IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Traitement du Signal
Vol. 41, No. 6, December, 2024, pp. 3075-3082
Journal homepage: http://iieta.org/journals/ts

Signal Processing Mechanisms for Cloud-Edge-Terminal Collaborative MEC Networks

Anqi Pang, Yingjun Lv*

Jinan Campus Swinburne College, Shandong University of Science and Technology, Jinan 250031, China

Corresponding Author Email: sdkdlyj@163.com

https://doi.org/10.18280/ts.410624

Received: 8 June 2024
Revised: 12 October 2024
Accepted: 29 October 2024
Available online: 31 December 2024

Keywords:
mobile edge computing, cloud-edge-terminal collaboration network, computation offloading, deep reinforcement learning

ABSTRACT

This paper investigates the application of Intelligent Reflecting Surfaces (RIS) in 5G wireless communications, focusing on enhancing the performance of Internet of Things (IoT) networks. The rapid growth of IoT applications has exacerbated challenges in traditional wireless communication, including signal interference and limited bandwidth. To overcome these issues, we propose an innovative optimization approach utilizing deep learning algorithms to dynamically adjust the configuration of RIS. This approach significantly improves signal quality and network throughput. Our experimental results demonstrate the effectiveness of the proposed solution in enhancing communication performance under various real-world scenarios. The findings provide valuable insights into the practical deployment of RIS in 5G IoT networks, offering a promising solution to address key performance bottlenecks.

1. INTRODUCTION

The European Telecommunications Standards Institute (ETSI) has introduced Multi-Access Edge Computing (MEC) to address the challenges of rapidly evolving mobile and wireless networks. MEC extends cloud computing (CC) capabilities to the edge of the mobile network, overcoming the limitations of traditional cloud infrastructure. It also serves as a supplement to corporate data and processing centers by providing computing, storage, networking, and data analysis resources closer to the data source. The advent of 5G technology, with its high data rates (up to 10 GB/s), ultra-low latency (below 1 ms), ultra-reliable performance (99.99999%), energy efficiency (90% reduction), and support for up to 300,000 devices per cell, is a key driver for MEC's development. To meet these requirements, migrating services closer to users is essential, which MEC enables [1].

Computing offloading, a core MEC technology, integrates internet and wireless communication networks by shifting task processing and data storage from centralized cloud centers to the user side. This enhances local data processing, storage, and environmental awareness capabilities. Mobile edge computing (MEC) platforms, due to their proximity to users, offer lower latency and higher service rates compared to other computing platforms. Additionally, MEC platforms have smaller data centers, making them easier to deploy than mobile access control platforms [2, 3].

Resource allocation is vital for the development of emerging technologies, especially in the evolving power network. As the power grid becomes increasingly complex, mobile edge computing is being applied to the electric power Internet of Things (IoT) to improve network reliability and reduce delays [4]. This technology, when integrated into power systems, can enhance the operational efficiency and security of mining power grids, especially in remote areas where traditional communication infrastructure is difficult to deploy [5, 6].

In summary, as the digitalization of MEC networks accelerates and terminal access density increases, traditional optical fiber communication in power grids faces limitations, including high costs and poor scalability. This hampers the development of smart grids. However, 5G wireless technology can help address these challenges. Despite the substantial delays in mining power grids, MEC's low-latency and high-reliability characteristics offer promising solutions. This paper explores the potential applications of MEC in mining power grid systems, where cloud, edge, and terminal collaborative computing not only supports remote cloud services but also provides local, low-latency, and reliable solutions to improve grid management, interaction, and electrical safety.

2. RELATED WORK

The current integration and development of MEC and the power IoT is largely dependent on the large-scale deployment of edge nodes. The current automation and intelligent construction of power networks is accompanied by a corresponding growth in the data generated by power grid-related applications and services. In light of the low-delay, localization, and intelligence characteristics of edge computing, one must contemplate the construction of a power IoT system utilizing MEC technology. The real-time transmission and processing information, while ensuring reliability, has become a current research hotspot [7-9].

In recent years, many studies have focused on real-time information processing in smart grids. For instance, Khan et al. [10] proposed a task allocation strategy based on task priority

to optimize business offloading scheduling, demonstrating that the algorithm significantly enhances system revenue through simulations. The challenge of rapidly analyzing real-time power data from smart devices in smart grids is also critical. The work [11] addressed this by designing a greedy algorithm to minimize task offloading costs, validating its superiority through simulation. To meet the reliability and delay requirements of smart grids, Shi et al. [12] designed a system architecture for 4G and 5G mobile networks based on MEC infrastructure. Additionally, the authors of studied [13] an artificial intelligence center based on MEC, analyzing typical AI applications in smart grids from various aspects. In study [14], the authors proposed an effective information transmission strategy and a charging information dissemination algorithm for MEC servers and electric vehicles, optimizing communication in smart grids. Given the low-latency, high-reliability requirements and limited computing capabilities of power terminals, study [15] introduced a joint optimization strategy for task and resource allocation, which was validated through simulation in resource-limited environments. Despite these advances, challenges remain in securing communication data in the power grid, improving response speed, and applying joint optimization strategies for task offloading and transmission in mining power grids.

In the field of smart grids, real-time information processing and resource management are crucial for ensuring system stability and efficient operation [16]. However, these processes face challenges such as latency, limited computational resources, and security concerns. This is particularly true in specialized environments like mining areas, where traditional centralized cloud computing solutions, due to high transmission latency, no longer meet real-time requirements [17]. Mobile Edge Computing (MEC) reduces latency by moving data processing closer to the network edge, but it introduces new challenges related to resource allocation and task offloading. Additionally, data transmission in smart grids involves numerous terminals and devices, making it a significant challenge to ensure data security while maintaining high performance. This paper addresses these issues by proposing a joint optimization model aimed at reducing task completion time through optimized task offloading and resource allocation strategies, while ensuring the security of data transmission. This provides an efficient solution for smart grid applications in specialized environments, such as mining power grids.

3. COMPUTING TASK OFFLOADING MODEL

3.1 Task offloading architecture

This article considers the potential of utilizing the 'cloud-edge-end' MEC structure, as depicted in Figure 1, for the transmission and calculation of equipment tasks. The network consists of N substation base stations, each serving K_n mining devices in its area. Consequently, the network accommodates a total of $K^{total} = \sum_{n=1}^{N} K_n$ users. Given that each base station has an MEC server, due to the limited computing power of local equipment in the mining area, the computational tasks of the device may be transferred to the corresponding base station in order to minimize processing delays. Each base station is capable of transferring a portion of its computational workload to the cloud through wired fronthaul line transmission. Figure 1 shows the entire process of task transmission and processing of MEC networks [18].

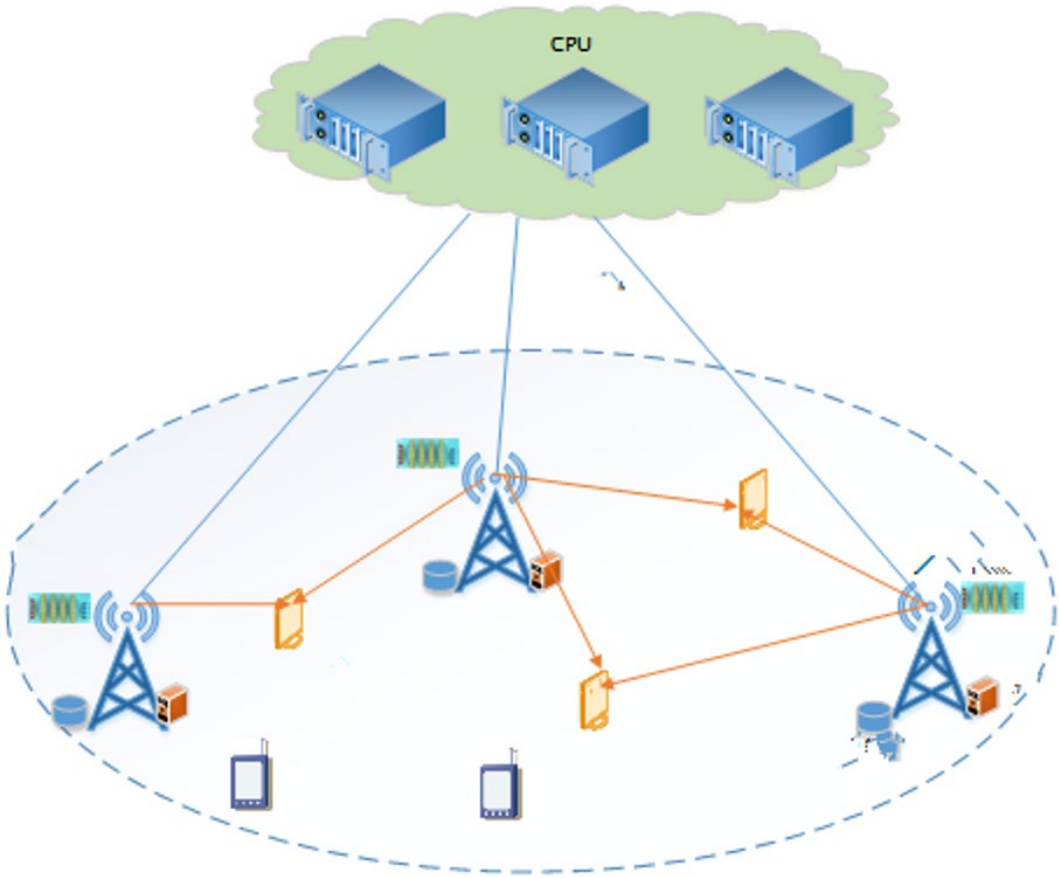

Figure 1. Architecture of cloud-edge-terminal collaborative MEC networks

3.2 Communication model

In this system, each area has a task to process in each time slot. These tasks can be computed by local power terminal devices, MEC, or a server in the cloud. Considering the long distance between the mining equipment and the cloud, it is necessary to use the base station as a relay to offload tasks to the cloud.

In this scenario, the equipment terminals in the mining area and the substation base station transmit and receive signals simultaneously. Each base station is equipped with multiple antennas. Let B denote the bandwidth, Let $h_{n,k}$ denote the channel gain from device k to base station n, $p_{n,k}^{u}$ denote the transmission power. During this time frame, the rate at which data is sent from the device to the base station in the uplink direction is determined by the following calculation [19]:

$$R_{n,k}^{u} = B\log_2(1+\frac{h_{n,k}p_{n,k}^{u}}{BN_0}) \tag{1}$$

where, N_0 is the noise power spectral density. It is worth noting that in the cloud-edge-terminal coordinated comprehensive power network in mining areas, base stations are considered to be isolated from each other. Therefore, there will be no interference between base stations. At the same time, the rate at which data is uploaded from base station n to the cloud center during this interval can be denoted as R_n^b.

3.3 Computational model

3.3.1 Task offloading model

Within the MEC network, every device possesses a computational task that requires transfer and execution [20]. For a device k associated with a base station n, the computing task is denoted $Q_{n,k} = (D_{n,k}, C_{n,k})$, where $D_{n,k}$ is the size of the task input data (in bits), $C_{n,k}$ is the number of CPU cycles required to compute 1 bit of input data. Then the total CPU cycles required by task $Q_{n,k}$ is $C_{n,k}^{total} = D_{n,k}C_{n,k}$. For base station n, the task set is expressed as $Q_n^b = \{Q_{n,1}, Q_{n,2}, \ldots, Q_{n,K_n}\}$. Computing tasks are divisible, on this basis a partial offloading model can be adopted.

3.3.2 Computing model

For a given task $Q_{n,k}$, the structure of the offloading model is described in the following manner:

1) Local computing: Define $\alpha_{n,k} \in [0,1]$ denote the fraction of data handled by the local devices within the mining area. Subsequently, the volume of data handled by the device and the necessary CPU cycles are denoted as $\alpha_{n,k}D_{n,k}$ and $\alpha_{n,k}D_{n,k}C_{n,k}$ respectively. The computing power of the device is expressed in $f_{n,k}^{u}$ (CPU cycles/second). If $\alpha_{n,k} = 1$, this indicates that task $Q_{n,k}$ is entirely handled by the device itself, with no transfer to the base station. Conversely, if $\alpha_{n,k}$ is less than 1, a portion of the computational tasks will be transferred to the corresponding base station n, with $(1-\alpha_{n,k})D_{n,k}$ representing the data volume sent to the base station.

2) Edge computing: Define $\beta_{n,k} \in [0,1]$ as the fraction of data handled by the base station within the mining region [21]. Then, the amount of data processed under base station n and the required CPU cycles are $\beta_{n,k}D_{n,k}$ and $\beta_{n,k}D_{n,k}C_{n,k}$ respectively. The total computing capability of each base station is represented by F^b (unit is CPU cycle/second), and computing resources must be distributed among the tasks within Q_n^b. Let $f_{n,k}^{b}$ be the MEC computing capability allocated to task $Q_{n,k}$, and require $\sum_{K=1}^{K_n} f_{n,k}^{b} \le F^b$, if when $\alpha_{n,k} + \beta_{n,k} = 1$, this implies that task $Q_{n,k}$ is entirely handled by the device and base station. If $\alpha_{n,k} + \beta_{n,k} < 1$, then a portion of the computational tasks will be transferred to the cloud, with the data volume sent to the cloud being $(1-\alpha_{n,k}-\beta_{n,k})D_{n,k}$.

3) Cloud computing: $\kappa_{n,k} \in [0,1]$ is used to represent the proportion of data generated in the mining area that is offloaded to the cloud, and $\alpha_{n,k} + \beta_{n,k} + \kappa_{n,k} = 1$. In addition, the execution time on the cloud is negligible [22].

3.4 Delay analysis

3.4.1 Local computing delay

For each task $Q_{n,k}$, the equipment in the area must handle a complete set of data $\alpha_{n,k}D_{n,k}$. The delay associated with local computation is represented by:

$$t_{n,k}^{u,comp} = \frac{\alpha_{n,k}D_{n,k}C_{n,k}}{f_{n,k}^{u}} \tag{2}$$

The overall delay required to handle data processed locally is:

$$t_{n,k}^{u} = t_{n,k}^{u,comp} \tag{3}$$

3.4.2 Edge computing delay

For every task $Q_{n,k}$, the MEC at the base station handles an aggregate of $\beta_{n,k}D_{n,k}$. The overall delay in processing edge computing data encompasses the time taken for communication from the device to the base station, as well as the computation delay occurring at the base station. Consequently, the mean transmission delay for task $Q_{n,k}$, when offloaded from the device k to base station n, can be calculated as follows:

$$t_{n,k}^{u,tran} = \frac{(1-\alpha_{n,k})D_{n,k}}{R_{n,k}^{u}} \tag{4}$$

As with many current studies, the delay of sending back the computation results from the base station is negligible. Once offloaded to the base station, the delay in processing task $Q_{n,k}$ can be determined by:

$$t_{n,k}^{b,comp} = \frac{\beta_{n,k}D_{n,k}C_{n,k}}{f_{n,k}^{b}} \tag{5}$$

Accordingly, the overall time required to handle edge computing data is:

$$t_{n,k}^{b} = t_{n,k}^{u,tran} + t_{n,k}^{b,comp} \tag{6}$$

3.4.3 Cloud computing delay

For every task $Q_{n,k}$, the cloud processes a cumulative $\kappa_{n,k}D_{n,k}$ bits of data. The overall delay in handling cloud computing data encompasses the time taken for communication from the device to the base station, as well as from the base station to the cloud center. Likewise, the time

delay for transmitting task $Q_{n,k}$ from base station n to the cloud center can be represented by:

$$t_{n,k}^{b,tran} = \frac{(1-\alpha_{n,k}-\beta_{n,k})D_{n,k}}{R_n^b} \tag{7}$$

Subsequently, the overall delay required for handling data in cloud computing is:

$$t_{n,k}^{c} = t_{n,k}^{u,tran} + t_{n,k}^{b,tran} \tag{8}$$

3.4.4 Data encryption delay

The substations are all connected through the central platform information system and the computer rooms of each mining area. The tele-control signals, video surveillance data, and office network data of the company's substations are not isolated from other business flows of the central platform information system, and cannot be dedicated to private networks, posing potential data security risks. Therefore, in this study, we propose a traditional Advanced Encryption Standard encryption scheme that is more effective in terms of security and performance [23].

Let $z_{n,k} \in \{0,1\}$ denote the security choice for each device. Specifically, when $z_{n,k} = 1$, the data associated with the computing task is encrypted before being sent to the MEC server. Conversely, if $z_{n,k} = 0$, the data is offloaded to the MEC server without encryption. Security choices are determined by the privacy needs of the data being transmitted and are manually decided by the mobile device. Consequently, this paper will randomly assign security decisions in the simulation section.

Lastly, considering the security layer, the extra time required for processing computing tasks remotely on the MEC server can be calculated separately:

$$t_{n,k}^{enc+dec} = z_{n,k}\left(\frac{\omega(1-\alpha_{n,k})D_{n,k}}{f_{n,k}^{u}} + \frac{\omega\beta_{n,k}D_{n,k}}{f_{n,k}^{b}}\right) \tag{9}$$

where, ω represents the number of CPU cycles required for encryption and decryption of each bit of data in the computing task [24].

Based on the above, for each task $Q_{n,k}$, the choice to offload it to the base station or the cloud is determined by the task division ratio $\alpha_{n,k}$, $\beta_{n,k}$, $\kappa_{n,k}$. The overall delay for task $Q_{n,k}$ is represented as:

$$t_{n,k}^{total} = \begin{cases} \max\{t_{n,k}^{u,comp}, t_{n,k}^{u,tran} \\ +t_{n,k}^{b,comp} + t_{n,k}^{enc+dec}\}, \kappa_{n,k} = 0 \\ \max\{t_{n,k}^{u,comp}, t_{n,k}^{u,tran} + t_{n,k}^{b,comp} \\ +t_{n,k}^{enc+dec}, t_{n,k}^{u,tran} + t_{n,k}^{b,tran}\}, \kappa_{n,k} > 0 \end{cases} \tag{10}$$

3.5 Problem formulation

This research seeks to minimize the total delay across all mining power equipment. The optimization challenge is structured as follows:

$$\min_{\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\kappa},f^b} \sum_{n=1}^{N}\sum_{k=1}^{K_n} t_{n,k}^{total} \tag{11a}$$

$$\text{s.t. } \alpha_{n,k}, \beta_{n,k}, \kappa_{n,k} \in [0,1], \forall n,k, \tag{11b}$$

$$\alpha_{n,k} + \beta_{n,k} + \kappa_{n,k} = 1, \forall n,k, \tag{11c}$$

$$z_{n,k} \in \{0,1\}, \forall n,k, \tag{11d}$$

$$\sum_{k=1}^{K_n} f_{n,k}^{b} = F^b, f_{n,k}^{b} \geq 0, \forall n \tag{11e}$$

Among them, the optimization variables are the task division rates $\alpha = [\alpha_{1,1},\ldots,\alpha_{N,K_N}]$, $\beta = [\beta_{1,1},\ldots,\beta_{N,K_N}]$, $\kappa = [\kappa_{1,1},\ldots,\kappa_{N,K_N}]$, and the allocation of computing capability $f^b = [f_{1,1}^b,\ldots,f_{N,K_N}^b]$, In addition, (11b) and (11c) assign constraints to the task division of each computing task, (11d) is the secure encryption decision constraint for each computing task, (11e) is the total computing capacity constraint of the MEC in each mining area base station. Obviously, to achieve the least delay across all tasks, the best strategy is to make full use of the computing resources available at each base station. Thus, we incorporate equality constraints in (11e) rather than using inequality constraints.

The optimization challenge presented in (11a) is non-convex and cannot be tackled directly because of the objective function's discontinuity and non-convex nature. To address this intricate and non-convex optimization issue, we further propose a cloud-edge-terminal collaborative computing strategy based on deep reinforcement learning to optimize the delay performance of the network.

4. DEEP DETERMINISTIC POLICY GRADIENT ALGORITHM

In the context of modelling and optimization, the computing offloading strategy and computing resource allocation of the MEC system are solely contingent upon its current state, and are therefore unrelated to its past state. This is a typical Markov decision problem. The problem can be solved using the Deep Deterministic Policy Gradient (DDPG) algorithm. The DDPG algorithm mainly consists of actor and critic network structures. Each network is updated according to its own update rules to maximize the cumulative expected revenue. Among them, the state space $\mathcal{S} = \{\mathcal{Q}_n^b,\ldots,\mathcal{Q}_N^b\}$, the action space $\mathcal{A} = \{\alpha,\beta,\kappa,f^b\}$, and reward function $\mathcal{R}(\mathcal{S},\mathcal{A}) = -\sum_{n=1}^{N}\sum_{k=1}^{K_n} t_{n,k}^{total}$. Previous analysis has proven that if only a single Q neural network algorithm is used, the learning process is very unstable because the parameters of the Q network are frequently updated with gradients and are used to calculate the gradients of the Q network and the policy network. Based on this, DDPG creates two neural network copies for the policy network and Q network respectively, called target networks. One benefit is that the target network's parameters undergo minimal changes, which aids in computing the gradient of the current network during training. This stability makes convergence easier. However, a drawback is that the small parameter adjustments can slow down the learning process. Initially, the actor chooses an action a_t based on its behavior policy and sends it to the environment for execution. In the next step, the environment performs action a_t, then provides feedback in the form of rewards and updates the state. Finally, the actor logs this state transition into the experience replay buffer, which is used as a dataset for training

the current network. The fourth step is to randomly sample multiple transformation data from the experience pool as training data. The fifth step is to calculate the gradient of the current Q network through the target Q network and y_i. The sixth step uses the optimizer to update θ^Q. Similarly, the seventh step calculates the policy gradient of the policy network and updates θ^μ of the current policy network like the Q network, and finally soft updates the two target networks [25].

In the designed system model, the state that the Markov decision process focuses on consists of a collection of device request task data. For each device task request, offloading decisions and computing resource allocation operations need to be made, which can be included in actions. In the space, the reward function is designed from the perspective of minimizing the task processing of the device. The reward function is defined as the total time delay of all current tasks. The whole process of specific algorithm execution is shown in Algorithm 1:

Algorithm 1: DDPG solving process

1: Initialization of actor-critic network parameters;
2: The target network also initializes the parameters;
3: **for** episodes=1 to MAX_EPISODES **do**
4: **for** step=1 to MAX_STEPS **do**
5: The actor network is responsible for the generation of actions;
6: Return rewards and next state based on action interaction;
7: Save to experience replay pool;
8: Get target value through critic network;
9: Obtain independent target values using target network;
10: The gradient is determined by the target value of the actor-critic network and the target network target value;
11: Update parameters in actor and critic networks based on gradients;
12: Update parameters in the target network based on weights in the actor and critic networks;
13: **end for**
14: **end for**

5. EXPERIMENTAL ANALYSIS

In the actual application, the MEC network benefits from the optimization of the communication network in terms of safety management and load balancing. For example, the gold mine power grid project in Western Australia has seen a significant increase in demand for power grid stability and data communications as production in the mining area has expanded. Traditional communication networks in remote areas rely on old copper wires, resulting in frequent slow responses and processing interruptions. This can be achieved through the introduction of advanced wireless communications and mobile edge computing technologies, which will increase data transmission rates, which are critical for real-time monitoring of grid status, especially during periods of high demand or during emergencies. The high-speed network ensures that the large amount of data collected from various sensors and monitoring equipment can be quickly and accurately transmitted to the control center, thereby achieving real-time understanding of the status of the mine power grid. At the same time, it enhances network reliability and security, and uses advanced encryption technology and security protocols to provide higher data security for the mine power grid. At the same time, the application of wireless communication technology improves the redundancy of the network, so that the system can maintain stable operation even when some nodes fail.

In this section, numerical results are presented through the Python simulation environment to demonstrate the performance of the mining area power grid system with the suggested three-tier computing framework and task offloading method. In the simulation setting, the total number of all mining base stations is N=10. Each base station can serve 5 mining equipment within a coverage radius of 2500m, and the wireless channel is modeled as a Rayleigh channel. The input data size for each task is uniformly distributed at N=10 kb, and the CPU cycles needed to process one bit of data are $C_{n,k}$ = 1000 cycle/bit. Key simulation parameters are detailed in Table 1 [26].

Table 1. Simulation parameters

Parameters	Value
B	10MHz
N=10	30dBm
N=10	-174dBm/Hz
N=10	N=10 cycle/s
N=10	N=10 cycle/s

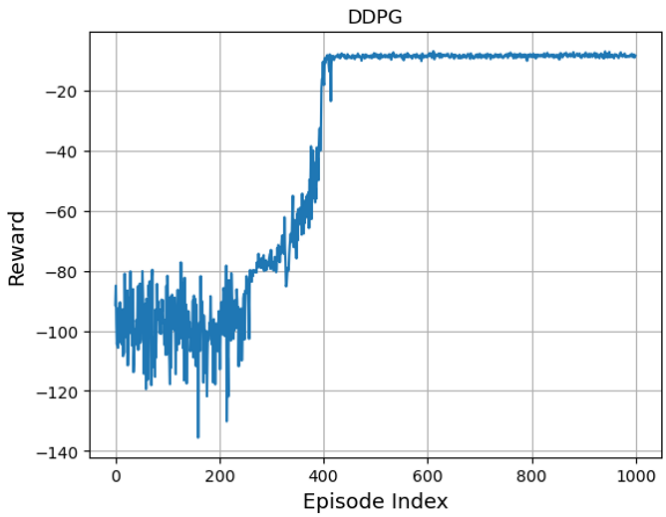

Figure 2. Convergence process of the proposed cloud-edge-end collaborative offloading strategy

First, Figure 2 illustrates how the suggested collaborative cloud-edge-terminal computing approach converges, where the y-axis indicates the overall delay of the system. As the number of training steps increases, the agent learns effectively, and we observe that the algorithm converges quickly, even with the scale of the setup. Specifically, after approximately 400 iterations, when the training steps reach a certain threshold, the total system delay stabilizes. This indicates that the algorithm has successfully optimized the offloading strategy and reached a point of diminishing returns, where further iterations yield minimal improvements. This quick convergence highlights the efficiency of our approach in adapting to the system's dynamics, ensuring fast and reliable performance for real-time tasks.

In Figure 3, we compare the proposed model performs both with and without incorporating a security layer. The figure

illustrates the overall overhead involved in executing a computational task for various data sizes. As expected, the model with the security layer introduces a higher total overhead compared to the model without it. This increase in overhead is primarily due to the additional time required for encryption, decryption, and the communication processes associated with large data volumes. Although this added security layer increases the delay, it is a necessary trade-off to ensure data confidentiality and integrity. The results suggest that while the security layer does introduce some additional delay, it provides significant protection against potential security threats, making it a critical component for applications in sensitive environments, such as power grid systems.

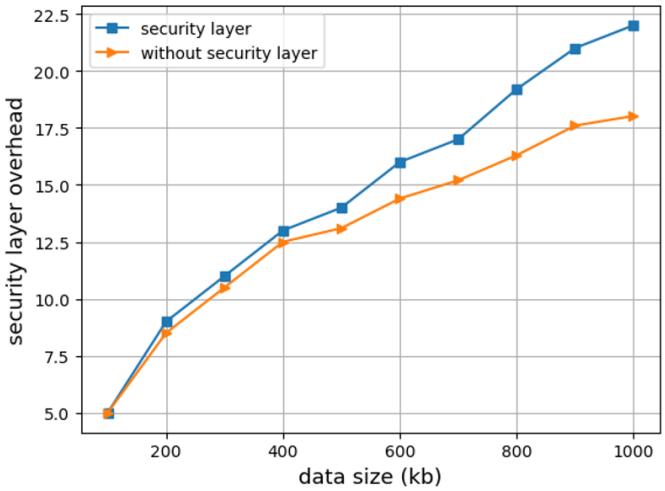

Figure 3. Impact of the security layer on the total overhead in different data sizes

Moreover, three distinct task transfer methods are executed to evaluate and compare performance, detailed as follows [27]:

1. Collaborative cloud-edge-terminal: The cloud-edge-terminal collaborative task offloading strategy proposed in this paper. The DDPG algorithm is used to solve the offloading strategy, and the computing tasks are collaboratively processed by power equipment, power grid base stations, and the cloud.

2. Mobile cloud computing: executing computational tasks on both devices and the cloud, while excluding the consideration of edge computing at base stations.

3. Local computing: every computational task is executed directly on the user's own device.

Figure 4 presents a comparison of the performance of three task offloading methods as the average task data size increases. It is clear that the suggested collaborative cloud-edge computing approach results in the least delay among the three. Furthermore, for smaller data sizes, the local computing method performs similarly to the mobile cloud computing method. This is because edge computing resources are relatively plentiful, making it advantageous to distribute computing tasks to both local devices and base station MECs. Consequently, the collaborative cloud-edge approach achieves lower delay than the others. For data size of 100kb, the mobile cloud computing method is 107% greater than that of the collaborative cloud-edge-terminal computing approach, primarily due to the longer transmission delay between the base station and the cloud. Offloading some tasks to the cloud helps to reduce the overall system delay. Therefore, when the data surpasses 800kb, the delay trends of the collaborative cloud-edge-terminal and mobile cloud computing methods converge.

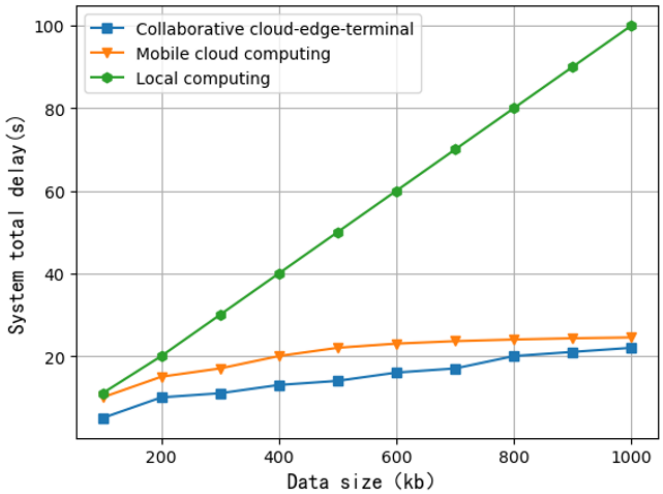

Figure 4. Performance comparison of offloading strategies in different data sizes

The proposed collaborative cloud-edge-terminal computing offloading framework has significant potential applications in smart grids and other IoT systems. In the context of smart grids, this framework can enhance real-time data processing by efficiently offloading tasks between local devices, edge servers, and cloud infrastructures. For instance, in power grid monitoring and fault detection, the ability to process data closer to the source—at the edge—can reduce latency, enabling quicker responses to grid anomalies and improving the reliability of power distribution. Additionally, for applications like predictive maintenance, where large volumes of sensor data are generated, the framework can optimize the offloading of computational tasks to cloud servers for advanced analytics, while keeping time-sensitive tasks at the edge. Beyond smart grids, this approach is also applicable to industrial IoT systems, where equipment condition monitoring and process optimization require low-latency processing combined with high computational capacity. By leveraging edge and cloud resources dynamically, the proposed framework can help achieve scalability and real-time responsiveness, which are essential for the effective operation of large-scale IoT networks. In summary, the proposed offloading strategy provides a flexible and efficient solution to address the computational and communication challenges faced by smart grids and IoT systems, paving the way for smarter, more responsive infrastructure.

Based on the simulation results, while the proposed collaborative cloud-edge-terminal computing strategy shows significant improvements in system performance, several limitations and challenges remain. Scalability is a key concern, as the current setup assumes a relatively small number of base stations and tasks. As the network size grows, the system may struggle with resource allocation and communication overhead, leading to potential bottlenecks, especially at the edge and cloud layers. Additionally, the robustness of the approach in real-world scenarios is another challenge. The simulation assumes ideal conditions, but in practice, factors like network interference, fluctuating signal strength, and varying computational capabilities of edge devices could introduce variability and delays, potentially undermining the

performance. To address these issues, dynamic resource management and adaptive offloading strategies could be implemented to handle fluctuating workloads and network conditions. Moreover, while the security layer is essential for protecting sensitive data, its overhead in terms of encryption and communication delays could degrade the system's overall efficiency, especially as data size increases. Future work could focus on optimizing security protocols to minimize performance trade-offs. Overall, while the approach demonstrates potential, its practical deployment will require addressing these scalability, robustness, and efficiency challenges.

6. CONCLUSION

In the MEC networks, the business operation based on optical fiber communication has been widely used in the transmission lines of the power system. However, the power grid centralized control terminals have many points, the layout is scattered, and the number of optical fiber cables erected is large. This paper addresses the digital and intelligent construction needs of the power grid. It combines the characteristics of the distribution of the substation power equipment with the mobile edge computing technology to propose a cloud, edge, and end synergy oriented to the security control of the MEC networks. The task offloading method of the three-layer computing framework optimizes the offloading strategy for the computing tasks of the grid terminal equipment, thereby minimizing the overall delay of the safety processing tasks of the electric equipment in the mining area. Combining the DDPG algorithm to solve for the optimal solution to the device offloading policy. Experimental evidence indicates that the method exhibits robust model convergence performance and is capable of dynamic adaptation to complex environments. The numerical results demonstrate that the proposed cloud-side-end collaborative task offloading strategy is effective in reducing system delay in the MEC networks. This confirms the algorithm's practicality and success, advances the digitization and intelligence of the grid, and significantly mitigates the security risks associated with intelligent control in MEC networks.

REFERENCE

[1] Baldoni, G., Quevedo, J., Guimaraes, C., de la Oliva, A., Corsaro, A. (2023). Data-centric service-based architecture for edge-native 6G network. IEEE Communications Magazine, 62(4): 32-38. https://doi.org/10.1109/MCOM.001.2300178

[2] Abbas, N., Zhang, Y., Taherkordi, A., Skeie, T. (2017). Mobile edge computing: A survey. IEEE Internet of Things Journal, 5(1): 450-465. https://doi.org/10.1109/JIOT.2017.2750180

[3] Papa, A., von Mankowski, J., Vijayaraghavan, H., Mafakheriy, B., Gorattiy, L., Kellerer, W. (2023). Enabling 6G applications in the sky: Aeronautical federation framework. IEEE Network, 38(1): 254-261. https://doi.org/10.1109/MNET.132.2200526

[4] Kiani, A.Y., Hassan, S.A., Su, B., Pervaiz, H., Ni, Q. (2020). Minimizing the transaction time difference for NOMA-based mobile edge computing. IEEE Communications Letters, 24(4): 853-857. https://doi.org/10.1109/LCOMM.2020.2966442

[5] Bozorgchenani, A., Mashhadi, F., Tarchi, D., Monroy, S.A.S. (2020). Multi-objective computation sharing in energy and delay constrained mobile edge computing environments. IEEE Transactions on Mobile Computing, 20(10): 2992-3005. https://doi.org/10.1109/TMC.2020.2994232

[6] Sharif, Z., Jung, L.T., Razzak, I., Alazab, M. (2021). Adaptive and priority-based resource allocation for efficient resources utilization in mobile-edge computing. IEEE Internet of Things Journal, 10(4): 3079-3093. https://doi.org/10.1109/JIOT.2021.3111838

[7] Elgendy, I.A., Zhang, W.Z., Zeng, Y., He, H., Tian, Y.C., Yang, Y. (2020). Efficient and secure multi-user multi-task computation offloading for mobile-edge computing in mobile IoT networks. IEEE Transactions on Network and Service Management, 17(4): 2410-2422. https://doi.org/10.1109/TNSM.2020.3020249

[8] Abrar, M., Ajmal, U., Almohaimeed, Z.M., Gui, X., Akram, R., Masroor, R. (2021). Energy efficient UAV-enabled mobile edge computing for IoT devices: A review. IEEE Access, 9: 127779-127798. https://doi.org/10.1109/ACCESS.2021.3112104

[9] Henna, S., Davy, A. (2020). Distributed and collaborative high-speed inference deep learning for mobile edge with topological dependencies. IEEE Transactions on Cloud Computing, 10(2): 821-834. https://doi.org/10.1109/TCC.2020.2978846

[10] Khan, L.U., Yaqoob, I., Tran, N.H., Kazmi, S.A., Dang, T.N., Hong, C.S. (2020). Edge-computing-enabled smart cities: A comprehensive survey. IEEE Internet of Things Journal, 7(10): 10200-10232. https://doi.org/10.1109/JIOT.2020.2987070

[11] Tout, H., Mourad, A., Kara, N., Talhi, C. (2021). Multi-persona mobility: Joint cost-effective and resource-aware mobile-edge computation offloading. IEEE/ACM Transactions on Networking, 29(3): 1408-1421. https://doi.org/10.1109/TNET.2021.3066558

[12] Shi, W., Zhang, J., Zhang, R. (2019). Share-based edge computing paradigm with mobile-to-wired offloading computing. IEEE Communications Letters, 23(11): 1953-1957. https://doi.org/10.1109/LCOMM.2019.2934411

[13] Shokouhi, M.H., Hadi, M., Pakravan, M.R. (2024). Mobility-aware computation offloading for hierarchical mobile edge computing. IEEE Transactions on Network and Service Management, 21(3): 3372-3384. https://doi.org/10.1109/TNSM.2024.3386845

[14] Yousafzai, A., Yaqoob, I., Imran, M., Gani, A., Noor, R.M. (2019). Process migration-based computational offloading framework for IoT-supported mobile edge/cloud computing. IEEE Internet of Things Journal, 7(5): 4171-4182. https://doi.org/10.1109/JIOT.2019.2943176

[15] Alfakih, T., Hassan, M.M., Gumaei, A., Savaglio, C., Fortino, G. (2020). Task offloading and resource allocation for mobile edge computing by deep reinforcement learning based on SARSA. IEEE Access, 8: 54074-54084. https://doi.org/10.1109/ACCESS.2020.2981434

[16] Bozorgchenani, A., Mashhadi, F., Tarchi, D., Monroy, S.A.S. (2020). Multi-objective computation sharing in energy and delay constrained mobile edge computing environments. IEEE Transactions on Mobile Computing,

20(10): 2992-3005. https://doi.org/10.1109/TMC.2020.2994232
[17] Cha, N., Wu, C., Yoshinaga, T., Ji, Y., Yau, K.L.A. (2021). Virtual edge: Exploring computation offloading in collaborative vehicular edge computing. IEEE Access, 9: 37739-37751. https://doi.org/10.1109/ACCESS.2021.3063246
[18] Kumar, V., Mukherjee, M., Lloret, J., Zhang, Q., Kumari, M. (2022). Delay-optimal and incentive-aware computation offloading for reconfigurable intelligent surface-assisted mobile edge computing. IEEE Networking Letters, 4(3): 127-131. https://doi.org/10.1109/LNET.2022.3187720
[19] Waheed, A., Shah, M.A., Mohsin, S.M., Khan, A., Maple, C., Aslam, S., Shamshirband, S. (2022). A comprehensive review of computing paradigms, enabling computation offloading and task execution in vehicular networks. IEEE Access, 10: 3580-3600. https://doi.org/10.1109/ACCESS.2021.3138219
[20] Samanta, A., Chang, Z. (2019). Adaptive service offloading for revenue maximization in mobile edge computing with delay-constraint. IEEE Internet of Things Journal, 6(2): 3864-3872. https://doi.org/10.1109/JIOT.2019.2892398
[21] Nguyen, T.H., Thi, H.L.N., Le Hoang, H., Tan, J., Luong, N.C., Xiao, S., Kim, D.I. (2024). Coded distributed computing for vehicular edge computing with dual-function radar communication. IEEE Transactions on Vehicular Technology, 73(10): 15318-15331. https://doi.org/10.1109/TVT.2024.3409554
[22] Asheralieva, A., Niyato, D. (2019). Learning-based mobile edge computing resource management to support public blockchain networks. IEEE Transactions on Mobile Computing, 20(3): 1092-1109. https://doi.org/10.1109/TMC.2019.2959772
[23] Ndikumana, A., Tran, N.H., Ho, T.M., Han, Z., Saad, W., Niyato, D., Hong, C.S. (2019). Joint communication, computation, caching, and control in big data multi-access edge computing. IEEE Transactions on Mobile Computing, 19(6): 1359-1374. https://doi.org/10.1109/TMC.2019.2908403
[24] Do-Duy, T., Van Huynh, D., Dobre, O.A., Canberk, B., Duong, T.Q. (2022). Digital twin-aided intelligent offloading with edge selection in mobile edge computing. IEEE Wireless Communications Letters, 11(4): 806-810. https://doi.org/10.1109/LWC.2022.3146207
[25] Abou El Houda, Z., Brik, B., Ksentini, A., Khoukhi, L., Guizani, M. (2022). When federated learning meets game theory: A cooperative framework to secure IIoT applications on edge computing. IEEE Transactions on Industrial Informatics, 18(11): 7988-7997. https://doi.org/10.1109/TII.2022.3170347
[26] Naouri, A., Wu, H., Nouri, N.A., Dhelim, S., Ning, H. (2021). A novel framework for mobile-edge computing by optimizing task offloading. IEEE Internet of Things Journal, 8(16): 13065-13076. https://doi.org/10.1109/JIOT.2021.3064225
[27] Ng, J.S., Lim, W.Y.B., Xiong, Z., Niyato, D., Leung, C., Miao, C. (2021). A double auction mechanism for resource allocation in coded vehicular edge computing. IEEE Transactions on Vehicular Technology, 71(2): 1832-1845. https://doi.org/10.1109/TVT.2021.3131395