Reliable Task Offloading in Sustainable Edge Computing with Imperfect Channel State Information

Peng Peng, Wentai Wu, *Member, IEEE*, Weiwei Lin, *Senior Member, IEEE*, Fan Zhang, Yongheng Liu, and Keqin Li, *Fellow, IEEE*

***Abstract*—As a promising paradigm, edge computing enhances service provisioning by offloading tasks to powerful servers at the network edge. Meanwhile, Non-Orthogonal Multiple Access (NOMA) and renewable energy sources are increasingly adopted for spectral efficiency and carbon footprint reduction. However, these new techniques inevitably introduce reliability risks to the edge system generally because of i) imperfect Channel State Information (CSI), which can misguide offloading decisions and cause transmission outages, and ii) unstable renewable energy supply, which complicates device availability. To tackle these issues, we first establish a system model that measures service reliability based on probabilistic principles for the NOMA-based edge system. As a solution, a Reliable Offloading method with Multi-Agent deep reinforcement learning (ROMA) is proposed. In ROMA, we first reformulate the reliability-critical constraint into an long-term optimization problem via Lyapunov optimization. We discretize the hybrid action space and convert the resource allocation on edge servers into a 0-1 knapsack problem. The optimization problem is then formulated as a Partially Observable Markov Decision Process (POMDP) and addressed by multi-agent proximal policy optimization (PPO). Experimental evaluations demonstrate the superiority of ROMA over existing methods in reducing grid energy costs and enhancing system reliability, achieving Pareto-optimal performance under various settings.**

***Index Terms*—Edge computing, task offloading, reliability, deep reinforcement learning.**

Manuscript received 7 April 2024; revised 13 August 2024 and 3 September 2024; accepted 4 September 2024. Date of publication 9 September 2024; date of current version 20 December 2024. This work is supported by National Natural Science Foundation of China (62072187, 62402198), Guangdong Major Project of Basic and Applied Basic Research (2019B030302002), Fundamental Research Funds for the Central Universities (21624348), and in part by the Major Key Project of PCL (PCL2023A09). The associate editor coordinating the review of this article and approving it for publication was D. Huang. *(Corresponding authors: Wentai Wu; Weiwei Lin.)*

Peng Peng is with the School of Future Technology, South China University of Technology, Guangzhou 510000, China, and also with the Department of New Network Technologies, Pengcheng Laboratory, Shenzhen 518000, China (e-mail: pengp@pcl.ac.cn).

Wentai Wu is with the Department of Computer Science, College of Information Science and Technology, Jinan University, Guangzhou 510632, China (e-mail: wentaiwu@jnu.edu.cn).

Weiwei Lin is with the School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China, and also with the Department of New Network Technologies, Pengcheng Laboratory, Shenzhen 518000, China (e-mail: linww@scut.edu.cn).

Fan Zhang and Yongheng Liu are with the Department of New Network Technologies, Pengcheng Laboratory, Shenzhen 518000, China (e-mail: zhangf@pcl.ac.cn; liuyh01@pcl.ac.cn).

Keqin Li is with the Department of Computer Science, State University of New York, New Paltz, NY 12561 USA (e-mail: lik@newpaltz.edu).

Digital Object Identifier 10.1109/TNSM.2024.3456568

I. Introduction

THE ONGOING evolution of the Internet of Things (IoT) technology is propelling the exponential growth of connected devices and latency-sensitive applications, which poses great challenges to the traditional cloud computing paradigm. To satisfy stringent Quality of Service (QoS) guarantees, edge computing has emerged as a promising solution and received much attention from across the industry and the academia. With computation resources deployed closer to the data, latency-critical tasks can be offloaded to and handled by mobile edge computing (MEC) servers without being transmitted to cloud centers through the backbone network. The potential of edge computing in mitigating network congestion and minimizing transmission delays makes it an effective approach for realizing low-latency services in scenarios involving computation-intensive and latency-sensitive applications [1].

Recent advances in edge computing involve the application of next-generation wireless communication technologies and the promotion of system sustainability. On the one hand, the fifth-generation (5G) network with Non-Orthogonal Multiple Access (NOMA) are widely deployed to improve the spectrum utilization efficiency, accommodate high-density IoT devices (IoTD), and construct large-scale edge computing systems [2]. It enables concurrent transmission of superimposed signals over the same resource block, with successive interference cancellation (SIC) to disentangle signals at the receiver. On the other hand, environmental sustainability has attracted widespread attention. Commonly used grid energy relies on fossil fuels, which emit greenhouse gases and lead to environmental pollution. Therefore, embracing green energy sources, such as solar and wind power, is critical for achieving sustainability and reducing carbon emissions [3].

However, the introduction of NOMA and green energy supply brings new problems to the table where the system or service reliability is an outstanding concern [4]. The risk of task failure becomes more critical as edge computing services are increasingly deployed in safety-critical scenarios. Both data transmission and task computation require reliability assurance during task offloading. For the transmission stage, co-channel interference from superposed NOMA transmission introduces transmission failure risks and threatens system reliability [5]. Additionally, the complexity of channel

1932-4537 © 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: SOUTH CHINA UNIVERSITY OF TECHNOLOGY. Downloaded on December 22,2024 at 02:44:42 UTC from IEEE Xplore. Restrictions apply.

variations makes it difficult to obtain perfect CSI estimations in practice, which complicates scheduling and further threatens system reliability [6]. From the computational viewpoint, devices require a sufficient energy supply for task processing. Otherwise, energy shortages can cause task failures. Although relying on green energy has environmental benefits, it introduces instability, unpredictability, and intermittent energy supply. Overly allocating energy poses future energy shortage threats, while insufficient energy allocation cannot meet delay requirements [7]. Consequently, offloading decisions and resource allocations must be made with an awareness of the transmission channel status and energy provision situations. It is increasingly important to circumvent transmission failures and energy shortages, thereby enhancing system reliability.

Much effort has been paid to the optimization of computation offloading strategies including traditional mathematical optimizations and heuristic approaches [8]. However, these solutions often experience diminished performance as the complexity of advanced edge computing systems increases. Deep Reinforcement Learning (DRL), as an experience-based deep learning paradigm, is increasingly favored for its efficiency and performance across diverse decision-making environments. The integration with the deep neural network enhances the feature extraction capability to recognize patterns within high-dimensional state information, and the interaction-exploration-exploitation feature of DRL facilitates continuous learning and adaptation to dynamic environments. Hence, DRL has shown great potential to enable intelligent computation offloading [9]. In addition, the development of multi-agent DRL can further empower agents for decentralized decision-making that fits well in realistic edge computing over autonomic devices [10].

Motivated by the demand for reliability at the network edge and the potential of multi-agent DRL, we present ROMA, a **R**eliable **O**ffloading method based on **M**ulti-**A**gent DRL. ROMA makes offloading decisions and allocates resources in a decentralized and intelligent manner, with the goal of reducing grid energy costs while maintaining latency requirements and system reliability. The main contributions of this paper are as follows:

1) We present a system model that features renewable energy-supplied IoTDs and NOMA transmission with imperfect CSI. We construct a reliability metric by modeling wireless transmission failures and IoTD power shortages.
2) We propose a reliable offloading approach called ROMA. By integrating the long-term reliability constraint into an instantaneous objective with Lyapunov optimization, a multi-agent DRL algorithm is applied for decentralized offloading decision-making. Via discretization transform, resource allocation on MEC servers is decomposed into a 0-1 knapsack problem to reduce action space.
3) With extensive numerical evaluation we demonstrate that the proposed ROMA outperformed several traditional methods and DRL-based algorithms, achieving significantly enhanced system reliability and lower grid energy costs.

The remainder of this article is organized as follows. In Section II, we briefly review related works in the field. Section III introduces the system model with problem formulation. The proposed ROMA is described in details in Section IV. Section V summarizes and discusses the experimental results. Finally, the paper is concluded in Section VI.

II. Related Work

A. Task Offloading and Resource Allocation

With the advancement of edge computing technologies, how to simultaneously realize energy efficiency and low latency approaches has been a problem of great interest. In the domain, an important line of study is focused on task offloading and resource allocation, which is challenging since the optimization is in most cases combinatorial. Many prior works generally focused on applying mathematical optimization and heuristic approaches to obtain feasible solutions. For example, [11] formulated it as a bilevel optimization problem and used the ant colony algorithm for the upper-level offloading decisions. Reference [12] transform the problem into a fractional programming problem aimed at maximizing energy efficiency. They combine the Dinkelbach algorithm with the Lagrangian multiplier to iteratively search for the optimal efficiency and corresponding variables. Reference [13] design a genetic simulated annealing-based particle swarm optimization algorithm to obtain near-optimal solutions in partial offloading. Reference [14] decoupled the problem and adopted a partial order-based heuristic approach for task offloading and solved the resource allocation problem by Lagrangian duality with Karush-Kuhn-Tucker (KKT) conditions.

Recent efforts tend to utilize RL/DRL for better complex environment optimization ability. Reference [15] formulate the offloading problem between UAV and MEC servers as a submodular non-cooperative game and propose two RL-based approaches to find the Nash Equilibrium. Reference [16] construct a UAV-assisted edge computing system and combine the Long-Short Term Memory (LSTM) module with Deep Deterministic Policy Gradient (DDPG) to make offloading decisions. In [17], Deep Q-Learning (DQN) is used for offloading scheduling in a vehicular edge network, where the CPU frequency allocation problem is proved to be convex and the optimal solution is obtained by gradient descent. In addition, the decentralized nature of edge computing is well-suited for multi-agent DRL algorithms, and many recent studies leverage it. Reference [18] consider a DNN-task offloading problem and propose a Dueling DQN-based method to achieve coordinated optimization of energy cost, latency, and utility. Reference [19] consider a multi-UAV assisted network and use the Q-Mix algorithm to promote collaboration. For a multi-access edge computing system, [20] propose a multi-agent DDPG (MADDPG) based offloading approach that follows the Centralized Training Decentralized Execution (CTDE) paradigm. Reference [21] target on a satellite-based IoT system, and propose a multi-agent information broadcasting and judging algorithm to improve the cooperation between agents.

Authorized licensed use limited to: SOUTH CHINA UNIVERSITY OF TECHNOLOGY. Downloaded on December 22,2024 at 02:44:42 UTC from IEEE Xplore. Restrictions apply.

B. Reliability-Aware Offloading Approaches

The common objective of computation offloading is to minimize task processing latency and energy consumption. As edge computing needs to support applications in security-sensitive domains, ensuring system reliability has emerged as a critical concern.

From the perspective of communication, a naïve way is to model failure probability as a fixed value and use a penalty term to thereby discourage task offloading in certain situations [22]. However, it is unrealistic to ignore the dynamics and uncertainties of channel state information. It is also common to assume task processing failure events as Poisson distributions whose parameters are obtained by fitting historical data [23]. Probabilistic approaches can better describe the uncertainty of such events. Recent research has begun to focus on the impact of imperfect CSI on transmission processes. Reference [24] consider a multi-user MEC system with orthogonal transmission and focus on the impact of imperfect CSI on transmission rates and latency. They propose a probabilistic delay constraint to ensure successful task processing. Similarly, [25] incorporate imperfect CSI into their model, which only affects transmission rates without causing outages, and propose a mathematical approach for optimization. Closer to our work, [26] discuss transmission outages due to imperfect CSI. However, instead of deriving a probabilistic expression for transmission outages as we do, they approximate the probabilistic constraint as a non-probabilistic one and incorporate it into the optimization objective.

From the energy management perspective, an unreliable energy supply for IoTDs can lead to energy outage and computation failure that threatens system reliability. To control energy usage, [27] introduce battery power and energy cost constraints to ensure that scheduling actions comply with energy management requirements. Reference [28] employ DRL approaches, incorporating penalties for constraint violations to prompt improved energy management practices. Reference [29] formulate the energy constraint as a long-term average constraint and transform it into a single-step stability objective. Reference [30] convert the challenge into a timeslot partitioning problem for energy harvesting and task processing, and allocate computation resources according to the harvested energy. Reference [31] similarly consider obtaining energy from MEC via radio frequency transmission and completely spending it on task processing at each timeslot, thus transforming the problem into a timeslot partitioning problem. To expand energy sources, [7] advocate for a hybrid energy supply model that combines renewable and grid energy to alleviate the insufficient and unstable renewable energy supply. The optimization target is to reduce grid energy dependency and enhance overall energy efficiency. Furthermore, [32] explore the adoption of wireless power transfer from MEC servers to IoTDs to secure a more consistent power supply, noting the importance of carefully managing energy transmission loss and channel occupancy.

The aim of our study is to ensure the edge system reliability while leveraging the sustainable energy sources. Our solution is an integrated offloading decision-making framework that considers both communication and computation risks.

TABLE I
List of Key Notations

Notaion	Definition
M	Number of APs
N	Number of IoTDs
y_n^t	Task data size for IoTD n
c_n^t	Task required CPU cycles for IoTD n
q_n^t	Task latency constraints for IoTD n
μ_n^t	Offloading decision by IoTD n
τ_n^t	Local resource allocation by IoTD n
$\tau_{n,m}^t$	Resource allocation of AP m to task from IoTD n
$h_{n,m}^t$	Actual Rayleigh fading channel gain
$\hat{h}_{n,m}^t$	Estimated Rayleigh fading channel gain
$\epsilon_{n,m}^t$	CSI estimation error term
f_n	Computation capacity of IoTD n
f_m	Computation capacity of AP m
b^{ub}	Battery capacity of all IoTDs
b_n^t	Available energy of IoTD n
p	Renewable energy arriving probability
w	Amount of energy per harvest
ψ_z	Requirement of reliability expectation

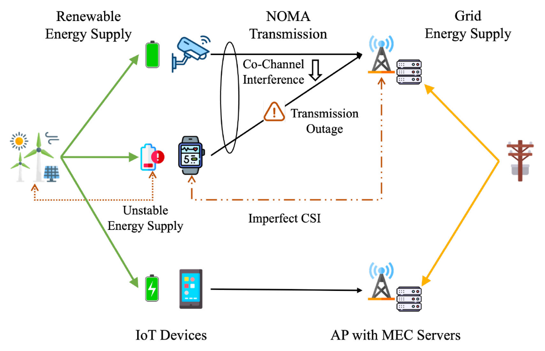

Fig. 1. Illustration of the constructed NOMA-based edge computing system with renewable energy supply.

III. System Model and Problem Formulation

To address the stringent reliability demands of edge computing systems, we first present our system model shown in Fig. 1. For ease of reading, key notations are summarized in Table I. We model system dynamics with discretized time slots $t \in \mathcal{T} = \{1, \ldots, T\}$. The network model consists of M multi-access points (APs), each combined with a MEC server to provide computing services, denoted by $m \in \mathcal{M} = \{1, \ldots, M\}$. Without loss of generality, APs and MEC servers are treated as integrated entities without distinction. N IoTDs indexed by $n \in \mathcal{N} = \{1, \ldots, N\}$ are deployed. The task, which is described as a triplet union (y_n^t, c_n^t, q_n^t), where y_n^t represents task data size, c_n^t denotes compute cycles required, and q_n^t is the tolerable delay, is generated by IoTDs. For the energy supply, APs typically contain high-power electrical devices that cannot be met by the limited energy output of sustainable resource sensors, while the energy requirements of IoTDs are much lower. To provide stable services while maintaining sustainability, IoTDs are assumed to only rely on sustainable energy and APs are powered by grid energy to provide robust energy support.

Authorized licensed use limited to: SOUTH CHINA UNIVERSITY OF TECHNOLOGY. Downloaded on December 22,2024 at 02:44:42 UTC from IEEE Xplore. Restrictions apply.

A. Communication Model

In the considered model, the wireless communication for task offloading between IoTDs and APs is based on NOMA, which facilitates simultaneous transmissions within the same spectrum frequency but introduces co-channel interference. We assume each AP operates on a unique spectrum frequency to eliminate co-channel interference between APs. For the multiple task uploading signals to the same AP, they are first integrated based on superposition coding(SC) and then transmitted. Denoting the set of IoTDs $\overline{\mathcal{N}}_m^t$ that send signals to AP *m* at time *t*, the received signal of AP *m* is given by

$$y_m = \sum_{n\in\overline{\mathcal{N}}_m^t} \Big(|h_{n,m}^t|^2/l_{n,m}\Big)\sqrt{p}S_n + a_n, \tag{1}$$

where *p* is the transmit power, S_n is the uploading signal, and a_n is the additive white Gaussian noise (AWGN) with variance σ^2. $|h_{n,m}^t|^2/l_{n,m}$ is the channel gain, in which $h_{n,m}^t$ is the small-scale Rayleigh fading and $l_{n,m}$ is the path loss. Note that in dynamic and complex wireless environments, Rayleigh fading is time-varying and difficult to accurately obtain through CSI estimation. Thus, following [26], [33], [34], we formulate the realistic Rayleigh fading channel gain as

$$h_{n,m}^t = \hat{h}_{n,m}^t + \epsilon_{n,m}^t, \tag{2}$$

where $\hat{h}_{n,m}^t \sim \mathcal{CN}(0, 1-\sigma_e^2)$ denotes the estimated Rayleigh fading obtained from the imperfect CSI estimation, and $\epsilon_{n,m}^t \sim \mathcal{CN}(0, \sigma_e^2)$ is a random variable to represent the estimation error.

After the overlapped signals arrive at the AP, SIC is employed to sequentially decode the signals in descending order of channel gain. Stronger signals are reconstructed and removed first, while weaker signals are extracted afterward. The weaker signals are considered as co-channel interference noise when decoding the stronger ones. Denoting the bandwidth as *B* and the weaker signal set that interfering signal from IoTD *n* as $\mathcal{N'}_{n,m}^t = \{n'|\forall n' \in \mathcal{N}, n' \neq n, \mu_{n'}^t = m, |h_{n',m}^t|^2/l_{n',m} < |h_{n,m}^t|^2/l_{n,m}\}$, the uplink data rate from IoTD *n* to AP *m* is calculated as

$$R_{n,m}^t = B\log_2\left(1 + \frac{p(|h_{n,m}^t|^2/l_{n,m})}{\sum_{n'\in\mathcal{N'}_{n,m}^t} p(|h_{n',m}^t|^2/l_{n',m}) + \sigma^2}\right). \tag{3}$$

When the achievable data rate $R_{n,m}^t$ falls lower than a threshold a^{th}, it is considered as a transmission outage. After simplifying the expression by ignoring subscript *t* and *m*, e.g., $\hat{h}_{n,m}^t$ to $\hat{h}_n$, the transmission failure probability given the estimated channel gain $\hat{h}_n$ is formulated as

$$P_{n,m}^{t,otg} = Pr\left\{B\log_2 \left(1 + \frac{p(|\hat{h}_n + \epsilon_n|^2/l_n)}{\sum_{n'\in\mathcal{N}_n'} p\Big(|\hat{h}_{n'} + \epsilon_{n'}|^2/l_{n'}\Big) + \sigma^2}\right) < a^{th}\right\}. \tag{4}$$

In Theorem 1 we derive the closed-form expression of $P_{n,m}^{t,otg}$.

Theorem 1: For the transmission outage probability described by (4) and let $\phi = 2^{a^{th}/B} - 1, \omega_n = (p\sigma_e^2)/(l_n\sigma^2), u_n = (2|\hat{h}_n|^2)/(\sigma_e^2)$, the approximate closed-form expression can be expressed as

$$P_{n,m}^{t,otg} = \begin{cases} F_{\Gamma\left(\frac{d_n}{2},2\right)}\left(\frac{\phi-\beta_n}{\alpha_n}\right), & \alpha_n > 0, \\ 1 - F_{\Gamma\left(\frac{d_n}{2},2\right)}\left(\frac{\phi-\beta_n}{\alpha_n}\right), & \alpha_n < 0, \end{cases} \tag{5}$$

where $F_{\Gamma(\cdot)}$ is the cumulative distribution function of the gamma distribution $\Gamma(\cdot)$, and

$$\alpha_n = \frac{\omega_n^3(2+3u_n^2) + \sum_{n'\in\mathcal{N}_n'}(-\phi\omega_{n'})^3(2+3u_{n'}^2)}{\omega_n^2(2+2u_n^2) + \sum_{n'\in\mathcal{N}_n'}(-\phi\omega_{n'})^2(2+2u_{n'}^2)},$$

$$\beta_n = \omega_n\Big(2+u_n^2\Big) + \sum_{n'\in\mathcal{N}_n'}(-\phi\omega_{n'})\Big(2+u_{n'}^2\Big) - \frac{\Big[\omega_n^2(2+2u_n^2) + \sum_{n'\in\mathcal{N}_n'}(-\phi\omega_{n'})^2(2+2u_{n'}^2)\Big]^2}{\omega_n^3(2+3u_n^2) + \sum_{n'\in\mathcal{N}_n'}(-\phi\omega_{n'})^3(2+3u_{n'}^2)},$$

$$d_n = \frac{\Big[\omega_n^2(2+2u_n^2) + \sum_{n'\in\mathcal{N}_n'}(-\phi\omega_{n'})^2(2+2u_{n'}^2)\Big]^3}{\Big[\omega_n^3(2+3u_n^2) + \sum_{n'\in\mathcal{N}_n'}(-\phi\omega_{n'})^3(2+3u_{n'}^2)\Big]^2}. \tag{6}$$

Proof: See the Appendix. ■

Here a task offloading policy is denoted by $\boldsymbol{\mu}^t = \{\mu_n^t | n \in \mathcal{N}\}$, where $\mu_n^t \in [0, M]$, with $\mu_n^t = m$ indicating the task of IoTD *n* is offloaded to AP *m* and $\mu_n^t = 0$ representing local processing. Therefore, for a task with data size y_n^t, the transmit delay and energy cost can be expressed as

$$d_{n,m}^{t,trans} = y_n^t / R_{n,m}^t, \tag{7}$$

$$e_{n,m}^{t,trans} = p d_{n,m}^{t,trans}. \tag{8}$$

B. Computation Model

Given the offloading decisions, tasks must then be processed either locally on IoTDs or remotely at APs. For local execution, we characterize the computation capacity of IoTD *n* as f_n CPU cycles per timeslot, and a proportion $\tau_n^t \in [0,1]$ of computing resources are allocated through dynamic voltage and frequency scaling (DVFS), the resulting computation delay and energy cost for local processing are computed as

$$d_n^{t,comp} = c_n^t / \big(\tau_n^t f_n\big), \tag{9}$$

$$e_n^{t,comp} = \kappa c_n^t \big(\tau_n^t f_n\big)^2, \tag{10}$$

where κ is the energy coefficient. Similarly, if the task arising from IoTD *n* is offloaded to the AP *m* with the computation capacity f_m, and the allocated portion $\tau_{n,m}^t$ of resources, the delay and energy cost are calculated as

$$d_{n,m}^{t,comp} = c_n^t / \big(\tau_{n,m}^t f_m\big), \tag{11}$$

$$e_{n,m}^{t,comp} = \kappa c_n^t \big(\tau_{n,m}^t f_m\big)^2. \tag{12}$$

Without loss of generality, we follow the assumption that task output size (i.e., downlink payload) is negligible compared to the task data size [21], [25]. Thus, combining the above communication model, the processing delay d_n^t,

Authorized licensed use limited to: SOUTH CHINA UNIVERSITY OF TECHNOLOGY. Downloaded on December 22,2024 at 02:44:42 UTC from IEEE Xplore. Restrictions apply.

renewable energy cost e_n^t, and the grid energy cost $e_{n,m}^t$ on AP *m* are computed as

$$d_n^t = \mathbb{1}\Big(\mu_n^t = 0\Big) d_n^{t,comp} + \sum_{m\in\mathcal{M}} \mathbb{1}\Big(\mu_n^t = m\Big)\Big(d_{n,m}^{t,trans} + d_{n,m}^{t,comp}\Big), \tag{13}$$

$$e_n^t = \mathbb{1}\Big(\mu_n^t = 0\Big) e_n^{t,comp} + \sum_{m\in\mathcal{M}} \mathbb{1}\Big(\mu_n^t = m\Big) e_{n,m}^{t,trans}, \tag{14}$$

$$e_{n,m}^t = \sum_{m\in\mathcal{M}} \mathbb{1}\Big(\mu_n^t = m\Big) e_{n,m}^{t,comp}, \tag{15}$$

where the indicator $\mathbb{1}(\cdot) = 1$ if the condition inside is true.

C. Energy Harvesting Model

We consider system scenarios where IoTDs are independently charged by renewable green energy sources. Each IoTD contains an energy harvester to facilitate the collection of renewable energy. Due to the unstable and intermittent nature of the harvested energy, batteries are used for storing energy to support subsequent task executions. Similar to existing works [35], we model renewable energy arrivals as a Bernoulli distribution, wherein each IoTD harvests *w* units of renewable energy at the beginning of each timeslot with a probability of *p* to recharge batteries. By b_n^t we denote the battery level of IoTD *n* at the beginning of *t*, the available energy after harvesting is

$$\hat{b}_n^t = \min b^{ub}, b_n^t + \psi_n^t w, \tag{16}$$

where b^{ub} is the battery capacity, and $\psi_n^t \sim \mathrm{Bern}(p)$ indicates whether energy is successfully harvested.

In a timeslot, if the IoTD chooses to process a task locally, it incurs energy consumption for local computation; otherwise, the device consumes energy to transmit the data to APs. In both cases, the battery level is updated as

$$b_n^{t+1} = \max\{0, \hat{b}_n^t - e_n^t\}. \tag{17}$$

As a different role, APs use the grid power supply to maintain uninterrupted high availability, which also causes brown energy consumption. Thus, minimizing the total energy costs from APs is one of the ultimate goals to protect the environment and enhance sustainability.

D. Reliability Metric

System reliability, which is related to the probability of task completion, faces significant challenges from both communication and computation perspectives. On the one hand, transmission outages during task offloading to APs could lead to incomplete reception of task packets. On the other hand, insufficient energy in IoTDs compromises their ability to support CPU operations for task processing and even causes the devices to go offline. In addition, high processing latency can result in the breach of delay constraints. These factors can lead to task failures and pose risks to the overall system stability. To quantitatively represent the reliability of edge systems, we construct the reliability metric from the above two perspectives based on our theorem with the aid of simulations.

From the communication perspective, we combine the theoretical estimation of successful transmission probability and actual transmission outages to represent transmission reliability. If an unsuccessful transmission occurs, the transmission reliability is denoted as 0; otherwise, it is indicated as $1 - P_{n,m}^{t,\mathrm{otg}}$ to penalize the probability of transmission failures. Additionally, if a task is processed locally, the transmission reliability is fixed at 1 since no transmission process is involved. Thus, the transmission reliability is calculated as

$$z_n^{t,trans} = \begin{cases} 1, & if\ \mu_n^t = 0, \\ 1 - P_{n,m}^{t,otg}, & if\ \mu_n^t \neq 0 \text{ and } R_{n,m}^t \geq a^{th}, \\ 0, & if\ \mu_n^t \neq 0 \text{ and } R_{n,m}^t < a^{th}. \end{cases} \tag{18}$$

In addition to transmission failures, insufficient energy availability and slow response also pose risks to system reliability. If the scheduling decisions result in energy shortages, or the task processing latency is larger than requirement q_n, we mark the task failed and set the computation reliability to 0. Otherwise, it should be set to 1. Formally, the computation reliability is formulated as

$$z_n^{t,comp} = \begin{cases} 0, & if\ \hat{b}_n^t - e_n^t < 0\ or\ d_n^t > q_n, \\ 1, & otherwise. \end{cases} \tag{19}$$

Thus the overall service reliability metric can be expressed as the product of transmission reliability and computation reliability, i.e.,

$$z_n^t = z_n^{t,trans} z_n^{t,comp}. \tag{20}$$

The metric integrates both the theoretical model and real-world feedback to provide a general and robust representation. When transmission failures or energy shortages occur in reality, the reliability metric is significantly impacted and marked as 0. Otherwise, a relatively minor penalty is applied to the probability of transmission failure based on the theoretical model. Therefore, despite the minor but inevitable gap between theory and reality, our metric can still effectively reflect the actual system reliability.

E. Problem Formulation

In this paper, we aim to reduce the grid energy cost of APs while ensuring system reliability by jointly optimizing the offloading decisions and computation resource allocation. The optimization problem is formulated as

$$\begin{aligned} \mathrm{P1}: \min_{\boldsymbol{\mu},\boldsymbol{\tau}^L,\boldsymbol{\tau}^{AP}} \quad & \lim_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T}\left(\sum_{m=1}^{M}\sum_{n=1}^{N} e_{n,m}^t\right) \\ \text{s.t.} \quad & \mathrm{C1}: \lim_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\big[z_n^t\big] \geq \psi_z, \forall n \in \mathcal{N}, \\ & \mathrm{C2}: 0 \leq \mu_n^t \leq M, \mu_n^t \in \mathbb{Z}, \\ & \mathrm{C3}: 0 \leq \tau_n^t \leq 1, \\ & \mathrm{C4}: 0 \leq \sum_{n=1}^{N} \tau_{n,m}^t \leq 1, \end{aligned} \tag{21}$$

where $\boldsymbol{\tau}^L = \{\tau_n^t | n \in \mathcal{N}\}$ and $\boldsymbol{\tau}^{AP} = \{\tau_{n,m}^t | n \in \mathcal{N}, m \in \mathcal{M}\}$ respectively refers to resource allocation decisions by

Authorized licensed use limited to: SOUTH CHINA UNIVERSITY OF TECHNOLOGY. Downloaded on December 22,2024 at 02:44:42 UTC from IEEE Xplore. Restrictions apply.

IoTDs and APs. C1 specifies that the long-term system reliability should be larger than ψ_z in expectation; C2 limits the offloading decision between local processing and offloading to one of the M APs; C3 constrains the local resource allocation proportion to between 0 and 1; C4 limits the sum of resource fractions allocated by each AP to [0, 1].

IV. Our Method

The objective function in (21) presents a non-trivial mixed-integer non-linear programming (MINLP) subject to time-averaged constraints. To facilitate efficient solutions with realistic complexity, we present our Reliable Offloading framework with Multi-Agent DRL (ROMA). In ROMA, we first simplify the problem by converting the time-average reliability constraint into an instantaneous (single-timeslot) objective through Lyapunov optimization. We then discretize resource allocation to avoid discrete-continuous hybrid action space and thereafter decompose the resource allocation of APs as a 0-1 knapsack problem to reduce action space. Multi-agent PPO is employed for decentralized decision-making about offloading and local resource allocation on every IoTD in the system.

A. Transform of Reliability Constraint

We begin with transforming the long-term constraint of system reliability. We rewrite *C1* into

$$\lim_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\big[\psi_z - z_n^t\big] \le 0, \forall n \in \mathcal{N}, \tag{22}$$

and introduce virtual queues $\boldsymbol{Q}^t = \{Q_n^t | n \in \mathcal{N}\}$, where $Q_n^1 = 0$, $Q_n^{t+1} = \max\{Q_n^t + (\psi_z - z_n^t), 0\}$.

Based on the Lyapunov Optimization theory, when the virtual queues are stable, the reliability constraint is satisfied. Thus, the constraint C1 can be expressed as

$$\mathrm{C1}': \lim_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}(Q_n^t) = 0, \forall n \in \mathcal{N}. \tag{23}$$

Besides, we define the Lyapunov function $L(\boldsymbol{Q}^t) = \frac{1}{2}\sum_{n=1}^{N} {Q_n^t}^2$ and Lyapunov drift $\Delta(\boldsymbol{Q}^t) = L(\boldsymbol{Q}^{t+1}) - L(\boldsymbol{Q}^t)$. The upper bound of Lyapunov drift can be derived as

$$\begin{aligned}\Delta(\boldsymbol{Q}^t) &= \frac{1}{2}\sum_{n\in\mathcal{N}} {Q_n^{t+1}}^2 - \frac{1}{2}\sum_{n\in\mathcal{N}} {Q_n^t}^2 \\ &\le \frac{1}{2}\sum_{n=1}^{N}(\psi_z - z_n^t)^2 + \sum_{n=1}^{N} Q_n^t(\psi_z - z_n^t) \\ &\le B + \sum_{n=1}^{N} Q_n^t(\psi_z - z_n^t),\end{aligned} \tag{24}$$

where B is a constant term representing the upper bound of $\frac{1}{2}\sum_{n=1}^{N}(\psi_z - z_n^t)^2$.

To minimize the original objective while ensuring stable queues, we leverage the drift-plus-penalty minimization approach, which aims to minimize the upper bound of $\Delta(\boldsymbol{Q}^t)$ and the objective function weighted by coefficient v for each timeslot. Thus, P1 is converted into P1$'$ as follow:

$$\begin{aligned}\mathrm{P1}': \min_{\boldsymbol{\mu},\boldsymbol{\tau}^L,\boldsymbol{\tau}^{AP}} \lim_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T}\Bigg[& v\sum_{m=1}^{M}\sum_{n=1}^{N} e_{n,m}^t \\ &+ \sum_{n=1}^{N} Q_n^t(\psi_z - z_n^t)\Bigg] \\ \text{s.t. }\; & \mathrm{C2}, \mathrm{C3}, \mathrm{C4}.\end{aligned} \tag{25}$$

Intuitively, the added term $Q_n^t(\psi_z - z_n^t)$ serves to penalize increases in the virtual queue length thus encouraging a shorter queue and a smaller gap between the current reliability and its expected value ψ_z.

B. Action Discretization

Offloading decisions are discrete while resource allocation requires continuous actions. The resulting hybrid action space significantly complicates DRL by making the learning process, algorithm design, and convergence more difficult. To this end, we propose to discretize the continuous resource allocation.

For local resource allocation, we discretize $\boldsymbol{\tau}^L$ into $\hat{\boldsymbol{\tau}}^L = \{\hat{\tau}_n^t | \hat{\tau}_n^t \in \{0,1\}\}$, where $\hat{\tau}_n^t = 1$ means the IoTD tries to allocating sufficient resources to complete the task and vice versa. Similarly, $\boldsymbol{\tau}^{AP}$ are discretized to $\hat{\boldsymbol{\tau}}^{AP} = \{\hat{\tau}_{n,m}^t | \hat{\tau}_{n,m}^t \in \{0,1\}\}$. Now we provide proposition 1, which proves that when $\hat{\tau}_n^t$ or $\hat{\tau}_{n,m}^t$ is feasibly decided, the corresponding resource allocation ratio can be optimally determined.

Proposition 1: Given feasible, determined $\hat{\tau}_n^t$ and $\hat{\tau}_{n,m}^t$, the optimal resource allocation ratio for IoTDs and APs can be determined as

$$\tau_n^t = \begin{cases} 0, & \textit{if } \hat{\tau}_n^t = 0, \\ \frac{c_n^t}{q_n f_n}, & \textit{otherwise}, \end{cases} \tag{26}$$

$$\tau_{n,m}^t = \begin{cases} 0, & \textit{if } \hat{\tau}_{n,m}^t = 0, \\ \frac{c_n^t}{(q_n - d_{n,m}^{t,trans}) f_n}, & \textit{otherwise}. \end{cases} \tag{27}$$

Proof: For local resource allocation at timeslot t, suppose we have two feasible solution $\boldsymbol{\tau}^L$ and $\boldsymbol{\tau}^{L*}$, where the difference is $\exists n \in \mathcal{N}, \tau_n^{t*} \le \tau_n^t$.

When $\mu_n^t \ne 0$, the value of τ_n^t has no impact on $e_{n,m}^{t,trans}$ and $e_{n,m}^{t,comp}$. Therefore, the aforementioned difference has no influence on the system. When $\mu_n^t = 0$, given that $e_{n,m}^{t,comp}$ monotonically increases with respect to τ_n^t, we can deduce that $e_{n,m}^{t,comp} \le e_{n,m}^{t,comp*}$, which leads to higher risk of energy insufficiency in the following timeslots. If it occurs at $\hat{t}$, we have $z_{n,m}^{\hat{t},comp} \le z_{n,m}^{\hat{t},comp*}$, ultimately resulting in worse performance. Thus, iteratively updating the current policy to $\boldsymbol{\tau}^{L*}$ with fewer local resource allocations leads to consistent or better results, and the optimal fraction is the lowest one that satisfies demands.

Therefore, when $\hat{\tau}_n^t = 0$, the optimal value of τ_n^t should be 0 for energy preservation. When $\hat{\tau}_n^t = 1$, to satisfy the delay constraint the minimum valid value of τ_n^t is $\frac{c_n^t}{q_n f_n}$.

Similarly for APs, allocating the exact amount of required resources leads to less $e_{n,m}^t$ without affecting Q_n^t and z_n^t. Thus, the optimal allocation policy is obtained based on the

Authorized licensed use limited to: SOUTH CHINA UNIVERSITY OF TECHNOLOGY. Downloaded on December 22,2024 at 02:44:42 UTC from IEEE Xplore. Restrictions apply.

binary decision $\hat{\tau}_{n,m}^t = 0$ and the task demand. When $\hat{\tau}_{n,m}^t = 0$, the optimal value of $\tau_{n,m}^t$ should be 0. And when $\hat{\tau}_n^t = 1$, the delay constraint is expected to be satisfied, hence the minimum valid value of τ_n^t is $\frac{c_n^t}{q_n f_n}$. ∎

C. Task Offloading and Resource Allocation

From a practical perspective, task offloading and local resource allocation decisions should be autonomously made by each IoTD, whereas APs should independently determine their own resource allocation. Mathematically, the former decisions critically influence the time-evolving remaining energy and the virtual queue length of IoTDs that are directly linked to the long-term optimization goal. Conversely, the resource allocation decisions by APs have a relatively minor impact on future objectives. This distinction naturally leads to the decomposition of these problems. Hence, we decompose problem P1′ into two sub-problems, that is, P2 which is the optimization of task offloading and local resource allocation by IoTDs for long-term performance, and P3 which directs APs in resource allocation that focuses on the current timeslot.

$$\begin{aligned} \mathrm{P2}: \min_{\boldsymbol{\mu}, \hat{\boldsymbol{\tau}}^L} \ & \lim_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T}\Bigg[\sum_{m=1}^{M}\sum_{n=1}^{N} ve_{n,m}^t \\ & + \sum_{n=1}^{N} Q_n(t)\big(\psi_z - z_n^t\big)\Bigg] \\ \text{s.t.} \ & \mathrm{C2}, \\ & \mathrm{C5}: \hat{\tau}_n^t \in \{0,1\}, \\ & \mathrm{C6}: \tau_n^t \le 1, \forall n \in \mathcal{N}. \end{aligned} \tag{28}$$

$$\begin{aligned} \mathrm{P3}: \min_{\hat{\tau}_{n,m}^t} \ & \sum_{m=1}^{M}\sum_{n=1}^{N} v\hat{\tau}_{n,m}^t e_{n,m}^{t,comp} + \sum_{n=1}^{N} Q_n^t(\psi_z - z_n^t) \\ \text{s.t.} \ & \mathrm{C7}: \hat{\tau}_{n,m}^t \in \{0,1\}, \\ & \mathrm{C8}: \sum_{n=1}^{N} \hat{\tau}_{n,m}^t \tau_{n,m}^t \le 1, \forall m \in \mathcal{M}. \end{aligned} \tag{29}$$

We first discuss the sub-problem P3 of allocating resources for APs. Since each IoTD task can only be offloaded to one AP or processed locally and that $z_n^{t,trans}$ is fixed given the determined offloading decisions, we can rewrite $\sum_{n=1}^{N} Q_n(t)(\psi_z - z_n^t)$ as $\sum_{m=1}^{M}\sum_{n=1}^{N} \mathbb{1}(\mu_n^t = m) Q_n(t)(\psi_z - z_n^{t,trans}\hat{\tau}_{n,m}^t)$, where the missing terms about local processing is a constant term here and can be ignored. Thus, the joint optimization problem can be split into several independent problems, where each AP *m* focuses on the following problem:

$$\begin{aligned} \mathrm{P3_m}: \min_{\hat{\tau}_{n,m}^t} \ & \sum_{n=1}^{N} v\hat{\tau}_{n,m}^t e_{n,m}^{t,comp} \\ & + \mathbb{1}(\mu_n^t = m) Q_n^t(\psi_z - z_n^{t,trans}\hat{\tau}_{n,m}^t) \\ \text{s.t.} \ & \mathrm{C7}, \mathrm{C8}. \end{aligned} \tag{30}$$

For locally processed tasks or the tasks offloaded to AP other than *m*, we have $\mathbb{1}(\mu_n^t = m) = 0$, allocating $\hat{\tau}_{n,m}^t = 0$ can obviously minimize the objective function. For tasks that failed in transmission, we have $z_n^{t,trans} = 0$ that makes the latter item constant, thus the optimal decision is $\hat{\tau}_{n,m}^t = 0$. For the task set $N_m' = \{n' | n' \in \mathcal{N}, \mathbb{1}(\mu_n^t = m) = 1, R_{n,m}^t \ge a^{th}\}$, in which tasks successfully arrive at AP *m*, the problem can be further transformed into the following $\mathrm{P3_m'}$, where the constant term $Q_n(t)\psi_z$ is ignored:

$$\begin{aligned} \mathrm{P3_m'}: \min_{\hat{\tau}_{n,m}^t} \ & \sum_{n\in\mathcal{N}_m'} \hat{\tau}_{n,m}^t (ve_{n,m}^{t,comp} - Q_n^t(1 - P_{n,m}^{t,otg})) \\ \text{s.t.} \ & \mathrm{C7}, \mathrm{C8}. \end{aligned} \tag{31}$$

Obviously, the problem $\mathrm{P3_m'}$ is a 0-1 knapsack problem that could be solved by individual APs in a decentralized manner. Due to the disruption of transmission outages, only a limited number of tasks could be successfully offloaded to the same AP at the same time, so the complexity of the problem is minimal. Existing methods such as exhaustive search and dynamic programming (DP) are well established to solve this problem within a short delay.

Remark 1: With exhaustive search, the time complexity is $\mathcal{O}(2^{|\mathcal{N}_m'|})$. As an alternative, DP can achieve a linear time complexity but is only suitable for scenarios involving integer knapsack capacity and item weights. Thus, when $|\mathcal{N}_m'|$ is relatively small, exhaustive searching can be chosen. When $|\mathcal{N}_m'|$ becomes larger, DP can still be utilized by scaling and rounding up relative terms to achieve an integer space.

Subject to the remaining energy and the reliability virtual queue length of IoTDs, the optimization problem formulated in P2 is a long-term problem, where decisions will have an impact on the subsequent states. Traditional methods with local optimization insurance cannot guarantee long-term optimal solutions. Fortunately, the recent development of DRL shows advanced performances in such a long-term optimization problem. The model-free characteristic also allows DRL to update its policy through interactions with the real environment rather than relying solely on the theoretical model, which provides generalizability to real-world scenarios. In addition, since the decentralized nature of edge computing diverges from centralized DRL approaches, the multi-agent DRL algorithm appears to be a more appropriate approach, where each IoTD behaves as an independent agent to access local observations for decentralized execution. The prerequisite for adopting multi-agent DRL is to formulate the problem as a POMDP, represented by a tuple $< \mathcal{O}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma >$, where $\mathcal{O}$ is the partial observation space of agents. $\mathcal{A}$ is the joint action space. $\mathcal{P}$ is the state transition probability. $\mathcal{R}$ is the reward of taking actions and γ is the discount factor. In this system, the observation, action, and reward are defined as follows.

1) Observation: each agent can only observe the part of the system state related to itself. We formulate the observation of IoTD *n* at timeslot *t* as $\boldsymbol{o}_n^t = \{y_n^t, c_n^t, q_n^t, \hat{\boldsymbol{h}}_n^t, b_n^t, Q_n^t\}$, where y_n^t, c_n^t, q_n^t are task informations, $\hat{\boldsymbol{h}}_n^t = \{\hat{h}_{n,m}^t | m \in \mathcal{M}\}$ is the estimated Rayleigh fading channel gain. b_n^t indicates the available energy and Q_n^t is the length of the virtual queue. The global system state is the aggregate of observations, i.e., $\boldsymbol{s}^t = (\boldsymbol{o}_1^t, \ldots, \boldsymbol{o}_N^t)$.
2) Action: The agent of IoTD *n* should make task offloading decision μ_n^t and local computing resource allocation

Authorized licensed use limited to: SOUTH CHINA UNIVERSITY OF TECHNOLOGY. Downloaded on December 22,2024 at 02:44:42 UTC from IEEE Xplore. Restrictions apply.

$\hat{\tau}_n^t$ at timeslot t. Since resource allocation only has impacts when for local processing, we pack them into one action encoded as a_n^t, where $a_n^t = 0$ and $a_n^t = M + 1$ respectively refers to $\hat{\tau}_n^t = 0$ or $\hat{\tau}_n^t = 1$ with $\mu_n^t = 0$, and $1 \le a_n^t \le M$ matches $1 \le \mu_n^t \le M$ for offloading. The joint action is defined as $\boldsymbol{a}^t = (a_1^t, \ldots, a_N^t)$.

3) Reward: After taking joint action $\boldsymbol{a}^t$, the step reward is obtained to guide policy update. Since the target of DRL is to maximize the cumulative reward, we set the reward as the negative objective function, which is calculated as

$$R^t\big(\boldsymbol{s}^t, \boldsymbol{a}^t\big) = -\sum_{m=1}^{M}\sum_{n=1}^{N} ve_{n,m}^{t,comp} - \sum_{n=1}^{N} Q_n(t)\big(\psi_z - z_n^t\big). \quad (32)$$

Before presenting our ROMA, we start with the necessary preliminaries. For a multi-agent DRL system, the ultimate goal is to learn a joint policy $\boldsymbol{\pi}(\boldsymbol{a}|\boldsymbol{o}) = \prod_{n=1}^{N} \pi_n^{\theta_n}(a_n|o_n)$ that maximize the expected discounted cumulative reward $J(\boldsymbol{\pi}) = \mathbb{E}_{\boldsymbol{\pi}}[\sum_{t=0}^{\infty} \gamma^t R_t]$, where θ_n is the policy parameter of agent n. The value function $V_{\boldsymbol{\pi}}(\boldsymbol{s})$ denotes the cumulative reward under a given policy starting at state $\boldsymbol{s}$, the state-action function $Q_{\boldsymbol{\pi}}(\boldsymbol{s}, \boldsymbol{a})$ refers to the obtainable future return by taking joint action $\boldsymbol{a}$ in the state $\boldsymbol{s}$, and the advantage function $A_{\boldsymbol{\pi}}(\boldsymbol{s}, \boldsymbol{a})$ represents how better the joint action $\boldsymbol{a}$ compared to mean action, is defined as

$$V_{\boldsymbol{\pi}}(\boldsymbol{s}) = \mathbb{E}_{\boldsymbol{\pi}}\left[\sum_{i=t}^{\infty} \gamma^{i-t} R_i | \boldsymbol{s}_t = \boldsymbol{s}\right], \quad (33)$$

$$Q_{\boldsymbol{\pi}}(\boldsymbol{s}, \boldsymbol{a}) = \mathbb{E}_{\boldsymbol{\pi}}\left[\sum_{i=t}^{\infty} \gamma^{i-t} R_i | \boldsymbol{s}_t = \boldsymbol{s}, \boldsymbol{a}_t = \boldsymbol{a}\right], \quad (34)$$

$$A_{\boldsymbol{\pi}}(\boldsymbol{s}, \boldsymbol{a}) = Q_{\boldsymbol{\pi}}(\boldsymbol{s}, \boldsymbol{a}) - V_{\boldsymbol{\pi}}(\boldsymbol{s}). \quad (35)$$

Considering the characteristics of decision-making in discrete action space, we base our method on the multi-agent proximal policy optimization algorithm (MAPPO) [36].

In general, the MAPPO algorithm adopts a centralized training and decentralized execution (CTDE) framework following the actor-critic paradigm. Each IoTD is treated as an independent agent to select actions based on their local observation, and a global critic network is trained to evaluate current state values using the aggregated system state. During training, the actor networks are updated to optimize policies to maximize cumulative rewards, and the critic learns to accurately predict future rewards. Once trained, decentralized execution allows each agent to operate autonomously without relying on the centralized critic.

Specifically, the global critic network $V_{\boldsymbol{\pi}}^{\omega}$ with parameter ω is learned to predict the state function $V_{\boldsymbol{\pi}}$ for each state $\boldsymbol{s}$. Based on the Bellman Equation, the critic network can be iteratively updated by minimizing the following temporal-difference(TD) loss under the sampled trajectory ρ:

$$L(\omega) = \mathbb{E}_{(\boldsymbol{s}^t, \boldsymbol{a}^t, R^t, \boldsymbol{s}^{t+1}) \sim \rho}\left[\Big(V_{\boldsymbol{\pi}}^{\omega}(\boldsymbol{s}^t) - (R^t + \gamma V_{\boldsymbol{\pi}}^{\omega}(\boldsymbol{s}^{t+1}))\Big)^2\right]. \quad (36)$$

The objective of actor networks is to maximize the probability of taking action with a higher cumulative reward, which can be estimated by the advantage function value. Thus, the loss function of actor networks to be minimized is the negative of objective, which is formulated as

$$L(\theta_n) = -\mathbb{E}_{\rho}\Big[\min\Big(\eta_n(\theta_n)A^t, \operatorname{clip}(\eta_n(\theta_n), 1-\epsilon, 1+\epsilon)A^t\Big)\Big], \quad (37)$$

where $\eta_n(\theta_n) = \pi^{\theta_n}(a_n|o_n)/\pi_{old}^{\theta_n}(a_n|o_n)$ is the importance sampling ratio between updating policy and sampling policy, to enable multiple updates with single sampled trajectory. The function $\operatorname{clip}(\eta_n(\theta_n), 1-\epsilon, 1+\epsilon)$ controls $\eta_n(\theta_n)$ inside the interval $[1-\epsilon, 1+\epsilon]$ to avoid excessive update. The advantage function value can be obtained from the state value function estimated by the critic network through the generalized advantage estimator (GAE) [37] as

$$A^t = \sum_{l=0}^{h} (\gamma\lambda)^l \Big(R^t + \gamma V_{\pi}^{\omega}(\boldsymbol{s}^{t+l+1}) - V_{\pi}^{\omega}(\boldsymbol{s}^{t+l})\Big), \quad (38)$$

where h is the length of trajectory, λ is the hyper-parameter of GAE. Thus, θ_n can be updated via gradient descend $L(\theta_n)$.

Fig. 2 illustrates the overall framework of the proposed ROMA and Algorithm 1 shows the pseudo-code. We divide the operational time into episodes. In each episode, ROMA initially performs decentralized execution to interact with the environment and collect transitions over T timeslots. These data are subsequently used for centralized policy updates. Specifically, Lines 5-17 present the decentralized execution stage. In Lines 6-10, each IoTD collects observations and selects an action with its own deployed actor network in parallel at each timeslot. The action is then recovered and executed. From Lines 11 to 14, each AP independently collects information about the arrived tasks and solves the constructed 0-1 knapsack problem to allocate computational resources. Rewards are collected and transitions are stored for training (Lines 15-16). Note that these operations are fully decentralized. Each IoTD and AP focuses only on a small decision sub-problem. Consequently, ROMA is suitable for large-scale edge computing systems. After interactions, Lines 19-23 detail the centralized training stage. Transitions are collected to calculate advantage values, which helps to update the actor and critic networks.

D. Complexity Analysis

Since the N agents take actions in parallel, the total complexity of inference for task offloading and local resource allocation is primarily determined by the structure of the applied neural network for the actor.

Based on the given observation definition, the input size is $M + 5$, and the output size is $M + 2$, where each action corresponds to a log probability output. Assuming the number of neurons in each hidden layer is H and the number of hidden layers is K, and taking into account the activation function, the complexity of task offloading and local resource allocation can be expressed as $\mathcal{O}((M+5)H + KH^2 + (M+2)H + KH)$. Similarly, the resource allocation on APs can also be executed

Authorized licensed use limited to: SOUTH CHINA UNIVERSITY OF TECHNOLOGY. Downloaded on December 22,2024 at 02:44:42 UTC from IEEE Xplore. Restrictions apply.

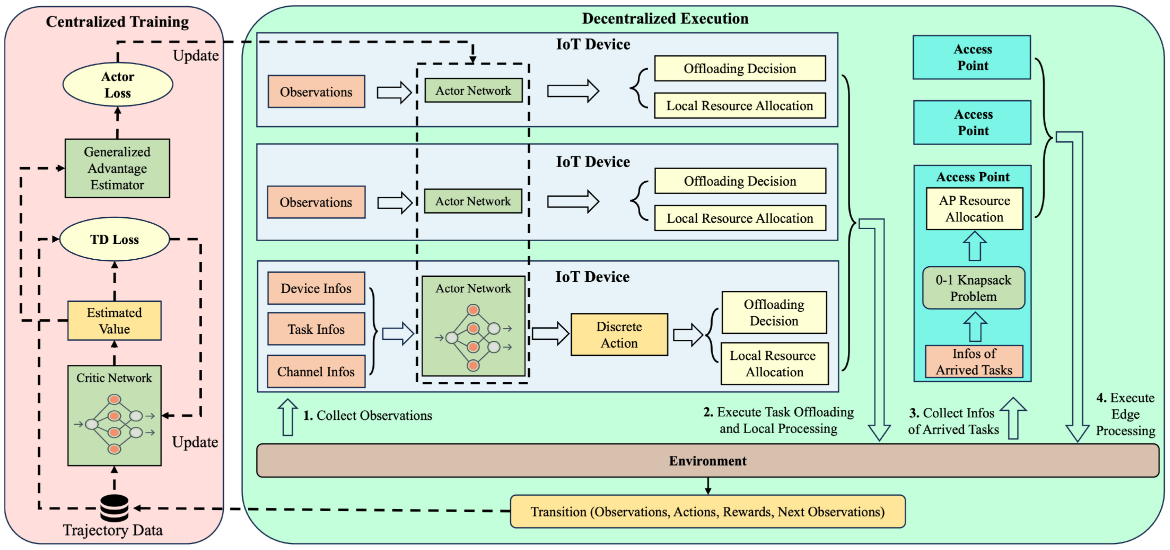

Fig. 2. The framework of ROMA. It follows the centralized-training-decentralized-execution paradigm. During the execution stage, IoTDs make decisions autonomously with their actor network. APs manage and allocate resources by solving the 0-1 knapsack problem based on the information of arrived tasks. A centrally deployed critic network assists with policy updates based on the collected interaction transitions only at the training stage.

Algorithm 1 ROMA

```
1: Randomly initialize actor network parameters $\theta_n$ for each IoTD and critic network parameters $\omega$. Set learning rate and hyper-parameters $\Delta, \psi_z, \lambda, \gamma$;
2: Initialize the length of the virtual queue and the available remaining energy;
3: for $episode = 1, 2, \dots, E$ do
4:   \* Decentralized Execution Stage. * \
5:   while $t < T$ do
6:     for $n \in \mathcal{N}$ in parallel do
7:       Collect observation $o_n^t$;
8:       Sample an action $a_n^t$ from $\pi_n(a_n^t|o_n^t)$, and recover it into $\mu_n^t$ and $\hat{\tau}_n^t$
9:       Obtain $\tau_n^t$ based on (26) and execute $\mu_n^t$ and $\tau_n^t$;
10:    end for
11:    for $m \in \mathcal{M}$ in parallel do
12:      For $n \in \mathcal{N}'_m$, find the optimal $\hat{\tau}_{n,m}$ by solving (31), otherwise allocate $\hat{\tau}_{n,m}^t = 0$;
13:      Obtain $\tau_{n,m}$ from (27) and execute it;
14:    end for
15:    Calculate reward $R^t$;
16:    Store transition $(\boldsymbol{s}^t, \boldsymbol{a}^t, R^t)$ to the trajectory $\rho$;
17:  end while
18:  \* Centralized Training Stage. * \
19:  Calculate Advantage function value by (38);
20:  for $n \in \mathcal{N}$ in parallel do
21:    Update actor network by gradient descent (37);
22:  end for
23:  Update critic network by gradient descent (36);
24: end for
```

in parallel. Since the upper bound of $|\mathcal{N}'_m|$ is N, assuming the scaling up factor is S, the complexity of DP becomes $\mathcal{O}(NS)$. Therefore, the overall complexity of the algorithm execution is the sum of the complexities of each DRL agent and that on each AP, i.e., $\mathcal{O}(H(2M + 7 + (H + 1)K) + NS)$.

During the training phase, the critic network plays a crucial role in the learning process. It takes inputs of size $N(M + 5)$ and produces an output of size 1. Considering a critic network with J hidden layers, each consisting of L neurons, the computational complexity can be expressed as $\mathcal{O}(L(N(M+5)+LJ+1))$. In each training episode spanning C timeslots, several interactions occur. Since the complexity of the Generalized Advantage Estimation (GAE) algorithm is proportional to C, and considering the training process over U episodes, the total complexity becomes $\mathcal{O}(U(C + 1)(H(2M + (H + 1)K) + NS) + UC + UL(N(M + 5) + LJ + 1))$.

V. Experiment

A. Experimental Setup

We built a simulated environment using Python 3.9 and Pytorch 1.12 on a workstation powered by Intel Core i9-10900K and Quadro RTX 4000. The simulation system encompasses a square 60m $\times$ 60m area. The default settings include 3 APs and 10 IoTDs uniformly distributed within the area. During the transmission phase, all IoTDs utilize a fixed transmit power of 100 mW, and each AP operates on an orthogonal channel with a bandwidth of 3 MHz. The parameter σ_e^2 representing the CSI estimation error is set to 0.1, and the transmission rate threshold a^{th} is set to 3. The path loss associated with transmission distance follows $140.7 + 36.7 \log_{10} d$. Uniform distributions are employed to simulate the task data size, required CPU cycles, and latency requirements. More parameters are listed in Table II.

During the training process, specific training parameters are configured as follows. Actor networks are designed with three layers, while the critical network comprises four layers. The dimension of hidden layers in both networks are set to 128. Tanh activation function is employed to optimize performance.

Authorized licensed use limited to: SOUTH CHINA UNIVERSITY OF TECHNOLOGY. Downloaded on December 22,2024 at 02:44:42 UTC from IEEE Xplore. Restrictions apply.

TABLE II
PARAMETER SETTINGS OF SIMULATION

Notations	Descriptions	Value
y	Task data size	$\mathcal{U}(0.4, 0.5)$ MB
c	Task required CPU cycles	$\mathcal{U}(0.4, 0.5)$ GCycles
q	Task delay requirement	$\mathcal{U}(0.7, 0.9)$ timeslots
b^{ub}	Capacity of IoTD battery	2 J
p	Probability of green energy arrivals	0.4
w	Amount of harvested energy	0.2 J
κ	Energy coefficient	10^{-27}
v	Lyaponov coefficient	1
ψ_z	Target reliable expectation	0.92
f_n	CPU Frequency of IoTDs	0.8 GHz
f_m	CPU Frequency of APs	4 GHz
σ^2	Additive white Gaussian noise	-114 dBm/MHz

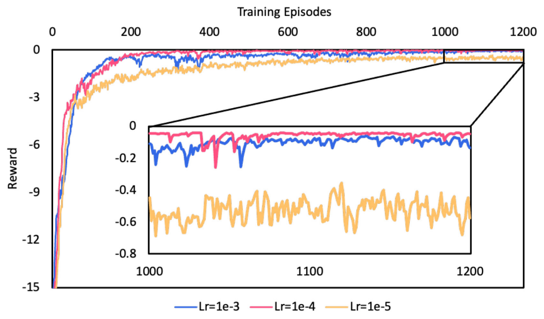

Fig. 3. Convergence analysis for ROMA under different learning rate. Exponential moving averaging is applied for clearer visualization.

Furthermore, the discount factor γ is set to 0.98, while the importance sampling ratio ϵ is assigned a value of 0.2. The hyper-parameter λ of GAE is set to 0.98. The learning rate is set to 1e$-$4. The model is trained for 1200 episodes with 256 timeslots in each episode. The update process is executed after every 1024 interactions with 10 data reuse times. The mini-batch size is set to 256. Adam optimizer with $\epsilon = 1\text{e}{-5}$ is applied. The model with the best performance was tested with 100 episodes of simulation, and the mean value of each metric was taken as the final result.

B. Hyperparameter Analysis

In this section, we first analyze the convergence of our strategy by examining its performance under various learning rates. We then investigate the impact of different Lyapunov coefficients v and target reliable expectations ψ_z on the system performance.

Learning Rate: Fig. 3 shows the system reward under three different learning rate settings. Note that the ceiling is zero in our negative rewarding design. The blue line corresponds to a high learning rate of 1e$-$3 which enables relatively fast convergence, reaching the plateau at around 200 episodes. However, large learning rates also result in overshooting and oscillations in subsequent stages, impeding reliable improvements. Conversely, when the learning rate is set to 1e$-$5 (yellow line), the convergence speed noticeably decreases and the model gets trapped in a local optimum. Therefore, in our learning scenario, a learning rate of 1e$-$4 is appropriate to achieve the best performance.

Lyaponov Coefficient: Fig. 4 shows the algorithm's performance concerning the grid energy cost of APs and the

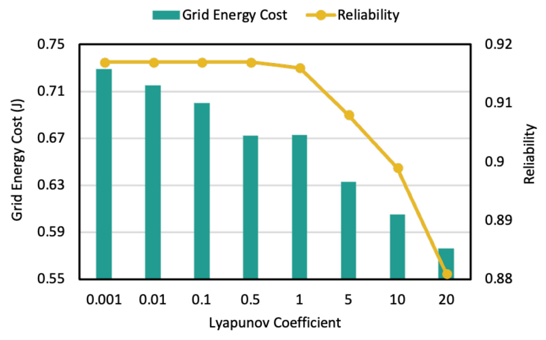

Fig. 4. Grid energy cost and reliability using different Lyapunov coefficients.

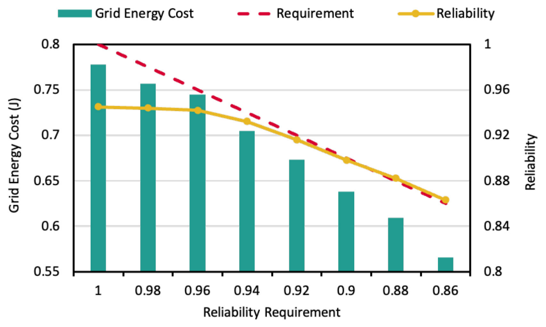

Fig. 5. Grid energy cost and reliability given different long-term reliability expectation requirements.

achieved reliability under different settings of the Lyapunov coefficient v. The value controls the trade-off between grid energy costs and reliability. Intuitively, results show that higher values of v prioritize grid energy preservation over the reliability requirement. As v increases from 1 to 20, the overall reliability degrades from 0.916 to 0.881, whilst the grid energy cost decreases from 0.673 to 0.576. This is typically achieved by learning to selectively drop some computation-intensive tasks. To balance between grid energy costs and reliability, we suggest adopting a Lyapunov coefficient of $v = 1$.

Reliability Requirement: Fig. 5 illustrates the impact of different long-term reliability expectation requirements ψ_z on the system. To encourage the optimization of grid energy costs, the reward function is designed to stop providing rewards once the reliability reaches the specified required value. Experimental results confirm the effectiveness of this design. When the reliability required values are set below 0.92, the obtained reliability metrics closely approximate the set value, and significantly reduce system energy costs. However, for larger required value, the algorithm faces challenges in effectively improving system reliability due to limitations in the ability of the simulated edge computing system. As an example, when the requirement is raised from 0.96 to 1, the grid energy consumption exhibits a linear increase, reaching a value of 0.778, while the actual achieved reliability only experiences a slight improvement, rising from 0.942 to 0.945. Therefore, we adopt $\psi_z = 0.92$ as the default setting.

C. Performance Evaluation

In this section, we compare and analyze our proposed method with two widely-used baseline methods and three

Authorized licensed use limited to: SOUTH CHINA UNIVERSITY OF TECHNOLOGY. Downloaded on December 22,2024 at 02:44:42 UTC from IEEE Xplore. Restrictions apply.

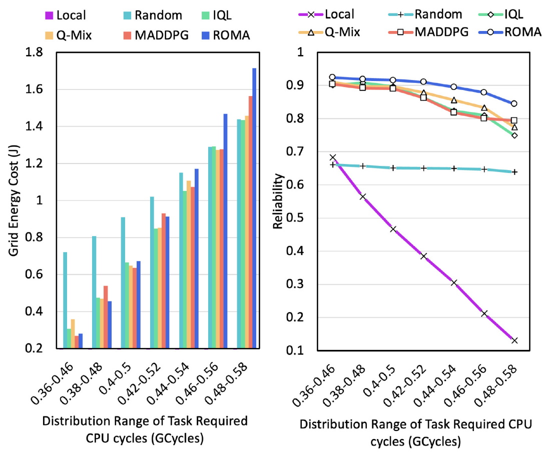

Fig. 6. Grid energy costs and reliability under different settings of task CPU cycles requirements. Local computing incurs no grid energy cost.

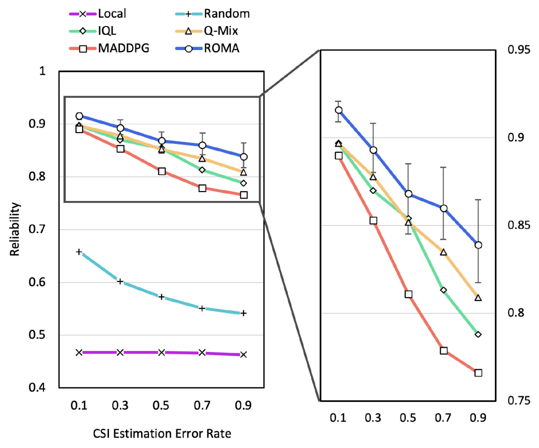

Fig. 7. Achieved reliability under different CSI estimation error rate settings.

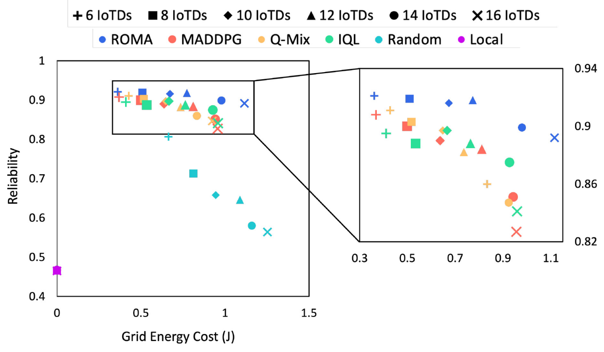

Fig. 8. Grid energy costs and reliability in environments of different number of IoTDs. IoTD population settings are distinguished by the shape of markers, and the results of different methods are rendered in different colors.

commonly used DRL algorithms under various environment settings. The baseline methods are specified below:

1) Local Computing: all tasks are executed locally and no offloading operation is performed.
2) Random: IoTDs make random decisions on task offloading to one of the edge servers or local execution.
3) Independent Q-Learning (IQL): IQL is a decentralized DRL algorithm in which multiple agents learn independently. Each agent maintains its own Q-network to make decisions based on self-observations. This method is employed in [38].
4) Q-Mix: Q-Mix follows the CTDE paradigm, where a centralized mixture network combines individual Q-values and global states to evaluate actions and guide updates. It is applied in [19].
5) MADDPG: MADDPG is a CTDE-based DRL algorithm tailored for multi-agent scenarios. Decentralized agents learn deterministic policies, which enables agents to effectively learn continuous actions, facilitating coordination and cooperation in complex environments. This method is widely used in the field [39].

Task intensity: We also compare different approaches under different distributions of task CPU cycles. As shown in Fig. 6, simple local computing (no offloading), yields poor reliability that drops from 0.683 to 0.13 as IoTDs constantly get short of energy in their batteries. Random decision-making somehow avoids local device energy shortage but still fails to attain a healthy reliability level due to stochastic failures in transmission. In comparison to other DRL methods, our approach exhibits lower grid energy costs when tasks require fewer CPU cycles, while ensuring higher reliability through more rational scheduling decisions under high load of computation. This empirically proves that ROMA is energy-efficient to go with strong reliability promises.

Channel Uncertainty: Fig. 7 visualizes the performance evaluation under different levels of channel uncertainty represented by the CSI estimation error rate σ_e^2. For fair comparison, we provide the range of results obtained from multiple test episodes of our method. Our first observation is that all DRL-based approaches consistently demonstrate significantly better performance, showing ability to learn complex policies. ROMA consistently offers superior reliability compared to other approaches, even in the worst case (lower bound of the bars). Our method exhibits the slightest decline in reliability as the CSI estimation error increases. In ideal situations where the CSI estimation error rate σ_e^2 is small, both IQL and Q-Mix have similar performance. However, as the error increases, the performance of IQL drops notably due to the absence of centralized training in IQL, which limits the ability of agents to effectively collaborate in complex, uncertain environments. Besides, we notice that MADDPG shows the worst performance among DRL-based approaches. We attribute this to the fact that MADDPG is not designed for discrete action spaces. After generating action probabilities, sampling is required. This discrepancy between probability values and actual actions makes deterministic policy updates inaccurate, leading to the performance gap.

System Scale: In order to compare different strategies in environments of varied scales, we experimented in multiple environments with different numbers of IoTDs. Results in Fig. 8 indicate that the Local method (overlapped purple markers) cannot guarantee service reliability, with task success rate staying below 0.5. Random offloading exhibits a sharp decrease of the metric as the number of IoTDs increases. This boils down to stronger resource contention and

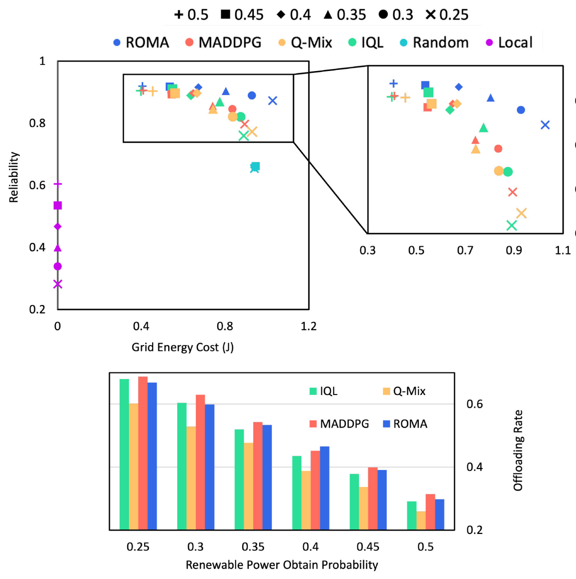

Fig. 9. Grid energy costs, reliability, and offloading rates under varied probabilities of renewable energy arrivals. 6 settings of renewable energy arrival were explored (shown in different marker shapes). Offloading rates for the *Local* and *Random* methods are around 0 and 0.75 respectively and thus excluded for clarity.

co-channel interference. In comparison to other DRL methods, the proposed ROMA shows advantages in the balance between reliability and energy efficiency at all scale — the performance of ROMA is consistently on the Pareto frontier with strong reliability guarantee in all cases. With an increasing number of IoTDs, the performance points of our method scatter from the upper left to the upper right, while other strategies experience notable drop in service reliability. This is because their insufficiency in learning to avoid both critical low battery and "risky" offloading decisions, resulting in unreliable decisions, lower AP utilization, and a significant reliability decrease.

Renewable Energy Supply: Fig. 9 provides the performance comparison in different conditions of renewable energy availability, demonstrating the effectiveness of the proposed method in systems with sufficient and insufficient renewable energy supply. In the situation where green energy is scarce (e.g., $p = 0.25$), DRL-based algorithms have high task offloading ratio to minimize local battery usage (shown in the bar chart). However, the scatter plot shows that popular methods like IQL and QMix struggle to effectively maintain reliable transmission, resulting in decreased system reliability that is approximately 10% to 20% lower than our ROMA. By contrast, in the cases where renewable energy is relatively ample, ROMA optimizes the utilization of IoTD batteries and ensures high reliability without paying the price for more grid energy consumption.

VI. Conclusion

Reliability can be critical in edge computing systems. It is of great challenge to achieve that with uncertain communication channel conditions and unstable renewable energy supply. In this paper, we present ROMA, a reliable approach based on multi-agent DRL, to ensure high service reliability via computation offloading. We first derive the probability of NOMA transmission failure given imperfect CSI estimation and combine it with the battery factors to define a comprehensive reliability metric. We employ Lyapunov optimization to ensure long-term reliability in a discretized space of actions. Our solution exploits Multi-agent DRL and 0-1 knapsack problem solving to jointly optimize task offloading and resource allocation. Experimental results demonstrate that our proposed approach outperforms various existing algorithms in terms of reliability and grid energy costs, showing strong promises in NOMA-based sustainable edge systems where reliability is a major concern. However, optimizing service reliability solely through offloading actions is a challenging task. As part of future work, we plan to explore proactive methods and consider variable constraints.

Appendix

Proof of Theorem 1

Before presenting the proof of the theorem, it is necessary to provide the following lemma, which offers an estimation of the weighted sum of random variables with non-central chi-squared distributions.

Lemma 1: For random variables A_r following non-central chi-squared distributions $\chi_2^2(u_r^2)$, the weighted sum $S = \sum_{r=1}^q C_r A_r$ can be approximated as a random variable taking the form $R = \alpha\Gamma(d/2, 2) + \beta$, where $\Gamma(\cdot)$ refers to the gamma distribution, and

$$\alpha = \frac{\sum_{r=1}^q C_r^3(2+3u_r^2)}{\sum_{r=1}^q C_r^2(2+2u_r^2)},$$
$$\beta = \sum_{r=1}^q C_r\left(2+u_r^2\right) - \frac{\sum_{r=1}^q C_r^2(2+2u_r^2)}{\sum_{r=1}^q C_r^3(2+3u_r^2)},$$
$$d = \frac{\left(\sum_{r=1}^q C_r^2(2+2u_r^2)\right)^3}{\left(\sum_{r=1}^q C_r^3(2+3u_r^2)\right)^2}. \tag{39}$$

Proof: According to [40], the approximation $S \approx R$ is valid if the cumulants $\mathcal{K}_l(S), l = 1, 2, \dots,$ of S are equal to the cumulants $\mathcal{K}_l(R)$ of $R = \alpha\chi_d^2 + \beta$. From the cumulants generating function(CGF) of non-central chi-squared distributions and the properties of cumulants, simple algebra gives that

$$\mathcal{K}_l(S) = 2^{l-1}(l-1)! \sum_{r=1}^q C_r^l\left(2+lu_r^2\right),$$
$$\mathcal{K}_1(R) = \alpha d + \beta,$$
$$\mathcal{K}_l(R) = 2^{l-1}(l-1)!\alpha^l d, l = 2, 3, \dots, \tag{40}$$

By letting the first three cumulants of T and R be equal, i.e., $\mathcal{K}_l(S) = \mathcal{K}_l(R), l = 1, 2, 3,\ \alpha, \beta, \delta$ can be determined as described in (39). Besides, since d is often not an integer, R can be represented in the form of the gamma distribution, i.e., $T \approx R = \alpha\Gamma(d/2, 2)\ + \beta$. ■

Authorized licensed use limited to: SOUTH CHINA UNIVERSITY OF TECHNOLOGY. Downloaded on December 22,2024 at 02:44:42 UTC from IEEE Xplore. Restrictions apply.

Defining $\phi = 2^{a^{th}/B} - 1$, $\omega_n = (p\sigma_e^2)/(l_n\sigma^2)$, and $u_n = (2|\hat{h}_n|^2)/(\sigma_e^2)$ as described in the theorem, the original expression can be transformed into the following probability:

$$P_{n,m}^{t,otg} = Pr\left\{ B\log_2\left(1 + \frac{p|h_n|^2/(l_n\sigma^2)}{\sum_{s\in\mathcal{S}_n} p|h_s|^2/(l_s\sigma^2)+1}\right) < a^{th}\right\}$$
$$= Pr\left\{\omega_n\left(2/\sigma_e^2|h_n|^2\right) + \sum_{s\in\mathcal{S}_n} -\phi\omega_s\left(2/\sigma_e^2|h_s|^2\right) < \phi\right\}. \quad (41)$$

Under imperfect CSI, as in (2), the actual Rayleigh fading is represented as the sum of the estimated value and the estimation error. Referring to the definition of the complex Gaussian distribution, h_n, $\hat{h}_n$, and ϵ_n can each be decomposed into their real and imaginary components as

$$\hat{h}_n = \hat{h}_n^1 + i\hat{h}_n^2,$$
$$\epsilon_n = \epsilon_n^1 + i\epsilon_n^2,$$
$$h_n = h_n^1 + ih_n^2 = \left(\hat{h}_n^1 + \epsilon_n^1\right) + i\left(\hat{h}_n^2 + \epsilon_n^2\right),$$

where $\epsilon_n^1, \epsilon_n^2$ follows $\mathcal{N}(0, \sigma_e^2/2)$, $h_n^1 \sim \mathcal{N}(\hat{h}_n^1, \sigma_e^2/2)$, and $h_n^2 \sim \mathcal{N}(\hat{h}_n^2, \sigma_e^2/2)$. Therefore, we have

$$2/\sigma_e^2|h_n|^2 = \left((\sqrt{2}/\sigma_e|h_n^1|)^2 + (\sqrt{2}/\sigma_e|h_n^2|)\right)^2$$
$$\sim \chi_2^2\left(2/\sigma_e^2|\hat{h}_n|^2\right).$$

Similarly, we have $2/\sigma_e^2|h_s|^2 \sim \chi_2^2(2/\sigma_e^2|\hat{h}_s|^2)$. Obviously, the left-hand side of the inequality in $P_{n,m}^{t,otg}$ is a weighted sum of multiple random variables ($2/\sigma_e^2|h_n|^2$ and $2/\sigma_e^2|h_s|^2$) following the non-central chi-square distribution. Thus, with Lemma 1, we can simplify (41) into the following expression:

$$P_{n,m}^{t,otg} \approx Pr\{\alpha_n\Gamma(d_n/2, 2) + \beta_n < \phi\}, \quad (42)$$

where parameters α_n, β_n, d_n are shown as

$$\alpha_n = \frac{\omega_n^3\left(2+3u_n^2\right) + \sum_{n'\in\mathcal{N}_n'}(-\phi\omega_{n'})^3\left(2+3u_{n'}^2\right)}{\omega_n^2\left(2+2u_n^2\right) + \sum_{n'\in\mathcal{N}_n'}(-\phi\omega_{n'})^2\left(2+2u_{n'}^2\right)},$$
$$\beta_n = \omega_n\left(2+u_n^2\right) + \sum_{n'\in\mathcal{N}_n'}(-\phi\omega_{n'})\left(2+u_{n'}^2\right)$$
$$- \frac{\left[\omega_n^2\left(2+2u_n^2\right) + \sum_{n'\in\mathcal{N}_n'}(-\phi\omega_{n'})^2\left(2+2u_{n'}^2\right)\right]^2}{\omega_n^3\left(2+3u_n^2\right) + \sum_{n'\in\mathcal{N}_n'}(-\phi\omega_{n'})^3\left(2+3u_{n'}^2\right)},$$
$$d_n = \frac{\left[\omega_n^2\left(2+2u_n^2\right) + \sum_{n'\in\mathcal{N}_n'}(-\phi\omega_{n'})^2\left(2+2u_{n'}^2\right)\right]^3}{\left[\omega_n^3\left(2+3u_n^2\right) + \sum_{n'\in\mathcal{N}_n'}(-\phi\omega_{n'})^3\left(2+3u_{n'}^2\right)\right]^2}. \quad (43)$$

Since $\alpha_n \in \mathbb{R}$, after moving α_n and β_n to the right-hand side, the complex probability could be simplified into the CDF of the gamma distribution $\Gamma(d_n/2, 2)$. Formally, the approximation of the transmission outage probability is represented as

$$P_{n,m}^{t,otg} \approx \begin{cases} F_{\Gamma\left(\frac{d_n}{2},2\right)}\left(\frac{\phi-\beta_n}{\alpha_n}\right), & \alpha_n > 0, \\ 1 - F_{\Gamma\left(\frac{d_n}{2},2\right)}\left(\frac{\phi-\beta_n}{\alpha_n}\right), & \alpha_n < 0, \end{cases} \quad (44)$$

where $F_\Gamma(\cdot)$ denotes the CDF of the gamma distribution.

References

[1] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2322–2358, Oct. 2017.

[2] J. Du, H. Wu, M. Xu, and R. Buyya, "Computation energy efficiency Maximization for NOMA-based and wireless-powered mobile edge computing with backscatter communication," *IEEE Trans. Mobile Comput.*, vol. 23, no. 6, pp. 6954–6970, Jun. 2024.

[3] L. Zhang, A. Celik, S. Dang, and B. Shihada, "Energy-efficient trajectory optimization for UAV-assisted IoT networks," *IEEE Trans. Mobile Comput.*, vol. 21, no. 12, pp. 4323–4337, Dec. 2022.

[4] T. Park, G. Lee, W. Saad, and M. Bennis, "Sum rate and reliability analysis for power-domain nonorthogonal multiple access (PD-NOMA)," *IEEE Internet Things J.*, vol. 8, no. 12, pp. 10160–10169, Jun. 2021.

[5] Y. Liu, M. Derakhshani, and S. Lambotharan, "Outage analysis and power allocation in uplink non-orthogonal multiple access systems," *IEEE Commun. Lett.*, vol. 22, no. 2, pp. 336–339, Feb. 2018.

[6] S. Guo and X. Zhou, "Robust resource allocation with imperfect channel estimation in NOMA-based heterogeneous vehicular networks," *IEEE Trans. Commun.*, vol. 67, no. 3, pp. 2321–2332, Mar. 2019.

[7] J. A. Ansere et al., "Optimal computation resource allocation in energy-efficient edge IoT systems with deep reinforcement learning," *IEEE Trans. Green Commun. Netw.*, vol. 7, no. 4, pp. 2130–2142, Dec. 2023.

[8] B. Kar, W. Yahya, Y.-D. Lin, and A. Ali, "Offloading using traditional optimization and machine learning in federated cloud–edge–fog systems: a survey," *IEEE Commun. Surveys Tuts.*, vol. 25, no. 2, pp. 1199–1226, 2023.

[9] Z. Zabihi, A. M. Eftekhari Moghadam, and M. H. Rezvani, "Reinforcement learning methods for computation offloading: a systematic review," *ACM Comput. Surveys*, vol. 56, no. 1, pp. 1–41, Jan. 2024.

[10] T. Li et al., "Applications of multi-agent reinforcement learning in future internet: a comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 24, no. 2, pp. 1240–1279, 2022.

[11] P.-Q. Huang, Y. Wang, K. Wang, and Z.-Z. Liu, "A Bilevel optimization approach for joint offloading decision and resource allocation in cooperative mobile edge computing," *IEEE Trans. Cybern.*, vol. 50, no. 10, pp. 4228–4241, Oct. 2020.

[12] J. Du, M. Xu, S. S. Gill, and H. Wu, "Computation energy efficiency Maximization for intelligent reflective surface-aided wireless powered mobile edge computing," *IEEE Trans. Sustain. Comput.*, vol. 9, no. 3, pp. 371–385, May 2024.

[13] J. Bi, H. Yuan, S. Duanmu, M. Zhou, and A. Abusorrah, "Energy-Optimized partial computation offloading in mobile-edge computing with genetic simulated-annealing-based particle swarm optimization," *IEEE Internet Things J.*, vol. 8, no. 5, pp. 3774–3785, Mar. 2021.

[14] W. Zhan, C. Luo, G. Min, C. Wang, Q. Zhu, and H. Duan, "Mobility-aware multi-user offloading optimization for mobile edge computing," *IEEE Trans. Veh. Technol.*, vol. 69, no. 3, pp. 3341–3356, Mar. 2020.

[15] G. Fragkos, N. Kemp, E. E. Tsiropoulou, and S. Papavassiliou, "Artificial intelligence empowered UAVs data offloading in mobile edge computing," in *ICC 2020 - 2020 IEEE Int. Conf. Commun. (ICC)*, Jun. 2020, pp. 1–7.

[16] Y.-H. Xu, Q.-M. Sun, W. Zhou, and G. Yu, "Resource allocation for UAV-aided energy harvesting-powered D2D communications: a reinforcement learning-based scheme," *AD HOC NETWORKS*, vol. 136, Nov. 2022.

[17] J. Gao, Z. Kuang, J. Gao, and L. Zhao, "Joint offloading scheduling and resource allocation in vehicular edge computing: a two layer solution," *IEEE Trans. Veh. Technol.*, vol. 72, no. 3, 2023.

[18] M. Xue, H. Wu, G. Peng, and K. Wolter, "DDPQN: an efficient DNN offloading strategy in local-edge-cloud collaborative environments," *IEEE Trans. Services Comput.*, vol. 15, no. 2, pp. 640–655, Mar. 2022.

[19] S. Yin and F. R. Yu, "Resource allocation and trajectory design in UAV-aided cellular networks based on Multiagent reinforcement learning," *IEEE Internet Things J.*, vol. 9, no. 4, pp. 2933–2943, Feb. 2022.

[20] Z. Gao, L. Yang, and Y. Dai, "Large-scale computation offloading using a multi-agent reinforcement learning in heterogeneous multi-access edge computing," *IEEE Trans. Mobile Comput.*, vol. 22, no. 6, pp. 3425–3443, Jun. 2023.

[21] Y. Lyu, Z. Liu, R. Fan, C. Zhan, H. Hu, and J. An, "Optimal computation offloading in collaborative LEO-IoT enabled MEC: a Multiagent deep reinforcement learning approach," *IEEE Trans. Green Commun. Netw.*, vol. 7, no. 2, pp. 996–1011, Jun. 2023.

[22] X. Xie, H. Wang, and M. Weng, "A reinforcement learning approach for Optimizing the age-of-computing-enabled IoT," *IEEE Internet Things J.*, vol. 9, no. 4, pp. 2778–2786, Feb. 2022.

Authorized licensed use limited to: SOUTH CHINA UNIVERSITY OF TECHNOLOGY. Downloaded on December 22,2024 at 02:44:42 UTC from IEEE Xplore. Restrictions apply.

[23] T. Long, Y. Ma, Y. Xia, X. Xiao, Q. Peng, and J. Zhao, "A mobility-aware and fault-tolerant service offloading method in mobile edge computing," in *2022 IEEE Int. Conf. Web Services (ICWS)*. Barcelona, Spain: IEEE, Jul. 2022, pp. 67–72.
[24] J. Wang, D. Feng, S. Zhang, A. Liu, and X.-G. Xia, "Joint computation offloading and resource allocation for MEC-enabled IoT systems with imperfect CSI," *IEEE Internet Things J.*, vol. 8, no. 5, pp. 3462–3475, Mar. 2021.
[25] W. He, Y. Xu, D. He, and Y. Guan, "Energy minimization in RIS-assisted MEC systems with imperfect CSI," in *Proc. IEEE 98th Veh. Technol. Conf. (VTC2023-Fall)*. Hong Kong, Hong Kong: IEEE, Oct. 2023, pp. 1–5.
[26] F. Fang, K. Wang, Z. Ding, and V. C. M. Leung, "Energy-efficient resource allocation for NOMA-MEC networks with imperfect CSI," *IEEE Trans. Commun.*, vol. 69, no. 5, pp. 3436–3449, May 2021.
[27] L. Wang and G. Zhang, "Deep reinforcement learning based joint partial computation offloading and resource allocation in mobility-aware MEC system," *China Commun.*, vol. 19, no. 8, pp. 85–99, Aug. 2022.
[28] X. Zhou, L. Huang, T. Ye, and W. Sun, "Computation bits Maximization in UAV-assisted MEC networks with fairness constraint," *IEEE Internet Things J.*, vol. 9, no. 21, pp. 20997–21009, Nov. 2022.
[29] Z. Wang, B. Lin, Q. Ye, Y. Fang, and X. Han, "Joint computation offloading and resource allocation for maritime MEC with energy harvesting," *IEEE Internet Things J.*, vol. 11, no. 11, pp. 19898–19913, Jun. 2024.
[30] Y. Hu, X. Deng, C. Zhu, X. Chen, and L. Chi, "Resource allocation for heterogeneous computing tasks in Wirelessly powered MEC-enabled IIOT systems," *ACM Trans. Manage. Inf. Syst.*, vol. 14, no. 1, pp. 1–17, Mar. 2023.
[31] A. Gao, S. Zhang, Y. Hu, W. Liang, and S. X. Ng, "Game-combined multi-agent DRL for tasks offloading in wireless powered MEC networks," *IEEE Trans. Veh. Technol.*, vol. 72, no. 7, pp. 9131–9144, Jul. 2023.
[32] X. Jiao et al., "Deep reinforcement learning empowers wireless powered mobile edge computing: towards energy-aware online offloading," *IEEE Trans. Commun.*, vol. 71, no. 9, pp. 5214–5227, Sep. 2023.
[33] X. Wang, F.-C. Zheng, P. Zhu, and X. You, "Energy-efficient resource allocation in coordinated Downlink Multicell OFDMA systems," *IEEE Trans. Veh. Technol.*, vol. 65, no. 3, pp. 1395–1408, Mar. 2016.
[34] D. W. K. Ng, E. S. Lo, and R. Schober, "Energy-efficient resource allocation in OFDMA systems with large numbers of base station antennas," *IEEE Trans. Wireless Commun.*, vol. 11, no. 9, pp. 3292–3304, Sep. 2012.
[35] J. Zhang, J. Du, Y. Shen, and J. Wang, "Dynamic computation offloading with energy harvesting devices: a hybrid-decision-based deep reinforcement learning approach," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 9303–9317, Oct. 2020.
[36] C. Yu et al., "The surprising effectiveness of PPO in cooperative multi-agent games," *Adv. Neural Inf. Process. Syst.*, vol. 35, pp. 24611–24624, Dec. 2022.
[37] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," Oct. 2018.
[38] Van Dat Tuong, W. Noh, and S. Cho, "Delay minimization for NOMA-enabled mobile edge computing in industrial internet of things," *IEEE Trans. Ind. Informat.*, vol. 18, no. 10, pp. 7321–7331, Oct. 2022.
[39] Z. Wang, H. Rong, H. Jiang, Z. Xiao, and F. Zeng, "A load-balanced and energy-efficient navigation scheme for UAV-mounted mobile edge computing," *IEEE Trans. Netw. Sci. Eng.*, vol. 9, no. 5, pp. 3659–3674, Sep. 2022.
[40] J.-T. Zhang, "Approximate and asymptotic distributions of chi-squared–type mixtures with applications," *J. of Amer. Statistical Assoc.*, vol. 100, no. 469, pp. 273–285, Mar. 2005.

Peng Peng received the B.S. degree from the School of Computer Science and Engineering, South China University of Technology, Guangzhou, China, in 2022, where he is currently pursuing the Ph.D. degree with the School of Future Technology. He is also with Pengcheng Laboratory, Shenzhen, China. His current research interests include edge computing, Internet of Things, and deep reinforcement learning.

Wentai Wu (Member, IEEE) received the bachelor's and master's degrees from the South China University of Technology in 2015 and 2018, respectively, and the Ph.D. degree (Sponsored by CSC) in computer science from the University of Warwick, U.K., in 2022. He is currently an Associate Professor with the Department of Computer Science, College of Information Science and Technology, Jinan University. He has published over 20 refereed papers in the domain. His research interests mainly include distributed systems, federated learning, and sustainable edge intelligence.

Weiwei Lin (Senior Member, IEEE) received the B.S. and M.S. degrees from Nanchang University in 2001 and 2004, respectively, and the Ph.D. degree in computer application from the South China University of Technology in 2007. He was a Visiting Scholar with Clemson University from 2016 to 2017. He is currently a Professor with the School of Computer Science and Engineering, South China University of Technology. He has published more than 150 papers in refereed journals and conference proceedings. His research interests include distributed systems, cloud computing, and AI application technologies. He has been a reviewer for many international journals, including IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, IEEE TRANSACTIONS ON SERVICES COMPUTING, IEEE TRANSACTIONS ON CLOUD COMPUTING, IEEE TRANSACTIONS ON COMPUTERS, and IEEE TRANSACTIONS ON CYBERNETICS. He is a Distinguished Member of CCF.

Fan Zhang received the M.S.E. and Ph.D. degrees in computer science from Northwestern Polytechnical University, China, in 2003 and 2009, respectively. He is currently an Associate Research Fellow with Pengcheng Laboratory. His research interests include edge computing, real-time operating system, and embedded software.

Yongheng Liu received the M.Eng. degree in communication engineering from Xidian University. He is with the Department of New Network Technologies, Pengcheng Laboratory and also currently pursuing the Doctoral degree with the South China University of Technology. His main research interests include cloud OS and industrial IoT resource scheduling optimization.

Keqin Li (Fellow, IEEE) is a SUNY Distinguished Professor of Computer Science with the State University of New York. He is also a National Distinguished Professor with Hunan University, China. His current research interests include fog computing and mobile edge computing, energy-efficient computing and communication, embedded systems and cyber-physical systems, heterogeneous computing systems, big data computing, high performance computing, computer architectures and systems, computer networking, ML, intelligent, and soft computing. He has served on the editorial boards of the IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, the IEEE TRANSACTIONS ON COMPUTERS, the IEEE TRANSACTIONS ON CLOUD COMPUTING, the IEEE TRANSACTIONS ON SERVICES COMPUTING, and the IEEE TRANSACTIONS ON SUSTAINABLE COMPUTING. He is an AAIA Fellow. He is also a member of Academia Europaea (Academician of the Academy of Europe).

Authorized licensed use limited to: SOUTH CHINA UNIVERSITY OF TECHNOLOGY. Downloaded on December 22,2024 at 02:44:42 UTC from IEEE Xplore. Restrictions apply.