Energy-Efficient Resource Allocation for MEC and Blockchain-Enabled IoT via CRL Approach

Meng Li*†, Pan Pei*, F. Richard Yu‡, Pengbo Si*†, Ruizhe Yang*†, and Zhuwei Wang*
*Faculty of Information Technology, Beijing University of Technology, Beijing, P.R. China
†Beijing Laboratory of Advanced Information Networks, Beijing, P.R. China
‡Department of Systems and Computer Engineering, Carleton University, Ottawa, ON, Canada
Email: limeng720@bjut.edu.cn, peipan@emails.bjut.edu.cn, richard.yu@carleton.ca, {sipengbo, yangruizhe, wangzhuwei}@bjut.edu.cn

GLOBECOM 2022 - 2022 IEEE Global Communications Conference | 978-1-6654-3540-6/22/$31.00 ©2022 IEEE | DOI: 10.1109/GLOBECOM48099.2022.10001421

***Abstract*—Driven by numerous emerging mobile devices and various quality of service requirements, mobile edge computing (MEC) has been recognized as a prospective paradigm to promote the computation capability of mobile devices, as well as reduce energy overhead and service latency of applications for the Internet of Things (IoT). However, there are still some open issues in the existing research works: 1) limited network and computing resource, 2) simple or non-intelligent resource management, 3) ignored security and reliability. In order to cope with these issues, in this article, 6G and blockchain technology are considered to improve network performance and ensure the authenticity of data sharing for the MEC-enabled IoT. Meanwhile, a novel intelligent optimization method named as collective reinforcement learning (CRL) is proposed and introduced, to realize intelligent resource allocation, meet distributed training results sharing and avoid excessive consumption of system resources. Based on the designed network model, a cloud-edge collaborative resource allocation framework is formulated. By joint optimizing the offloading decision, block interval and transmission power, it aims to minimize the consumption overheads of system energy and service latency. Then the formulated problem is designed as a Markov decision process, and the optimal strategy can be obtained by the CRL. Some evaluation results reveal that the system performance based on the proposed scheme outperforms other existing schemes obviously.**

***Index Terms*—mobile edge computing, Internet of Things, 6G, blockchain, collective reinforcement learning.**

I. Introduction

Focused on the most Internet of Things (IoT) application scenarios, it is crucial to ensure the communications between devices and devices with high transmission rate and low energy overhead. However, the fifth-generation (5G) communication networks are inefficient to satisfy the data-intensive applications [1]. Moreover, the security and authenticity of data interaction or sharing process are usually hard to be guaranteed adequately since there is no trusted entity to ensure the privacy protection for sensitive and personal data [2]. Meanwhile, since lots of the IoT nodes are lightweight, the massive or complicated data tasks are hardly to be performed by these IoT nodes independently.

This work is partially supported through the National Natural Science Foundation of China under Grant 61901011, the Foundation of Beijing Municipal Commission of Education under Grant KM202110005021 and KM202010005017, and the Beijing Natural Science Foundation under Grant L211002 and L202016.

Fortunately, the sixth-generation (6G) networks are anticipated to offer communication with high transmission rate for the IoT scenarios. Specifically, terahertz (THz) band (0.1-10THz), as one of the potential candidate spectrum band, is proposed for 6G wireless communications [3]. On the other hand, mobile edge computing (MEC) has attracted more attention since it has many features including higher computing efficiency, lower service latency as well as fewer energy overhead [4], [5]. Therefore, combining MEC and 6G for the IoT has been proposed into several researches [6]–[8].

Several excellent works have been studied to minimize the energy overhead and service latency by the resource allocation of computing/network and computation offloading based on the MEC [9], [10]. In addition, the cloud-edge collaborative architecture has been proposed to improve computing capacity and relieve overhead of MEC server [11]. Meanwhile, due to the dynamic features of network resource in the IoT, several intelligent optimization algorithms are usually deployed to execute resource allocation and computation offloading.

Although MEC server have more computing resources than IoT devices, they are still subject to resource-constrained. Meanwhile, the conventional machine learning (ML) methods usually need single individual training, and also rely on smart devices to have powerful computing resources. Therefore, it is a challenge for MEC node to perform own computation tasks and execute data learning or training at the same time. In order to alleviate the problem about insufficient resource of single MEC node for executing ML algorithm, we employ the sharing of learning results to improve the ML algorithm, which is called by collective reinforcement learning (CRL). However, we also need consider how to guarantee the sharing of learning results safely and reliably.

For the security and reliability issues for data sharing, blockchain is recognized as a promising technology recently. In the IoT scenarios, the features of blockchain such as decentralized, sharing and tamper-proof can be applied to ensure data security and reliability. Moreover, thanks to the same decentralized characteristics of blockchain and MEC, it will become natural that combines two advanced technologies [12]. Nevertheless, the problem of energy overhead and complicated computation caused by the blockchain consensus procedure

cannot be ignored.

To solve above problems and challenges, in this paper, we explore a cloud-edge collaborative computation offloading framework for blockchain-enabled IoT system through 6G networks. Specifically, blockchain technology is utilized to protect the security and authenticity of the data sharing or interaction. Besides, for the dynamic and cooperative computation offloading problem, it can be formulated as a Markov decision process (MDP). Then, we utilize the CRL method to deal with the dynamic and complicated optimization problem.

The remainder of this article is organized as follows. In the following section, we present the system model and architecture. The optimization problem is proposed in Section III. Then, we introduce CRL method to solve the formulated problem in Section IV. In Section V, several experiment results are discussed and analysed. Finally, this paper is concluded and some future works are proposed in Section VI.

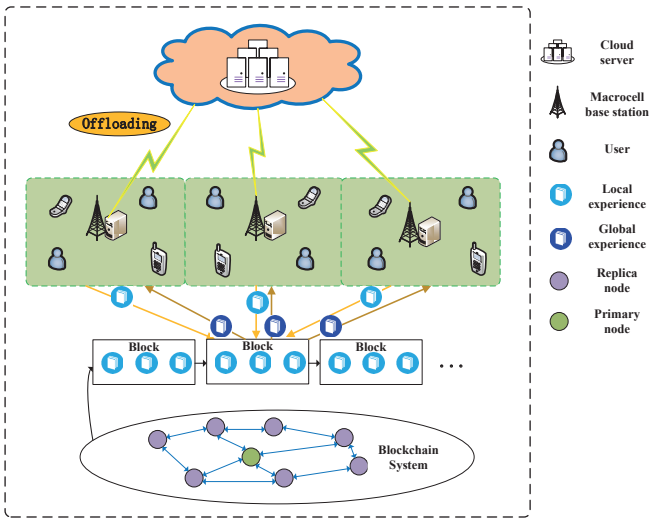

Fig. 1: System model.

II. System Model and Architecture

The proposed system model is presented in this section, including network model, transmission model, blockchain model as well as computation model.

A. Network Model

As presented in Fig. 1, each base station (BS) is equipped with an MEC server. The set $M = \{1, 2, ..., m\}$ denotes MEC servers collection and the energy of each MEC server is denoted by $\nu(t) = \{\nu_1(t), \nu_2(t), ..., \nu_m(t)\}$. Meanwhile, we consider that all of the BSs form a blockchain network where each BS acts as a node. The private training data collected by MEC server can be recorded as a transaction and transmitted to the blockchain for information validation and consensus at each time slot t. In this paper, we suppose that each MEC node has received a large amounts of computing tasks offloaded from multiple mobile devices simultaneously via 6G mobile networks. Therefore, in order to alleviate the computing load of MEC server, a cloud server would be deployed to promote energy efficiency in the proposed system.

B. Transmission Model

For the transmission model, THz channel is used for data transmission from edge server to cloud server. To derive the wireless transmission rate, we need to obtain the total path loss for uplink THz wireless transmission at first. As shown in [13], the total path loss is obtained by multiplying the free-space loss H_{spread} and the molecular absorption attenuation H_{abs}, which can be represented as

$$H_{total}(w, l) = H_{spread}(w, l) \cdot H_{abs}(w, l), \tag{1}$$

where w means the carrier frequency, l represents the distance from MEC server to cloud server.

Specifically, the free-space propagation loss is related with w and l, which can be represented as

$$H_{spread}(w, l) = (\frac{4\pi wl}{c})^2, \tag{2}$$

where $c = 3 \times 10^8 m/s$ is the velocity of light. Meanwhile, the absorption loss is given by

$$H_{abs}(w, l) = \frac{1}{\tau(w, l)} = e^{\kappa_{abs}(w)l}, \tag{3}$$

where $\tau(w, l)$ is defined as the transmittance of the medium, it is obtained by following the Beer-Lambert law. And the medium absorption coefficient is denoted by $\kappa_{abs}(w)$.

Based on the defined total path loss, the signal-noise ratio of channel for THz communications can be represented as

$$\gamma_u = \frac{P_u(t) \cdot G_t(t) \cdot G_r(t) \cdot H_{total}(w, l)}{N_0}, \tag{4}$$

where $P_u(t)$ is denoted as the transmission power at the edge server, N_0 is the power of received noise at the cloud server, $G_t(t)$ and $G_r(t)$ are the transmitting and receiving antenna gain, respectively.

Hence, the data rate of uplink channel for THz communication can be represented as

$$R_u(t) = B_u \cdot log_2(1 + \gamma_u), \tag{5}$$

where B_u represents the communication bandwidth between the MEC server and the cloud server.

C. Blockchain Model

For the blockchain system, there are m consensus nodes, which are denoted by $M = \{1, 2, ..., m\}$. We adopt the practical Byzantine fault tolerance (PBFT) [14] as the smart consensus protocol in this paper. According to the PBFT, the consensus process is divided into five phases:

a) *Request*: In a period of time, the edge nodes submit the data consensus requirement to the blockchain system. The primary node, which is assigned by blockchain system randomly, verifies its signature firstly. If it is valid, then the message authentication code (MAC) will be verified. Meanwhile, the transactions will be packaged a new block by the primary node. This procedure is performed in the block interval $i(t)$. It is assumed that the

required CPU cycles for generating/verifying an MAC and a signature are given by δ and θ, respectively.

Hence, the consumed CPU cycles can be defined as

$$c_1(t) = (\theta + \delta) \cdot \frac{g(t)}{o(t)}, \tag{6}$$

where $g(t)$ is defined as the total transaction batch size transmitted, and $o(t)$ represents the average size of transaction.

b) *Pre-prepare*: The primary node broadcasts the signed block and generates $(m-1)$ MACs and one signature to replica nodes for validation. As long as receiving above messages sent by the primary node, the MAC and signature of the block will be verified by replica nodes firstly, and the signature and MAC of the transactions will be checked secondly.

Thus, the consumed CPU cycles at the primary node and replica nodes are defined as

$$c_{2p}(t) = \theta + (m-1) \cdot \delta, \tag{7}$$

and

$$c_{2r}(t) = (\theta + \delta) \cdot (1 + \frac{\lambda \cdot g(t)}{o(t)}), \tag{8}$$

where λ denotes the probability of the correct transactions verified in the request phase.

c) *Prepare*: The verified replica nodes generate $(m-1)$ MACs and one signature, as well as send to all the other nodes. Afterwards, all of the MACs and signatures will be verified by both replica nodes and primary node. Once upon receipt that failed validation nodes less than $f = (m-1)/3$, it enters the commit phase.

Thus, the consumed CPU cycles can be represented as

$$c_{3p}(t) = \underbrace{2f \cdot (\theta + \delta)}_{verify}, \tag{9}$$

and

$$c_{3r}(t) = \underbrace{2f \cdot (\theta + \delta)}_{verify} + \underbrace{\theta + (m-1) \cdot \delta}_{generate}. \tag{10}$$

d) *Commit*: The difference from the previous period is that all of nodes will generate $(m-1)$ MACs and a signature, and deliver them to all the blockchain nodes. In the verifying side, if the number of incorrectly fault nodes less than f, the next step will be continued.

Thus, the consumed CPU cycles can be expressed as

$$c_4(t) = \underbrace{2f \cdot (\theta + \delta)}_{verify} + \underbrace{\theta + (m-1) \cdot \delta}_{generate}. \tag{11}$$

e) *Relay*: After verified the committed message, each of nodes delivers one reply message which including one MAC and one signature to the edge server. Moreover, if the correctly verified nodes have exceeded $2f$, it means that the consensus process has been completed, and the valid block will be appended to the blockchain system.

Then, the consumed CPU cycles can be given by

$$c_5(t) = \underbrace{2f \cdot (\theta + \delta)}_{verify} + \underbrace{(\theta + \delta)}_{generate}. \tag{12}$$

In term of above analysis, the total consumed CPU cycles of the whole process can be given by

$$\begin{aligned} c_{total}(t) =& [(1+\lambda) \cdot \frac{g(t)}{o(t)} + 6f + 5] \cdot \theta \\ &+ [(1+\lambda) \cdot \frac{g(t)}{o(t)} + 6f + 3m - 1] \cdot \delta. \end{aligned} \tag{13}$$

D. Computation Model

In our system model, the MEC server receives the various computation data by mobile devices and blockchain system. However, it is impossible to execute these complex and heavy computing tasks for MEC server solely. In order to improve computing efficiency, the MEC server will offload computing tasks to the cloud server for data processing. Hence, if the MEC server selects to process computing tasks in local, the processing delay $t_c(t)$ generated by MEC server at time slot t can be defined as

$$t_c(t) = \frac{Q(t)}{F_m}, \tag{14}$$

and the energy overhead $e_c(t)$ generated by MEC server can be given by

$$e_c(t) = P_m \cdot \frac{Q(t)}{F_m}, \tag{15}$$

where $Q(t)$ denotes the consumed CPU cycles to execute the computation data tasks at time slot t, P_m means the computing power of the MEC server, and F_m stands for the CPU computation frequency of MEC server.

If the MEC server chooses the cloud server to execute the computation task, the latency and energy overhead can be defined as

$$t_s(t) = \underbrace{\frac{D(t)}{R_u}}_{transmission} + \underbrace{\frac{Q(t)}{F_s}}_{computation}, \tag{16}$$

and

$$e_s(t) = \underbrace{P_u \cdot \frac{D(t)}{R_u}}_{transmission} + \underbrace{P_s \cdot \frac{Q(t)}{F_s}}_{computation}, \tag{17}$$

where $D(t)$ is the size of data task at time slot t, F_s is the CPU computation frequency of cloud server, P_u is the transmission power of the edge server, and P_s means the computing power of the cloud server.

In addition, according to the total consumed CPU cycles required by one consensus process, the service latency generated by consensus process can be calculated as

$$T_b(t) = i(t) + \frac{c_{total}(t)}{F_b} + 4t_n, \tag{18}$$

where $i(t)$ represents the block interval, which is used to generate a new block. Moreover, $F_b = F_m$ or F_s, which depends on whether the agent chooses MEC server or cloud server to process the computation task of consensus. Furthermore, t_n is the broadcast latency between nodes.

As a result, the latency of whole system can be given by

$$T(t) = \begin{cases} t_c(t) + T_b(t), & \text{if without offloading,} \\ t_s(t) + T_b(t), & \text{else,} \end{cases} \tag{19}$$

and the energy overhead of whole system can be denoted as

$$E(t) = \begin{cases} e_c(t), & \text{if without offloading,} \\ e_s(t), & \text{else.} \end{cases} \tag{20}$$

III. Problem Formulation

In this section, the definition of state space, action space, and reward function are given and described in detail.

A. State Space

Let $\psi(t)$ represent the state space at time period t, it can be defined as

$$\psi(t) = \{\eta(t), \epsilon(t), G(t)\}, \tag{21}$$

where $\eta(t)$ and $\epsilon(t)$ represent the remaining computing resource availability of MEC server and cloud server, respectively. Besides, $G(t) = \{G_t(t), G_r(t)\}$ represents the wireless channels conditions from the MEC server to the cloud server.

B. Action Space

The action space includes the offloading decision $\rho(t)$, the block interval $i(t)$ and the transmission power $p(t)$. Correspondingly, the action space can be given by

$$a(t) = \{\rho(t), i(t), p(t)\}, \tag{22}$$

where $\rho(t) \in (0, 1)$ means the task offloading decision from MEC server to cloud server, $\rho(t) = 0$ denotes the MEC server processes the computing task in local, and $\rho(t) = 1$ means the MEC server offloading the computing task. In addition, $i(t) \in \{1, 2, ..., i\}$ indicates the block interval level, and $p(t)$ is the transmission power allocation when the data is transmitted from MEC server to cloud server.

C. Reward Function

In the cloud-edge collaborative IoT network, the optimization objective is to optimize both long-term system energy overhead and service latency. Therefore, the reward function can be calculated by

$$\begin{aligned} r(t) = & \begin{cases} \omega_1 \frac{1}{E(t)} + \omega_2 \frac{1}{T(t)}, & \text{if } C1 - C4 \text{ are satisfied,} \\ \omega_1 \frac{1}{E(t)} + \omega_2 \frac{1}{T(t)} - \varrho, & \text{otherwise,} \end{cases} \\ s.t. \quad & C1 : p(t) \in (0, P_{max}], \\ & C2 : T_b(t) \leq \beta \times i(t), \\ & C3 : D(t) \leq S(t), \\ & C4 : B \leq B_{max}, \end{aligned} \tag{23}$$

where ω_1 and ω_2 are the weight factor of the energy overhead and service latency, they should meet $\omega_1 + \omega_2 = 1$. ϱ is the penalty reward.

In this optimization problem, P_{max} in constraint $C1$ is the maximum transmission power between MEC server and cloud server, $C2$ represents the time delay limitation for block completion, where $\beta > 1$. $C3$ and $C4$ denote the size limitation of the task data and the limitation of the channel bandwidth, respectively.

IV. Problem Solution

In this section, we first introduce the DQN to solve the formulated optimization problem and achieve the long-term reward in edge node. After that the CRL algorithm executed on different edge nodes will be described in detail.

A. Local Deep Q Network Training in Edge Node

The DQN algorithm is composed of *Q*-learning algorithm and deep neural networks (DNNs). Based on the DQN method, the agent interacts with environment, learns from experience, and ultimately obtains and performs an optimization policy in light of the value function.

In the DQN, to disrupt the correlation between learning experiences, the experience poll technology has been used and it stores the experiences about $< \psi_t, a_t, r_t, \psi_{t+1} >$. Through randomly extracting some batches from experience poll, it makes the DNNs updating more efficient.

To assess action value in each step, there is a popular method called action-state value function $Q^\pi(\psi, a)$, which is defined as

$$Q^\pi(\psi, a) = E^\pi[\sum_{k=1}^{\infty} \gamma^k r_{t+k+1} | \psi_t = \psi, a_t = a], \tag{24}$$

where $E^\pi[*]$ means the mathematical expectation, $\gamma \in (0, 1)$ denotes the discounter factor to balance the immediate reward and the long-term reward.

Specifically, the DQN uses the temporal difference approach to update action-state values, which can be expressed as

$$\begin{aligned} Q^*(\psi_t, a_t) \leftarrow & Q^*(\psi_t, a_t) + \alpha[r(\psi_t, a_t) \\ & + \gamma \max_{a_{t+1}} Q^*(\psi_{t+1}, a_{t+1}) - Q^*(\psi_t, a_t)], \end{aligned} \tag{25}$$

where $\alpha \in (0, 1]$ means the learning rate.

In each training episode, the weights in the DNNs are continually adjusted to minimize the loss function $\mathbb{L}(\theta)$ to evaluate the real $Q^*(\psi_t, a_t)$, then the $\mathbb{L}(\theta)$ is denoted as

$$\begin{aligned} \mathbb{L}(\theta) = E[(r(\psi_t, a_t) + \gamma \max_{a_{t+1}} Q^*(\psi_{t+1}, a_{t+1}, \theta^*) \\ - Q^*(\psi_t, a_t, \theta))^2], \end{aligned} \tag{26}$$

where both θ and θ^* are the weights of the DNNs in the evaluated network and the target network, respectively.

B. Collective Reinforcement Learning among Edge Node

Considering most existing researches focus on the learning of an individual agent, they usually have not the ability to learn the experience from other agents. Meanwhile, even if the training results can be shared among different nodes, the conventional schemes cannot guarantee the data security and reliability in the process of data transmission and collection.

To address above issues, we consider using multiple agent nodes to share their training results, so as to reduce the computing and energy resource consumption of individual node. Meanwhile, in order to ensure the security and reliability of the shared data, the blockchain technology is considered and introduced for the CRL.

In the CRL algorithm, the blockchain plays an important role to store and distribute the local training results securely. After one intelligent node trains its DNNs locally, then packets the corresponding parameters of DNNs and the data of experience poll as a transaction send to blockchain system, each intelligent node can share the learning experience and leverage these parameters as the initialized parameters to start training their own DNNs model. Specifically, the updated loss function in this case can be expressed as

$$\mathbb{L}(\theta) = E[(r(\psi_t, a_t) + \gamma \max_{a_{t+1}} Q^*(\psi_{t+1}, a_{t+1}, \widehat{\theta}^*) - Q^*(\psi_t, a_t, \widehat{\theta}))^2], \tag{27}$$

where $\widehat{\theta}^* = \sum_{n=1}^{N} \frac{\theta_n^*}{N}$ and $\widehat{\theta} = \sum_{n=1}^{N} \frac{\theta_n}{N}$, both θ_n^* and θ_n are the parameters of the DNNs downloaded from blockchain. N is the number of intelligent nodes shared the training parameters.

Especially, in order to avoid the oscillation of DNNs and improve the rate of convergence, we consider that each intelligent node has the same structure, such as the same neural cells and the same layers.

V. Experiment Results and Discussion

In this section, we list the experiment parameters at first, and then present and discuss the experiment results.

A. Experiment Parameters

In this simulation, TensorFlow 1.13.1 with Python 3.6 on Windows 10 is employed to simulate experiments.

In the network scenario, there are six BSs equipped with MEC server, one cloud server and various mobile devices scattered in the area, and the average computing resources of MEC server and cloud server are 1 GHz and 10 GHz, respectively. The antenna gain range for transmitting or receiving is $0 - 20$ dBi [15]. The gaussian noise power spectral density is -174 dBm/Hz [8]. The CPU cycles for a signature and an MAC are 1 Mcycles and 10 Mcycles, respectively. The weight factor ω_1 and ω_2 are set as 0.5 and 0.5.

B. Performance Comparison and Analysis

We first present the convergence of the proposed scheme by comparing it with following three intelligent algorithms: 1) *Traditional DQN algorithm*: it means that no agent shares its learning experience to each other, thus each agent just uses its own training and learning results. 2) *Proposed CRL algorithm*: it means that multi-agents can collaboratively train and learn in this case, and each agent can learn the training experience shared from others by blockchain system. 3) *Conventional random selection scheme*: it means that the agent takes action without any optimization policy.

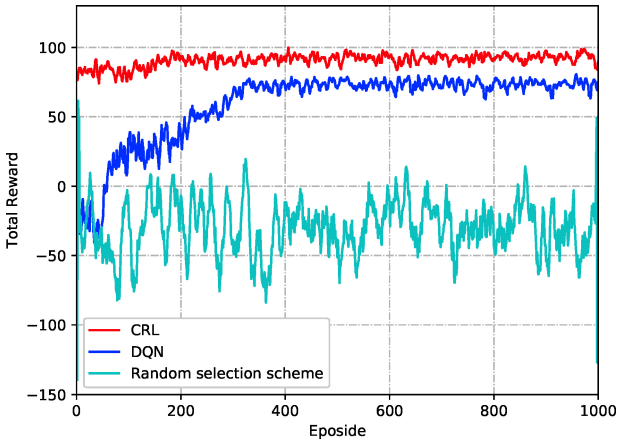

Fig. 2: Total reward with diverse learning algorithms.

As shown in Fig. 2, the convergence performance of CRL exceeds the other algorithms distinctly, whose total reward at beginning is near the converged result. The reason is that the edge agent will obtain DNNs training parameters from other agents through the CRL, there is no need to train the DNNs separately, thereby reducing its own computing expenditures. Moreover, it also means that the training efficiency can be improved significantly.

To further examine the validity of the proposed framework, four comparison schemes are considered in this paper: 1) *Proposed scheme without offloading decision*: the computation tasks are executed locally. 2) *Proposed scheme with fixed block interval*: the frequency of generating blocks is identical. 3) *Proposed scheme with fixed transmission power allocation*: the power of transmission is fixed. 4) *Existing work*: it is the conventional scheme without any adjustment or selection.

Fig. 3 shows how the task data size effects the weight of system consumption under various schemes. The system consumption weight represents the weighted value of total energy overhead and service latency. We can see that the system consumption weight increases with the increasing size of task data. Meanwhile, we can also find that the system consumption weight of the proposed scheme keeps optimal. When the size of task data increases, the systems computing tasks become heavier, it inevitably leads to longer processing time and more energy overhead. Based on the observation, it can be proved that selecting the appropriate offloading decision, block interval and transmission power can enhance the system performance significantly.

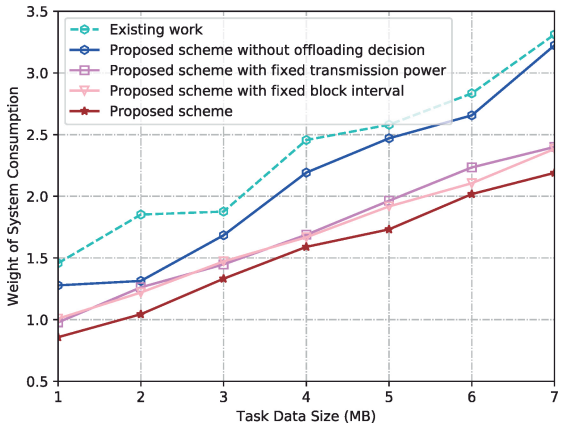

Fig. 3: Weight of system consumption versus the size of task data.

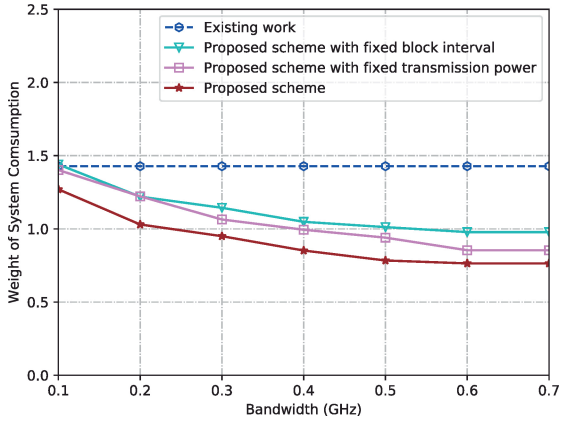

Fig. 4: Weight of system consumption versus channel bandwidth.

Fig. 4 depicts the change of the weight of system consumption with the transmission channel bandwidth under different schemes. We can find that the system consumption weight drops gradually and reaches a stable state with the increasing channel bandwidth. Based on the Shannon's Theorem, with the increasing channel bandwidth, the transmission rate can increase accordingly, resulting in that the weight of system consumption will reduce. Because of the constraint of channel bandwidth, when the bandwidth reaches the highest point, the energy overhead and service latency eventually tend to be stable. We can also observe that the system consumption weight always keeps stable with the increasing channel bandwidth based on the existing works. The reason is that the existing works have not offloading decision and the computing tasks are processed locally.

VI. Conclusions and Future Work

In this paper, we have designed and studied a cloud-edge collaborative resource allocation framework for MEC-enabled IoT. In order to improve the system performance effectively, the technology of THz communication from 6G was considered in the proposed network model. Meanwhile, to ensure the security of data interaction and sharing, the blockchain technology was also introduced and applied. Then, we jointly optimized the offloading decision, block interval and transmission power to minimize the system overhead. Since the high-dynamic and continuity of system state, we modeled the optimization problem as an MDP, and proposed a CRL algorithm to find the optimal policy with fast convergence performance to solve the problem. Evaluation results showed that our proposed scheme achieved better convergence and effectiveness than other existing schemes. In future work, the throughput of blockchain system should be considered and optimized via the CRL algorithm. Moreover, the resource allocation for content caching should also be concerned in the cloud-edge collaborative IoT.

References

[1] W. Dong, Z. Xu, X. Li, and S. Xiao, "Low-cost subarrayed sensor array design strategy for IoT and future 6G applications," *IEEE Internet of Things J.*, vol. 7, no. 6, pp. 4816–4826, Jun. 2020.

[2] M. Shen, X. Tang, L. Zhu, X. Du, and M. Guizani, "Privacy-preserving support vector machine training over blockchain-based encrypted IoT data in smart cities," *IEEE Internet of Things J.*, vol. 6, no. 5, pp. 7702–7712, Oct. 2019.

[3] Z. Chen, X. Ma, B. Zhang, Y. Zhang, Z. Niu, N. Kuang, W. Chen, L. Li, and S. Li, "A survey on terahertz communications," *China Communi.*, vol. 16, no. 2, pp. 1–35, Feb. 2019.

[4] M. Li, F. R. Yu, P. Si, and Y. Zhang, "Green machine-to-machine (M2M) communications with mobile edge computing (MEC) and wireless network virtualization," *IEEE Communi. Mag.*, vol. 56, no. 5, pp. 148–154, May 2018.

[5] M. Li, F. R. Yu, P. Si, R. Yang, Z. Wang, and Y. Zhang, "UAV-assisted data transmission in blockchain-enabled M2M communications with mobile edge computing," *IEEE Network*, vol. 34, no. 6, pp. 242–249, Nov./Dec. 2020.

[6] Z. Liao, J. Peng, J. Huang, J. Wang, J. Wang, P. K. Sharma, and U. Ghosh, "Distributed probabilistic offloading in edge computing for 6G-enabled massive Internet of Things," *IEEE Internet of Things J.*, vol. 8, no. 7, pp. 5298–5308, Apr. 2021.

[7] W. Sun, H. Zhang, R. Wang, and Y. Zhang, "Reducing offloading latency for digital twin edge networks in 6G," *IEEE Trans. Veh. Technol.*, vol. 69, no. 10, pp. 12 240–12 251, Oct. 2020.

[8] J. Du, F. R. Yu, G. Lu, J. Wang, J. Jiang, and X. Chu, "MEC-assisted immersive VR video streaming over terahertz wireless networks: A deep reinforcement learning approach," *IEEE Internet of Things J.*, vol. 7, no. 10, pp. 9517–9529, Oct. 2020.

[9] L. Yang, M. Li, P. Si, R. Yang, E. Sun, and Y. Zhang, "Energy-efficient resource allocation for blockchain-enabled industrial Internet of Things with deep reinforcement learning," *IEEE Internet of Things J.*, vol. 8, no. 4, pp. 2318–2329, Feb. 2021.

[10] J. Zhang, X. Hu, Z. Ning, E. Ngai, L. Zhou, J. Wei, J. Cheng, B. Hu, and V. Leung, "Joint resource allocation for latency-sensitive services over mobile edge computing networks with caching," *IEEE Internet of Things J.*, vol. 6, no. 3, pp. 4283–4294, Jun. 2019.

[11] X. Zhang, H. Zhang, X. Zhou, and D. Yuan, "Energy minimization task offloading mechanism with edge-cloud collaboration in IoT networks," in *Proc. IEEE 93rd Veh. Technol. Conf.* Helsinki, Finland, Jun. 2021, pp. 1–7.

[12] R. Yang, F. R. Yu, P. Si, Z. Yang, and Y. Zhang, "Integrated blockchain and edge computing systems: A survey, some research issues and challenges," *IEEE Communi. Surv. Tutor.*, vol. 21, no. 2, pp. 1508–1532, Secondquarter. 2019.

[13] J. M. Jornet and I. F., Akyildiz, "Channel modeling and capacity analysis for electromagnetic wireless nanonetworks in the terahertz band," *IEEE Trans. Wireless Communi.*, vol. 10, no. 10, pp. 3211–3221, Oct. 2011.

[14] L. Zhang and Q. Li, "Research on consensus efficiency based on practical byzantine fault tolerance," in *Proc. International Conf. on Modelling, Identification and Control.* Guiyang, China, Nov. 2018, pp. 1–6.

[15] C. Han, A. O. Bicen, and I. F. Akyildiz, "Multi-ray channel modeling and wideband characterization for wireless communications in the terahertz band," *IEEE Trans. Wireless Communi.*, vol. 14, no. 5, pp. 2402–2412, May 2015.