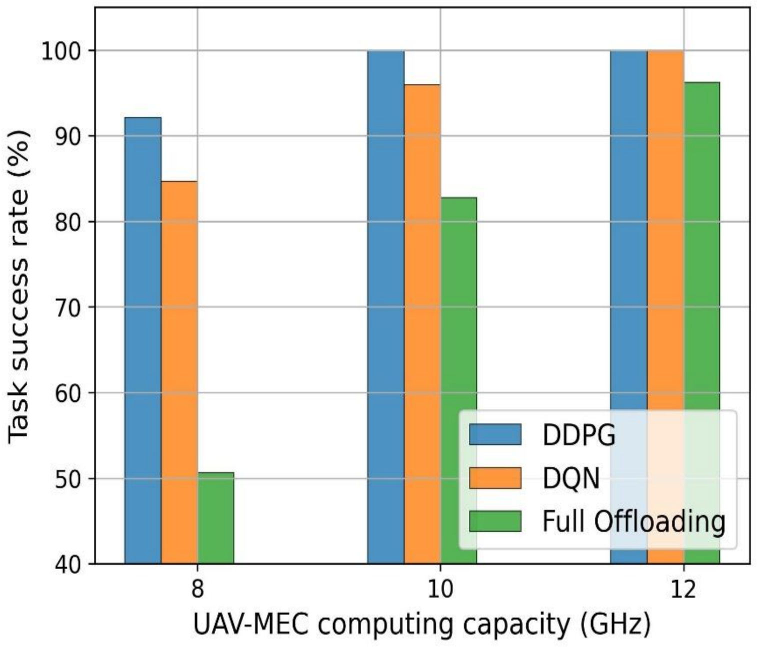

Figure 6. Task success rate versus the UAV-MEC computing capacity

In general, throughout the simulations, the DDPG-based algorithm with the continuous action space shows its efficiency under different scenarios.

5. Conclusions

This research examined how the advanced 5G and beyond network could be utilized to improve computation offloading in a MEC system. In this system, a UAV equipped with an edge server offers computational and communication services to user devices. The main objective is to maximize the number of completed tasks by considering both resource allocation optimization and UAV maneuverability. To achieve this, a non-convex objective function was formulated to address the computation offloading problem. To identify the most effective offloading approach, this paper proposed a DDPG-based algorithm and assessed its performance through simulations. The results showed that the suggested DDPG-based algorithm outperforms the baseline algorithms.

References

Cheng, N., Lyu, F., Quan, W., Zhou, C., He, H., Shi, W., & Shen, X. (2019). Space/aerial-assisted computing offloading for IoT applications: A learning-based approach. *IEEE Journal on Selected Areas in Communications, 37*(5), 1117-1129. doi:10.1109/JSAC.2019.2906789

Dao, N. N., Pham, V. Q., Ngo, T. H., Tran, T. T., Vo, B. N. Q., Lakew, D. S., & Cho, S. (2021). Survey on aerial radio access networks: Toward a comprehensive 6G access infrastructure. *IEEE Communications Surveys & Tutorials, 23*(2), 1193-1225. doi:10.1109/COMST.2021.3059644

Huda, S. A., & Moh, S. (2022). Survey on computation offloading in UAV-Enabled mobile edge computing. *Journal of Network and Computer Applications, 201,* Article 103341. doi:10.1016/j.jnca.2022.103341

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., ... Wierstra, D. (2016). *Continuous control with deep reinforcement learning.* Paper presented at the 4th International Conference on Learning Representations (ICLR 2016), San Juan, Puerto Rico.

Liu, Y., Qin, Z., Elkashlan, M., Ding, Z., Nallanathan, A., & Hanzo, L. (2017). Nonorthogonal multiple access for 5G and beyond. *Proceedings of the IEEE, 105*(12), 2347-2381. doi:10.1109/JPROC.2017.2768666

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*, *518*(7540), 529-533. doi:10.1038/nature14236

Nguyen, H. T., & Park, L. (2022). *A survey on deep reinforcement learning-driven task offloading in aerial access networks.* Paper presented at the 13th International Conference on Information and Communication Technology Convergence (ICTC 2022), Jeju Island, Republic of Korea. doi:10.1109/ICTC55196.2022.9952687

Nguyen, H. T., & Park, L. (2023). HAP-Assisted RSMA-enabled vehicular edge computing: A DRL-based optimization framework. *Mathematics*, *11*(10), Article 2376. doi:10.3390/math11102376

Nguyen, H. T., Truong, P. T., Dao, N. N., Na, W., Park, H., & Park, L. (2022). *Deep reinforcement learning-based partial task offloading in high altitude platform-aided vehicular networks*. Paper presented at the 13th International Conference on Information and Communication Technology Convergence (ICTC 2022), Jeju Island, Republic of Korea. doi:10.1109/ICTC55196.2022.9952890

Sun, C., Ni, W., & Wang, X. (2021). Joint computation offloading and trajectory planning for UAV-assisted edge computing. *IEEE Transactions on Wireless Communications, 20*(8), 5343-5358. doi:10.1109/TWC.2021.3067163

Truong, P. T., Dao, N. N., & Cho, S. (2022). HAMEC-RSMA: Enhanced aerial computing systems with rate splitting multiple access. *IEEE Access, 10,* 52398-52409. doi:10.1109/ACCESS.2022.3173125

Wang, H., Zhang, H., Liu, X., Long, K., & Nallanathan, A. (2022). Joint UAV placement optimization, resource allocation, and computation offloading for THz band: A DRL approach. *IEEE Transactions on Wireless Communications*, *22*(7), 4890-4900. doi:10.1109/TWC.2022.3230407

Wang, Y., Fang, W., Ding, Y., & Xiong, N. (2021). Computation offloading optimization for UAV-assisted mobile edge computing: A deep deterministic policy gradient approach. *Wireless Networks*, *27*(4), 2991-3006. doi:10.1007/s11276-021-02632-z

Yang, Z., Bi, S., & Zhang, Y. J. A. (2021). *Dynamic trajectory and offloading control of UAV-enabled MEC under user mobility*. Paper presented at the 2021 IEEE International Conference on Communications Workshops (ICC Workshops 2021), Montreal, QC, Canada. doi:10.1109/ICCWorkshops50388.2021.9473504

Zhu, S., Gui, L., Zhao, D., Cheng, N., Zhang, Q., & Lang, X. (2021). Learning-based computation offloading approaches in UAVs-assisted edge computing. *IEEE Transactions on Vehicular Technology, 70*(1), 928-944. doi:10.1109/TVT.2020.3048938