Journal of Computers Vol. 35 No. 6, December 2024, pp. 55-71
doi: 10.53106/199115992024123506005

In-Depth Analysis of MEC Resource Optimization and Reliability Under 5G Empowerment

Rongli Chen[1], Xiaozhong Chen[2], Lei Wang[3*], and Guanquan Wu[4]

[1] School of Artificial Intelligence, Dongguan Polytechnic,
Dongguan, 523808, Guangdong, China
23458920@qq.com

[2] School of Business and Trade, Dongguan Polytechnic,
Dongguan, 523808, Guangdong, China
9746042@qq.com

[3] Guangzhou City University of Technology, Engineering Research Institute,
Guangzhou, 510800, Guangdong, China
2019012@dgpt.edu.cn

[4] Department of Academic Affairs Office, Dongguan City University,
Dongguan, 523419, Guangdong, China
623948587@qq.com

Received 24 July 2024; Revised 6 August 2024; Accepted 6 August 2024

Abstract. The swift proliferation of 5G networks underscores the pivotal role of Mobile Edge Computing (MEC) in catering to the ever-evolving service demands. This study introduces a dynamic resource orchestration framework tailored for MEC within 5G standalone cellular architectures. This framework addresses the intricate challenges of computational resource provisioning and power management in a holistic manner. Leveraging convex optimization principles, we devise an optimal resource allocation strategy that incorporates reliability metrics to ensure robust performance. Furthermore, we employ Deep Reinforcement Learning (DRL) techniques to tackle a formulated Markov Decision Process (MDP), aimed at optimizing resource distribution for latency minimization. Our simulation outcomes demonstrate the efficacy of the proposed approach in mitigating task delays and enhancing the system's adaptability across diverse workload profiles and network environments, thereby contributing novel insights into the field with minimal overlap in existing literature.

Keywords: mobile edge computing, resource orchestration, reliability optimization, 5G standalone network architecture, deep reinforcement learning, latency minimization

1 Introduction

With the rapid development of mobile communication, the Internet of Things, 5G, artificial intelligence, big data and other fields, the rich variety of mobile terminal services and applications have become part of people's daily life. Complex services and emerging applications such as virtual reality (VR, Virtual Reality), augmented reality (AR, Augmented Reality), ultra-high definition video and online games can bring immersive user experience. Although these emerging mobile services and applications have greatly enriched people's lives, they also occupy a lot of computing, storage, network and battery resources for smart mobile devices [1]. The emergence of emerging businesses such as cloud games, video broadcast and other emerging services brings better user experience to mobile Internet users, and also brings huge data traffic. Cisco predicts that global network data traffic rises to 163 ZB. In 2025 Ericsson predicts that the total global mobile data traffic in 2026 will grow to 4.5 times that in 2020 to 226EB. per month. Restricted by the computing resources and energy reserves of the basic mobile network, the processing needs of massive data are difficult to be met. Mobile edge computing has emerged as an

* Corresponding Author

efficient solution, and it unloads the computing task to the edge server, using the powerful computing ability of the edge server to expand the resources of intelligent mobile devices, and alleviate the problems of intelligent mobile devices caused by insufficient resources [2]. 5G mobile network can realize the rapid processing of massive data, and the resource allocation and deployment of 5G network can effectively relieve the data processing pressure caused by the surging data traffic. As a new computing mode proposed after cloud computing, mobile edge computing sinks the cloud center computing power to the network edge, and the intelligent mobile device realizes the interaction with the edge server at close range, to meet the needs of low latency and low power consumption for mobile end services and applications [3].

With the development of 5G mobile network technology, more intelligent emerging applications and businesses continue to emerge [4]. China United Network Communications Group Co., Ltd. has carried out an edge-cloud (Edge-Cloud) scale pilot, building more than 30 intelligent model projects such as freight ports, industrial manufacturing, stadiums, and medical treatment in 15 provinces and cities, to promote the rapid implementation of mobile edge applications [5]. In order to break through the resource limitations of the mobile devices themselves, Mobile Cloud Computing (MCC, Mobile Cloud Computing) offloads tasks to the cloud and uses remote computing resources to process tasks. Remote processing of the task imposes a large additional load on the backhaul of the mobile network, and the data exchange in the wide area network also leads to a high latency [6]. Mobile Edge Computing (MEC), proposed by the European Telecommunications Standards Association, can sink remote clouds into the edge cloud of reliable users, avoid network blocking and extend the endurance of mobile devices [7].

This paper constructs a long-term delay minimizing Markov model for the system state and external environment dynamic uncertainty in the 5G single cellular network [8]. For the problem of high-dimensional continuous and difficult accurate characterization of random variables in state space and action space, the established Markov model is solved with the reinforcement learning method to obtain the approximate optimal dynamic resource allocation strategy, minimize the long-term delay of the task, and conduct simulation verification [9].

2 Related Work

2.1 The Application of Mobile Edge Computing

Mobile Edge Computing (MEC) is an emergent architecture where cloud computing services are extended to the edge of networks leveraging mobile base stations [10]. Mobile edge computing (MEC, Mobile Edge Computing) enables computing resources and storage resources to be pushed from the cloud computing center to the network edge, so the network edge device has the ability to perform computing and analyze data from the cloud computing center, which allows data services as well as other related processing tasks to operate near the mobile user [11]. Therefore, MEC not only reduces service latency, but also minimizes network traffic, both advantages are very important for time-limited services, enabling users to obtain a better service experience, it is a typical application in mobile computing [12].

Mobile edge computing has the advantages of independent deployment, alleviating data center and network pressure, close low latency and high user privacy security [13, 14]. MEC application scenarios can be divided into three categories: local diversion, data service and business optimization, and is currently widely used in the Internet of Vehicles, Enhanced Reality (AR) / virtual reality (VR), smart city, smart wear and other fields [15].

Combining mobile edge computing can both improve the network capacity and enhance the computing power of mobile devices [16]. In a 5G ultra-dense network combined with mobile edge computing, the density of access points and the edge server makes network resource management more complex, requiring to design an efficient and reasonable resource allocation strategy to optimize the performance of the system and reduce the energy loss of the system [17, 18].

2.2 Minimize the Execution Delay

The Markov decision to achieve comprehensive optimization of computational task status, use of local computing resources, wireless transmission channel utilization, and MEC server utilization for computational uninstallation [19]. The algorithmic goal of minimizing execution time can be achieved by analyzing data processing time and energy costs. Finding the best solution to compute the uninstall via a one-dimensional search while satisfying the constraints. The algorithm greatly reduces the execution time compared to no computational uninstall, but it is very complex [20].

The problem of waiting time minimizing in multi-user time division multi-access systems, and an optimal joint communication and computational resource allocation algorithm to significantly reduce the end-to-end waiting time [21].

The optimization of task migration cost in edge computational systems, considering task heterogeneity and node variability, involves solving an integer nonlinear programming problem. This is achieved by analyzing the problem's structure and converting it into a linear programming problem. Additionally, addressing task delay minimization in Multi-Layer UAV Networks entails proposing a bipartite search iterative algorithm. This algorithm targets minimizing task delay while reducing complexity and evaluating optimality. It achieves this by deriving optimal task allocation ratios and optimum unloading power [22, 23].

2.3 Resource Allocation

The dynamic resource allocation strategy for MEC-centric joint power control and computation resource allocation in a 5G single-cell network is analyzed, taking into account the dynamic and stochastic nature of user states and the requirements for service quality [24]. To simplify the complexity of the problem, the original dynamic resource allocation model is divided into two sub-problems: computation resource allocation and power control. Considering the dynamic characteristics of the power control problem, it is transformed into a Markov decision process (MDP) problem, and a dynamic resource allocation algorithm based on deep reinforcement learning (DRL) is proposed. Additionally, a heuristic task partitioning algorithm can be designed to minimize task execution latency, and a polynomial-time approximation algorithm can be proposed to minimize execution latency while satisfying specified resource utilization constraints [25].

To reduce energy consumption in mobile devices, a dynamic computational uninstallation algorithm utilizing Lyapunov optimization technology has been developed to determine which applications should be offloaded to the Mobile Edge Computing (MEC) server [26]. Additionally, joint management of radio and computational resources in a multi-user MEC system has been studied, addressing the average weighted sum power minimization problem while ensuring task buffer stability through a low-complexity online algorithm based on Lyapunov optimization. Furthermore, a convex optimization problem targeting total mobile device energy consumption reduction has been formulated, allowing mobile users to share edge server resources over time, with a focus on jointly optimizing offloading decisions and wireless bandwidth resource allocation using separable semidefinite relaxation methods [27]. Finally, a stochastic task-arrival model has been employed to solve the energy and delay-weighted average sum minimum in a multi-user MEC system, utilizing an optimized Lyapunov-based online algorithm [28].

In the context of software-defined network virtualization cellular networks, a novel MEC framework incorporating mobile user virtualization schemes is introduced to integrate wireless, computational, and storage resources for collaborative MEC computing tasks. This framework addresses virtual resource allocation through a joint optimization problem formulation [29]. To enhance efficiency, a distributed resource allocation algorithm utilizing alternating direction of multipliers is proposed, effectively reducing computational complexity and signaling overhead. Furthermore, extending the study to a multi-MEC server deployment model in ultra-dense networks, a delay-constraint-based energy minimization problem is developed. The results indicate that deploying multiple MEC servers in Ultra-Dense Networks (UDNs) can significantly enhance network performance, as demonstrated by a heuristic greedy offinstall solution [30].

In the network scenario with a single terminal device and multiple MEC candidate nodes, three algorithms based on heuristic search, reformed linearization technique, and semi-definite relaxation are proposed to minimize task latency and offloading failure probability [31]. Leveraging the concept of software-defined networking, the task offloading problem in ultra-dense networks is studied, transforming it into task placement and resource allocation sub-problems, and presenting an effective software-defined task offloading solution. Additionally, a unified framework is introduced to maximize the profit of Mobile Service Providers (MSPs) in MEC by jointly scheduling wireless bandwidth and computational resources [32]. This extends standard Lyapunov techniques and designs a variable-length algorithm for making online decisions on variable-length job requests within continuous time [33].

2.4 Task Scheduling

A task scheduling algorithm is essential for efficient execution in a wireless access network's edge, where MEC servers are extensively deployed. The distribution of mobile users across these areas can lead to computing load imbalances among servers. Therefore, a task scheduling scheme based on resource attribute selection is crucial. This approach selects the most suitable nodes for task execution by considering the resource requirements and fitness between the resource node and the task, thereby directly impacting system efficiency [34].

An efficient task flow scheduling algorithm is proposed, aiming to dynamically estimate the execution time of tasks, handle multiple applications simultaneously, and optimize resource configuration for virtual machines (VMs) to ensure tasks are completed within their cutoff [35]. To address resource load imbalance in cloud platforms, a scheduling strategy based on cloud computing and the Min-Min algorithm is suggested, reducing task completion time and enhancing resource utilization. Additionally, an improved heuristic heterogeneous algorithm is introduced to minimize communication costs during task forwarding, further shortening task completion time. Current task scheduling research has mainly focused on parallel independent tasks or single task flow scheduling, neglecting the potential of sharing distributed computing resources among multiple task streams [36]. By promoting cooperative resource sharing among task streams, idle computational slots can be efficiently utilized, leading to significant improvements in distributed system resource utilization, thus warranting further investigation.

2.5 Reduce the Maximum Energy Consumption

To enhance resource utilization efficiency in IoT edge computing systems, a system model integrates energy consumption of IoT mobile devices and processing delay of computational tasks. This model employs an efficient two-layer game greedy approximate off-loading scheme, aimed at minimizing total computational overhead. Addressing resource allocation as a Markov Decision Process (MDP), a Deep Reinforcement Learning (DRL) based algorithm is proposed. This algorithm collaboratively allocates resources to minimize the long-term weighted sum of average task completion time and average number of requested resources, achieving optimal resource allocation for the system [37, 38].

Optimizing MEC resource allocation in super-dense networks requires a sophisticated planning model. The approach integrates convex optimization principles with greedy strategies and golden segmentation methods to tackle power control challenges associated with channel joint allocation and user uploads. The objective is to enhance the efficiency of task upload data, minimizing both delay and energy consumption [39].

Previous research in computational uninstallation has often been limited, typically addressing only specific aspects of uninstallation decisions. This overlooks critical factors such as wireless and computing resource allocation, high interference, multiple access, and limited computational resources. Our study aims to bridge this gap by conducting a comprehensive investigation into computational uninstallation schemes.

3 Methodology

3.1 Dynamic Resource Allocation Strategy for Mobile Edge Computing in 5G Single Cellular Network Study

Around the joint dynamic allocation problem of MEC-oriented power and computational resources in the 5G single cellular network, a dynamic resource allocation model with the optimization objective of minimizing the long-term average delay of the total task is established. To describe the dynamic nature and sustainability generated by the task, the original dynamic resource allocation model is transformed into a MDP model.

5G Dynamic Resource Allocation System Model in Single Cellular Network. Various complex services and emerging businesses such as the industrial Internet of Things, driverless driving and intelligent fire fighting are constantly emerging. In general, UE has very limited computing power and resources such as electricity. To address limited user device resources, it can uninstall tasks to the MEC server for efficient execution. For the 5G single cellular network resource allocation problem, consider the scenario of multi-device dynamic unloading task, as shown in Fig. 1.

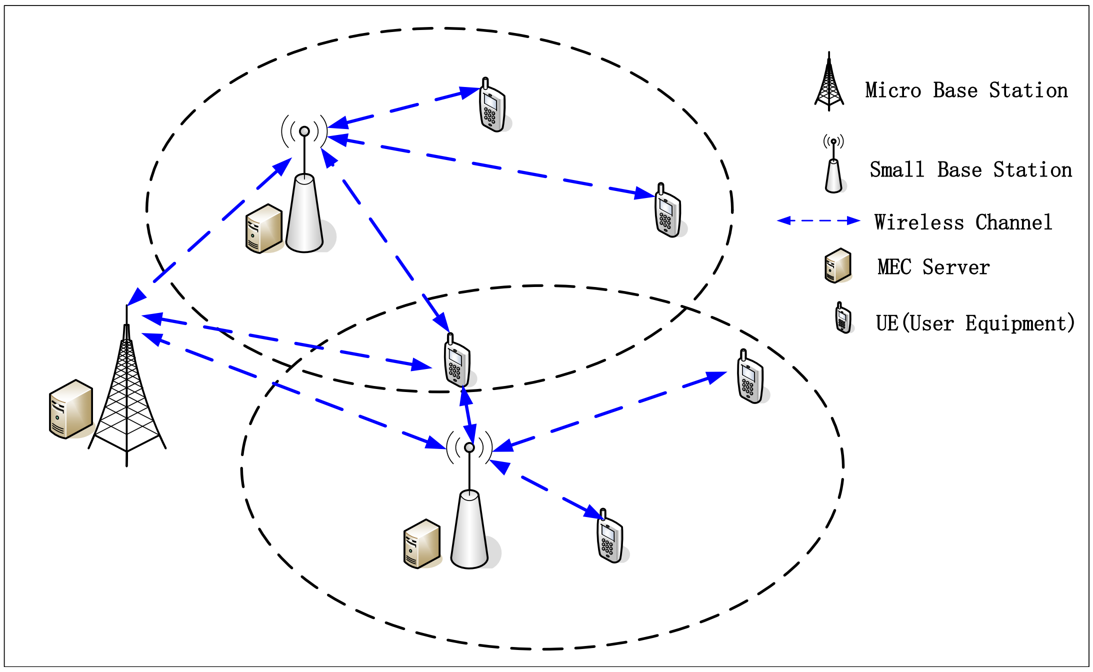

Fig. 1. MEC system

As show in Fig. 1, in the MEC system, the heterogeneous network consists of an macro base station (Macro Base Station, MBS) and several small base stations (Small Base Station, SBS) with overlapping cover regions between each base station, and user devices (User Equipment, UE) randomly scattered throughout the region. The network operator deploys the MEC server on each base station, with certain computing and storage capabilities and establishes wireless channel connections to the mobile devices through the base station., a base station and a MEC server are deployed in a 5G single cellular network center, the N UE s are distributed in 5G cells, and the collection of UE can be represented as N = [1, 2, ..., ..., N]. Using the discrete time slot model t∈{1, 2, ...T}, where the duration of each time slot is τ , i. e.. The calculation task generated by the n UE of each slot t, can be represented as the tuple $Q_n(t) = \{\omega_n(t), Z_n(t)\}$, where $\omega_n(t)$ represents the computation of the task, the number of CPU cycles required to complete the task; $Z_n(t)$ represents the data size of the task [40]. The maximum completion time for each task does not exceed each slot length τ.

Task Model. This design maximizes the system benefit as the optimization goal, and based on the above discussion, the UEn benefit function can be expressed as

$$U_n(a^s_{nm}, b^s_{nm}, f_n) = \frac{C_n - (Z^l_n + Z^r_{nm})}{C_n} \tag{1}$$

In formula (1), the calculated unloading decision configuration vector a^s_{nm} is given, when the local execution cost Z^l_n is higher than the cloud execution cost Z^r_{nm}, $a^s_{nm} = 1$, perform a calculation of the uninstallation. When the local execution costs Z^l_n are lower than the cloud execution costs Z^s_{nm}, $a^s_{nm} = 0$, the calculation task resides and performs locally. The problems of joint computing offloading decisions, wireless bandwidth, and computational resource allocation can be expressed as

$$\begin{aligned}
&\max_{\{A,B,F\}} U(A,B,F)=\sum\nolimits_{n\in N}\sum\nolimits_{m\in M}U_n(a_{nm}^s,b_{nm}^s,f_n)\\
&s.t C1: a_{nm}^s=\{0,1\},\forall n\in N,m\in M,s\in S\\
&C2:\sum\nolimits_{n\in N}a_{nm}\le 1,\forall n\in N\\
&C3:\sum\nolimits_{n\in N}a_{nm}^s\le 1,\forall m\in M\\
&C4:\sum\nolimits_{n\in N}\sum\nolimits_{s\in S}a_{nm}^s\le q_m,\forall n\in N\\
&C5:\sum\nolimits_{n\in N}f_{nm}^{mec}\le f_m^{\max},\forall m\in M\\
&C6:\sum\nolimits_{n\in N}b_{nm}^{mec}\le B_m^{\max},\forall m\in M\\
&C7: t_n\le T_n^d,\forall n\in N\\
&C8: Z_n\le C_n,\forall n\in N
\end{aligned} \quad (2)$$

In Equation (2), the constraint C1 defines this calculation unload as a binary unload, and the calculation task can only be performed locally or all uninstalled to the MEC server. Constraints C2 and C3 ensure that it can unload only one subchannel at a time and use only one subchannel at a time, and that each subchannel can only serve a single UE. at a timeConstrained C4 specify an limit on the limit of the number of service UE s per MEC server.C5 indicates that the number of computational resources provided cannot exceed the total computational resource capacity of the MEC server. Similarly, C6 gives that the number of bandwidth resources available per MEC server cannot be higher than the maximum amount of bandwidth available. C7, C8 limits the task execution delay from exceeding the specified time range and processing the computational task from exceeding the offload budget.

3.2 A Dynamic Resource Allocation Algorithm Based on Deep Reinforcement Learning in a 5G Single Cellular Network

There is no interdependence or interaction between the decision variables P (power control) and f (computational resource allocation). To reduce the complexity of the problem solving, we can first extract the computing resource allocation subproblems to obtain the optimal decision of the computational resource allocation. Based on this, the UE transmission power is dynamically controlled to achieve a joint allocation strategy for computing resources and transmission power.

Arithmetic Resource Allocation Subproblem Extraction. For computational resource allocation subproblems, only consider the calculation delay of the task on the MEC server. The task generated within each slot is completed within the current slot, and the computational resources occupied will be released at the next slot beginning. The current time slot will have no impact on the future time slot. When optimizing the computing resource allocation subproblem, it can be transformed into the static optimization problem, which only needs to minimize the calculation delay of the current time slot, formalized as:

$$\begin{aligned}
&\text{P1}:\min_{f}\sum\nolimits_{n=1}^{N}\frac{\omega_n(t)}{f_n(t)}\\
&\text{s.t. C1}: f_n(t)>0,\ \forall n\epsilon\aleph\\
&\text{C2}:\sum_{n=1}^{N}f_n(t)\le F
\end{aligned} \quad (3)$$

Arithmetic Resource Allocation Subproblem Solving. For the problem P1, its target function can be converted to:

$$\varphi(\mathrm{f}) = \sum_{n=1}^{N} \frac{\omega_n(t)}{f_n(t)} \tag{4}$$

$\varphi(f)$ is a function of the variable $\{f_1(t), f_2(t), ..., f_N(t)\}$, whose Hessian matrix is:

$$\mathrm{H} = \nabla^2\varphi = \begin{bmatrix} \frac{\partial^2\varphi}{\partial f_1^2(t)} & \frac{\partial^2\varphi}{\partial f_1(t)\partial f_2(t)} & \cdots & \frac{\partial^2\varphi}{\partial f_1(t)\partial f_N(t)} \\ \frac{\partial^2\varphi}{\partial f_2(t)\partial f_1(t)} & \frac{\partial^2\varphi}{\partial f_2^2(t)} & \cdots & \frac{\partial^2\varphi}{\partial f_2(t)\partial f_N(t)} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2\varphi}{\partial f_N(t)\partial f_1(t)} & \frac{\partial^2\varphi}{\partial f_N(t)\partial f_2(t)} & \cdots & \frac{\partial^2\varphi}{\partial f_N^2(t)} \end{bmatrix} \tag{5}$$

Each element can be represented as:

$$\mathrm{h_{ij}} = \frac{\partial^2\varphi}{\partial f_i(t)\partial f_j(t)} = \begin{cases} \frac{2\omega_i}{f_i^3(t)}, \mathrm{i=j} \\ 0, \quad \mathrm{i \neq j} \end{cases} \tag{6}$$

In formula (6), each ω_i and $f_i(t)$ are an integer, then $h_{ij} \geq 0$. For all the elements on the Hessian matrix H, diagonal are positive the other elements are equal to zero resulting that all the eigenvalues of H are positive and H is a positive definite matrix. $\varphi(f)$ is a convex function, The constraints of problem P1 take the form of convex functions, which makes problem P1 a convex optimization problem.

Problem P1 to augmented Lagrangian is available:

$$\mathrm{L}(\mathrm{f}, \lambda, \mu) = \sum_{n=1}^{N} \frac{\omega_n(t)}{f_n(t)} + \sum_{n=1}^{N} \lambda_n f_n(t) + \mu\left(\sum_{n=1}^{N} f_n(t) - F\right) \tag{7}$$

Available according to the Karush-Kuhn-Tucker (KKT) condition:

$$\begin{cases} \nabla\varphi(f^*) - \sum_{n=1}^{N} \lambda_n \nabla f_n^*(t) - \mu\nabla\left(F - \sum_{n=1}^{N} f_n^*(t)\right) = 0 \\ f_n^*(t) \geq 0, \forall n \epsilon N \\ F - \sum_{n=1}^{N} f_n^*(t) \geq 0 \end{cases} \tag{8}$$

Solution formula (8) is available for:

$$f_n^*(t) = \frac{F\sqrt{\omega_n(t)}}{\sum_{n=1}^{N} \sqrt{\omega_n(t)}}, \forall n \epsilon N \tag{9}$$

Since problem P1 is a convex optimization problem, then f^* is the global optimal computational resource allocation decision for problem P1.

Power Control Issues Conversion. Based on the optimal solution, the original optimization problem P1 can be remodeled as a stochastic optimization problem about the variable P, i. e.:

$$
\begin{aligned}
&\text{P2:} \min_{P} \frac{1}{T}\sum_{t=1}^{T}\frac{1}{N}\left(\sum_{n=1}^{N} d_n(t) + \sum_{n=1}^{N}\frac{\omega_n(t)}{\mathrm{f}_n^*(t)}\right)\\
&\text{s.t. C1:} P_{min} \le P_n(t) \le P_{max}, \forall n \epsilon N \\
&\text{C2:} \sum_{t=1}^{T} \mathrm{e}_n^w(t) \le E_n, \forall n \epsilon N
\end{aligned}
\tag{10}
$$

The time-slot states of the problem P2 and t + 1 grooves are only related to the state and action of the t-groove, but not to the power control decision before the t-groove, suggesting that it has Markovian properties.

Joint Allocation of Bandwidth and Computing Resources. The MEC server establishes the joint allocation principle of wireless bandwidth and computing resources, and defines the method of resource allocation. When the MEC service is available m also received a task uninstall request from k UE s, register as a collection $K = \{1, 2, ..., k, ...K\}$, satisfied constraint condition $K \subseteq N$, $K \le q_m$. B_m and F_m represents the total bandwidth and computing resources of the MEC server. The wireless bandwidth resources and computing resources allocated by the MEC server for the k th UE are expressed as b_{km} and f_{km}, satisfied $\sum_{k\in K} b_{km} \le B_m, \sum_{k\in K} f_{km} \le F_m$. To ensure the fairness of each UE, bandwidth resources B_m and computational resources F_m are allocated by task execution budget ratio. The bandwidth and calculated resource size allocated by UEk to can be represented as

$$
b_{km} \le \frac{c_k B_m}{\sum_{i\in K} c_i}, \forall k, m \tag{11}
$$

$$
f_{km} \le \frac{c_k F_m}{\sum_{i\in K} c_i}, \forall k, m \tag{12}
$$

The Mobile UEk task execution budget c_k is indicated in Equation (8). If the calculation offload delay is greater than the task execution deadline during the resource allocation process, the MEC server needs to reject a user access request with a low task uninstall priority and stop receiving new computing tasks. The rejected UE needs to re-select the target MEC server to send a computational uninstall request.

Total communication and calculation costs incurred from UEk calculation unloading are further obtained according to Equation (1).

$$
Z_{km}^r = \gamma_{km}^t (t_{km}^{off} + t_{km}^{exe}) + \gamma_{km}^e E_{km}^{off} + \gamma_{km}^c c_{km}, \forall k, m \tag{13}
$$

The total cost of UE in the MEC server m is obtained from the bandwidth and computational resources allocated by each UE.

$$
Z_m^r = \sum_{k\in K} Z_{km}^r \tag{14}
$$

The total system off-installation overhead is

$$Z^r = \sum_{m\in M}\sum_{k\in K} Z^r_{km} \tag{15}$$

When all UE s define their off install targets, a set of initial uninstall policies. However, this strategy is not the best matching combination. For example, in the initial state, the MEC server i reserved resource is larger than the MEC server j, both found by nearby UE, both selected the i th MEC server as the uninstall target, resulting in MEC server i has been task-busy, while MEC server j is always idle without receiving task uninstall messages, causing uneven resource distribution.

Dynamic Resource Allocation Algorithm based on Deep Reinforcement Learning Designs. In the problem P2, prior knowledge like state transfer probabilities is unknown, and the introduction of a model-free approach to reinforcement learning is required to obtain statistics about the unknown model. In the MDP problem transformed by the problem P2, the state and action spaces have higher dimensions, and each variable is a continuous variable. The discretization of state space and action space causes insufficient solution accuracy and space explosion. For example, Q-leaming's traditional RL method is inefficient due to computing value functions of all possible system and action spaces, such as DQN's deep reinforcement learning based on value functions can only be applied to the MDP problem in discrete space. Problem P2 is translated to solving the MDP problem by using a deep reinforcement learning framework.

Define $a(t) = \mu(S(t))$, a(t)) represents the desired return value that the agent obtains by selecting a (t) in the current state s (t). According to the Bellman agenda, $Q^{\mu}(s(t), a(t))$ can be represented as:

$$Q^{\mu}\left(s(t), a(t)\right) = IE[r(t) + \gamma Q(s(t+1), \mu(s(t+1)))] \tag{16}$$

$$Q(s(t), a(t)|\theta^Q) \approx Q(s(t), a(t)) \tag{17}$$

Where, γ is a discount factor that θ^Q represents the trainable parameter of the critic network.with θ^{μ} fit optimal policy μ^* i. e.:

$$\mu(s(t)|\theta^{\mu}) \approx \mu^*(s(t)) \tag{18}$$

Where the $\mu^*(s(t))$ represents the optimal behavior policy. To improve the stability of network training, the Critic and Actor networks are replicated respectively, introducing the target networks $Q'(s,a|\theta^Q)$ and $\mu'(s|\theta^{\mu})$. The construction mode is consistent with the original network structure.

To prevent the agent from falling into a local optimum during exploration, a noise micropert urbation mechanism is introduced in decision making. Adopt the strategy of introducing the noise:

$$\beta = \mu(s(t)|\theta^{\mu}) \tag{19}$$

In Actor networks, the performance of policy μ can be represented as

$$J_{\beta}(\mu) = E_{p^{\beta}}\left[Q^{\mu}(s(t), \mu(s(t)|\theta^{\mu}))\right] \tag{20}$$

Among them, p^{β} is a state probability distribution function based on different behavioral policies β. The optimal behavior policy, the name is, by maximizing the performance function

$$\mu^*\left(s(t)\right) = \arg \quad max J_{\beta}(\mu).\mathrm{T} \tag{21}$$

According to the DDPG framework, a DRL-based dynamic resource allocation algorithm (DDRA, DRLbased Dynamic Resource Allocation) is proposed under a 5G single cellular network. After the training of the participant network and the critic network, the parameters of the target network were updated according to the updating guidelines.

$$\begin{cases} \theta^{Q'} = \rho\theta^{Q} + (1-\rho)\theta^{Q'} \\ \theta^{\mu'} = \rho\theta^{\mu} + (1-\rho)\theta^{\mu'} \end{cases} \tag{22}$$

Where $\rho << 1$, the stability of DNN training can be improved
Specific steps of DDRA such as Algorithm 1 As shown.

Algorithm 1. 5G Dynamic Resource Allocation Algorithm (DDRA) in single cellular

```
Algorithm   Dynamic Resource Allocation Algorithm (DDRA)
1: Initialize the network parameters $\theta^{Q}$ and $\theta^{\mu}$;
2: Initialize the target network parameters: $\theta^{Q'} \leftarrow \theta^{Q}$ and $\theta^{\mu'} \leftarrow \theta^{\mu}$;
3: for UE n =1: N do
4:   Initialize the 5G single cellular network states;
5:     for m=1: M do
6:     Calculate a(t) according  Equation (16)
7:     Perform the a (t) to make a power control decision and allocate computing resources for the task according to Equation (9);
8:    Calculate the total communication and calculation costs according to  Equation (13);
9:    Calculate the total UEi overhead $Z^{r}$ according to  Equation (14);
10:    Update the target network parameters according to  Equation (22);
11:    end for
12: end for
```

4 Experiments

For the proposed DDRA method, the learning rate and discount factors analyze the influence of different algorithm parameters on the performance of DDRA algorithm. The DDRA algorithm is verified by comparative experiments under a variety of experimental environments and task indicators.

4.1 Simulation Platform Construction and Experimental Parameter Setting

Through Python3.7 and MATLAB 7 a simulation tools, build 5G single cellular network and DDRA algorithm simulation experimental environment, the experimental hardware is configured as Intel Xeon (R) CPU E5-2620v4@2.10GHz processor, Titan V@1455MHz 12G.

The software was configured as a version for the Ubuntu 16.04 Linux operating system, with video stored on a graphics card and 128 g of memory, and the DDRA is based on Tensorflow version 1.14.0. Deploy a BS, BS with a cover radius of 500 m in the built 5G single cellular network center with a MEC server with a computational capacity of 25GHz. The specific simulation parameters of the experiment are shown in Table 1.

Table 1. Simulation parameter setting

Simulation parameters	Numerical value	Parameter meaning
$\lambda_n^t, \lambda_n^e, \lambda_n^q$	0.4,0.4,0.2	Weight
N	[10,40]	Number of UE s
B	20MHz	Total system bandwidth
$P_n(t)$	[5,38]dBm	transmitted power
$g_n(t)$	127+30lobd	channel gain
$\omega_n(t)$	[0.5,2]Gigacycles	Average task calculation amount
$z_n(t)$	[2,10]MB	Average data amount of the task
E_{max}	1000J	Maximum battery capacity
P_n	100mW	The mobile user sends the power

C	3000	Reback memory buffer experience pool capacity
M	64	Small-batch data size
T	100	Total number of time gaps
w_n	1000MHz	Task calculation intensity
f^l_n	0.5GHz,0.8GHz,1.0Hz	Mobile user computing power
f_m	6.0GHz	MEC Server computing power
n_0	-100dB	White Gaussian noise, WGN
d_n	0~500m	The mobile user has a distance from the base station
l_n	0.5~1MB	The input data size

4. 2 Algorithm Parameter Analysis

Learning rate and discount factors are important parameters affecting the performance of the DDRA algorithm. The most appropriate algorithm parameters are set by comparing and analyzing the effects of different learning rates and discount factors on the convergence performance of the algorithm.

Influence of the Learning Rate on the Convergence. Fig. 2 describes the impact of different learning rates on the task average delay convergence performance in the DDRA algorithm, with the learning rates set to 0.1, 0.01, and 0.001. Comparative analysis can conclude that if the average delay converges, the faster the convergence at the greater the learning rate. When the convergence rate is too large, it is difficult to ensure that the convergence obtained value is optimal and even the average delay is always divergent. The smaller the learning rate, the smaller the step, makes it take more time to optimize. To ensure the average delay convergence rate and convergence quality, the learning rate of DDRA algorithm is set to 0.01.

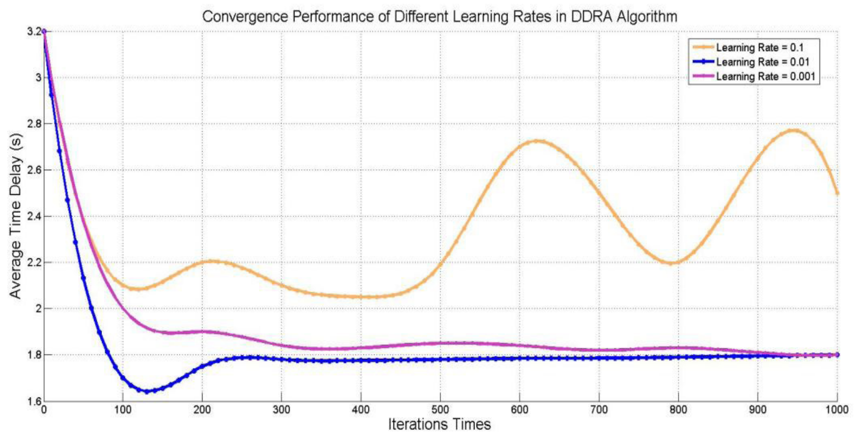

Fig. 2. The influence of the learning rate on the convergence performance of the DDRA algorithm average time delays (S)

Influence of the Discount Factor on the Convergence. In Fig. 3, if the factor of discount is 0.09, DDRA algorithm is in a fluctuating divergence until the number of 800, especially between the number of iterations of 350-500.After the number of iterations of 800, the task average delay curve approaches ates and decreases at a slow speed. After 1000 round iteration, the task average delay still not converges to optimal. If the factor of discount is 0.59, the fluctuates of task average time delay dramatically between iterations are 200-300 and slow decreases when iterations are 400. After 1000 round iteration training, the task average time delay does not converge to optimal. If the factor of discount is 0.99, the task average delay curve converges to the optimal level before the iterations are 200.

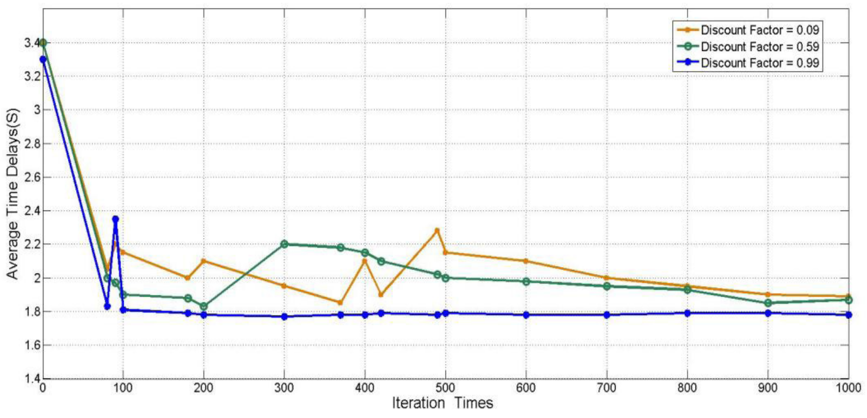

Fig. 3. Effect of the factors of discount on the convergence performance of the DDRA algorithm

The more important the agent in the DDRA algorithm is the current reward, resulting in the delay selected by the agent can only be reduced in the short term, and the long-term average delay cannot be optimized. When the discount factor is too hour, the agent ensures that the short-term delay is easy to produce some interference decisions that make the long-term delay high, which go into the playback memory buffer experience pool to replace some decisions that make the long-term delay low. Training for agents causes sharp fluctuations in the convergence curve due to the influence of interfering decisions. As the training proceeds, the fitting accuracy of the agent to the optimal decision is optimized and flattened. On the contrary, the greater the discount factor, the greater the proportion of the future reward, the minimization of the long-term average delay, less interference decisions, and the easier to call back when fluctuations are generated. To guarantee the convergence quality of the average delay, the discount factor of the DDRA algorithm is set to 0.99.

4.3 Algorithm Comparison Experiment and Reliability Analysis

To further verify the effectiveness of the DDRA algorithm, it evaluates the average delay of the task in terms of UE number, task computation amount, task data size (URA, Uniform Resource Allocation), Random Resource Allocation Algorithm (RRA, Random Resource Allocation) and asynchronous Advantage Operator Critics Algorithm (A3C, Asynchronous Advantage Actor-critic).

Effect of the Number of UE on the Task Average Delay. Fig. 4 depicts the task average time delays calculated by the DDRA, RRA, URA and A3C algorithms for different numbers of UEs, where the number of UEs is set to [10, 15, 20, 25, 30, 35, 40], respectively. Both the computational and data quantities follow a normal distribution, with an average of 1.25 gigabit function and 6MB.

From Fig. 4, the more UE numbers, the higher the average delay of the task. With the number of UE ports, less upside bandwidth per UE and less computational resources per task from the server. As the uplink bandwidth and the obtained computational resources decrease, the transmission and computational delay of each task increase, resulting in increases in the average delay of the task. Overall, the DDRA algorithm outperforms RRA, URA with A3C.The DDRA algorithm helps to reduce the average delay in the RRA. 88% and 26%, respectively, because of optimizing the average time delay of the task.

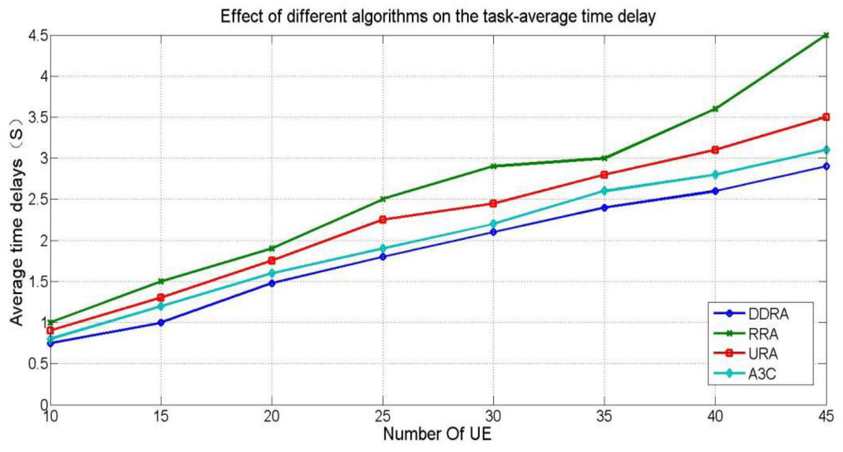

Fig. 4. The average task delay was affected by the number of UE

Impact of the Task Calculation Quantity on the Average Task Delay. Fig. 5 describes the average task delay calculated by the DDRA, RRA, URA and A3C algorithms for different tasks following a Gaussian distribution, with the mean set to [0.5, 0.751, 1.25, 1.5, 1.75, 2, 2.25] Gigacycles. The task data sizes are subject to the Gaussian distribution with the mean of 6MB, UE set to 25.

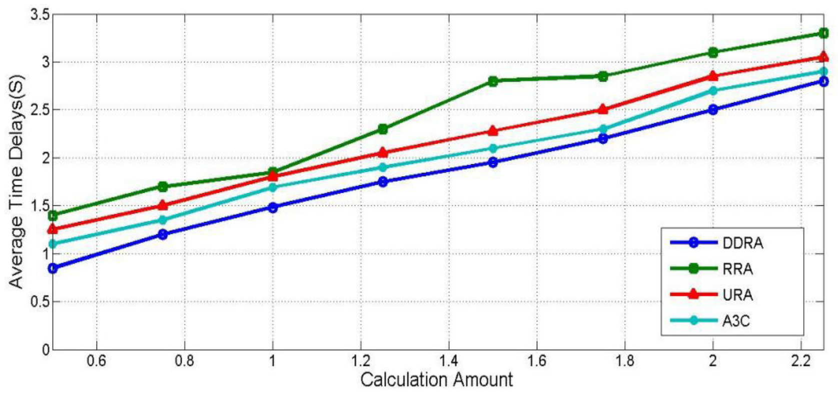

Fig. 5. Effects of the task calculation quantity on the average task delay

As shown from Fig. 5, the greater the computation of the task, the higher the average delay of the task. The reason is that when the total computational resources of the MEC server are fixed, the computation delay of each task becomes higher as the computational amount of each task increases. Therefore, the long-term average time delay for the total task increases. Moreover, DDRA-optimized tasks have lower average delays than RRA, URA and A3C.Overall, the DDRA-optimized task delay averaged 16.3% lower than URA, 26.1% lower than RRA, and 9.8% lower than A3C.

By calculating the average task latency of DDRA, RRA, URA, and A3C algorithms on different tasks and observing the results in Fig. 5, reliability analysis can be performed.

Firstly, the observed phenomenon of an increase in task computation correlating positively with average latency in MEC (Mobile Edge Computing) environments is reasonable. This is because, with constant computing resources, tasks with larger computational loads require more time to complete, resulting in an increase in average latency. Hence, this observation aligns with our fundamental understanding of edge computing environments.

Secondly, from the results in Fig. 5, it can be observed that tasks optimized through the DDRA algorithm exhibit an advantage in average latency compared to RRA, URA, and A3C. This is because the DDRA algorithm can schedule tasks more efficiently, reducing the execution time of tasks on MEC servers, thereby decreasing the average latency. This also reflects the efficiency of the DDRA algorithm in task processing.

Comparison of Dynamic Resource Allocation Algorithms that Reduce Task Latency and Improve System Reliability. Fig. 6 depicts the average time delay for DDRA, RRA, URA and A3C tasks under different task data quantities, the task obeying a Gaussian distribution with the mean set to [3, 4, 5, 6, 7, 8, 9, 10]MB. The distribution of computed quantities for this task follows a Gaussian distribution with a mean of 1.25 gigrings, and the UE is set to 25.

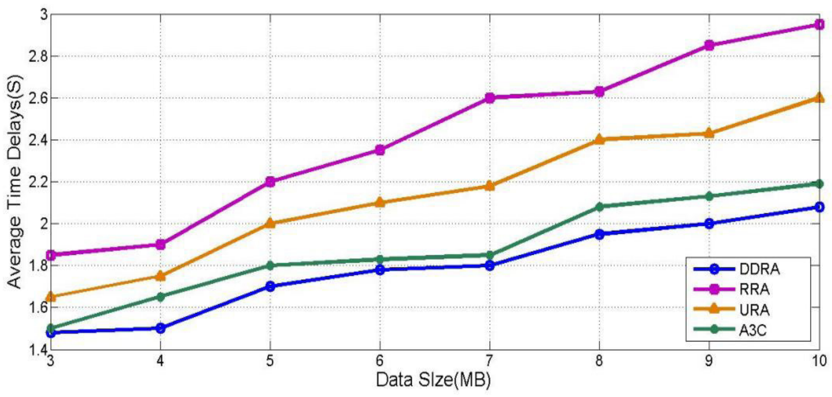

Fig. 6. Effect of task data volume on the average task delay

According to the data presented in Fig. 6, there is a positive correlation between the task data volume and the average latency of tasks. As the data volume per task increases while maintaining the uplink speed of user equipment (UE), the transmission delay for each task also increases, resulting in an overall increase in the average latency of all tasks. In comparison to traditional resource allocation algorithms such as Round Robin Algorithm (RRA), User-centric Round Robin Algorithm (URA), and Asynchronous Advantage Actor-Critic (A3C), the Dynamic Resource Allocation Algorithm (DDRA) demonstrates superior performance in optimizing average task latency.

Specifically, compared to URA, DDRA can reduce average task latency by 16.3%; compared to RRA, it reduces it by 24.3%; and compared to A3C, it achieves a reduction of 5.7%. This indicates that in a 5G single-cell network, employing DDRA for joint transmission power control and computational resource allocation can effectively reduce task transmission latency and enhance system performance [41, 42].

In the study, to simplify the complexity of the problem, the original optimization problem is decomposed into two equivalent subproblems. The first subproblem (P1) is a convex optimization problem, and its exact solution is found under the guidance of KKT conditions. For the second subproblem (P2), it is transformed into a Markov Decision Process (MDP) problem to ensure the minimization of long-term average latency. A DDRA algorithm based on Deep Reinforcement Learning (DRL) dimensions is proposed, combining the optimal solution of the first subproblem (P1).

The System's Reliability Analysis. To consider the system's reliability, the study introduces reliability assessment, taking into account the stability of the Multi-Access Edge Computing (MEC) system under different network conditions and workloads. By introducing reliability parameters and optimizing resource allocation strategies, the system can maintain high reliability in the face of various uncertainties and challenges. This comprehensive approach, considering both task latency and system reliability, demonstrates superior performance of the DDRA algorithm in complex network environments.

For reliability analysis, several aspects can be considered:

System Stability: The advantage of the DDRA algorithm in average latency might contribute to improving the system's stability because lower average latency implies more timely task completion, resulting in overall more stable system performance.

Fault Tolerance: In the results of Fig. 5, DDRA performs relatively well, indicating its stronger fault tolerance in handling exceptional situations such as fluctuations in computing resources or network instability.

Consistency: DDRA demonstrates relatively lower average latency across different tasks, suggesting that the algorithm may provide more consistent performance under various task loads.

Overall, by comparing tasks optimized through DDRA with other algorithms, it can conclude the superiority of DDRA in terms of average latency and potentially better performance in some reliability indicators. This provides robust support for task processing and optimization in MEC environments.

5 Conclusion

This paper studies different 5G network scenarios combined with mobile edge computing, comprehensively analyzes the task QoS and user QoE requirements, and proposes efficient resource allocation strategies. This paper studies the resource distribution strategy of 5G network for MEC, but 5G technology is fully commercial and more complex network environment and higher requirements for resource allocation optimization in the future. After dynamic user equipment, wireless charging and green energy technology, consider network scenarios under high speed movement, design flexible device network switching and resource management strategy, enhance battery life, energy saving and dynamic resource management strategy.

In the computational offinstall algorithm for wireless bandwidth and computing resources, the computational tasks are indecomposable, where all computational tasks can only be performed locally or on the MEC server. However, there are many mobile applications that can make fine-grained division. For such computational tasks, you can uninstall some of the subtasks to the MEC server while another resides locally. A more detailed offloading decision should be reworked to theoretically further reduce the task execution delay.

Acknowledgement

This research is supported by the Dongguan City Sci-tech Commissioner Program in 2022 (Grant No. 20221800500822. Project Title: Work Piece Assembly System Development Research Based on MES); 2022 Special innovation Projects of Universities in Guangdong Province, Research on key technologies of Automatic Guided Vehicle (AGV) Based on Deep learning and machine vision (Grant No. 2022KTSCX327); Dongguan City Sci-tech Commissioner Program in 2022 (Grant No. 20221800500792. Project Title: Research on Intelligent Control of Nonlinear Control System in Five-axis Machining; Dongguan Polytechnic "Virtual simulation demonstration course of Logistics system planning and design" (Grant No. XNFZ202201); Guangdong Province Continuing Education Quality Improvement Project in 2022: "Storage and Distribution Practice" High Quality Continuing Education Online Course (Grant No. JXJYGC2022GX058); The 2023 Educational Science Planning Project (Special Project for Higher Education) "Research on Teaching Reform Strategies Assisted by ChatGPT-like Artificial Intelligence" (Grant No. 2023GXJK630); 2023 Guangdong Higher Vocational Education Teaching Reform Research and Practice Project: Research on the Performance Evaluation Model and Improvement Path of Industry-Education Integration in Higher Vocational Colleges (Grant No. 2023JG036); Dongguan Polytechnic's School-level Quality Engineering Project in 2024: Excellent Course on Production and Operation Practice (Grant No. JPKC202405); Dongguan Polytechnic's School-level Quality Engineering Project in 2024: Research and Practice of Higher Vocational Education Reform from the Perspective of ChatGPT-like Artificial Intelligence (Grant No. JGXM202416); The Humanities and Social Sciences Research and Planning Foundation Project of the Ministry of Education (Grant No. 21YJAZH034); Characteristic Innovation Project of Guangdong Universities (Grant No. 2020WTSCX259); Guangdong University Students' Science and Technology Innovation Cultivation Special Fund Funding Project (Grant No. pdjh2022a1024).

The Special for key fields of colleges and universities in Guangdong Province (Grant No. 2024ZDZX1092.).

2024 Innovation Team Project of Guangdong Ordinary Colleges and Universities (Innovation Team of Modern Service Industry and High-Quality Economic Development) (Grant No. 2024WCXTD062).

References

[1] B. Mathur, S. Satapathy, An Analytical Comparison of Mobile Application Development using Agile Methodologies, 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI), 2019.

[2] Q. Zhao, M. Gerla, Energy Efficiency Enhancement in 5G Mobile Wireless Networks, 2019 IEEE 20th International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM), 2019.

[3] M. Sung, J. Kim, E. Kim, S. Cho, Y. Won, B. Lim, RoF-Based Radio Access Network for 5G Mobile Communication Systems in 28 GHz Millimeter-Wave, Journal of Lightwave Technology 38(2)(2020) 409-420.

[4] X. Lin, B. Zhou, Y. Xia, Online Recursive Power Management Strategy Based on the Reinforcement Learning Algorithm With Cosine Similarity and a Forgetting Factor, IEEE Transactions on Industrial Electronics 68(6)(2021) 5013-5023.

[5] D. Yao, C. Yu, L. Yang, H. Jin, Using Crowdsourcing to Provide QoS for Mobile Cloud Computing, IEEE Transactions on Cloud Computing 7(2)(2019) 344-356.

[6] X. Chen, H. Wang, Y. Ma, X. Zheng, L. Guo, Self-adaptive Resource Allocation for Cloud-based Software Services Based on Iterative QoS Prediction Model, Future Generation Computer Systems 105(2020) 287-296.

[7] G. Huang, X. Liu, Y. Ma, X. Lu, Y. Zhang, Y. Xiong, Programming Situational Mobile Web Applications with Cloud-Mobile Convergence: An Internetware-Oriented Approach, IEEE Transactions on Services Computing 12(1)(2019) 6-19.

[8] X. Chen, J. Lin, Y. Ma, B. Lin, H. Wang, G. Huang, Self-adaptive Resource Allocation for Cloud-based Software Services Based on Progressive QoS Prediction Model, SCIENCE CHINA Information Sciences 62(11)(2019) 219101.

[9] G. Huang, Y. Ma, X. Liu, Y. Luo, X. Lu, M. Blake, Model-Based Automated Navigation and Composition of Complex Service Mashups, IEEE Transactions on Services Computing 8(3)(2015) 494-506.

[10] N. Abbas, Y. Zhang, A. Taherkordi, T. Skeie, Mobile Edge Computing: A Survey, IEEE Internet of Things Journal 5(1) (2018) 450-465.

[11] Y. Wang, S. Cao, H. Ren, J. Li, K. Ye, C. Xu, X. Chen, Towards cost-effective service migration in mobile edge: a q-learning approach, Journal of Parallel and Distributed Computing 146(1)(2020) 175-188.

[12] X. Chen, A. Li, X. Zeng, W. Guo, G. Huang, Runtime model based approach to IoT application development, Frontiers of Computer Science 9(4)(2015) 540-553.

[13] J. Chakareski, Viewport-Adaptive Scalable Multi-User Virtual Reality Mobile-Edge Streaming, IEEE Transactions on Image Processing 29(2020) 6330-6342.

[14] X. Liu, G. Huang, Q. Zhao, H. Mei, M. Blake, iMashup: a mashup-based framework for service composition, Science China Information Sciences 54(1)(2014) 1-20.

[15] G. Huang, H. Mei, F. Yang, Runtime recovery and manipulation of software architecture of component-based systems, Automated Software Engineering 13(2)(2006) 257-281.

[16] J. Feng, F. Yu, Q. Pei, J. Du, L. Zhu, Joint Optimization of Radio and Computational Resources Allocation in Blockchain-Enabled Mobile Edge Computing Systems, IEEE Transactions on Wireless Communications 19(6)(2020) 4321-4334.

[17] H. Song, G. Huang, F. Chauvel, Y. Xiong, Z. Hu, Y. Sun, H. Mei, Supporting runtime software architecture: A bidirectional-transformation-based approach, Journal of Systems and Software 84(5)(2011) 711-723.

[18] G. Huang, T. Liu, H. Mei, Z. Zheng, Z. Liu, G. Fan, Towards Autonomic Computing Middleware via Reflection. International Computer Software and Applications Conference, 2004.

[19] X. Shan, H. Zhi, P. Li, Z. Han, A Survey on Computation Offloading for Mobile Edge Computing Information, 2018 IEEE 4th International Conference on Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing, (HPSC) and IEEE International Conference on Intelligent Data and Security (IDS), 2018.

[20] Y. Zhou, P. Yeoh, C. Pan, Offloading Optimization for Low-Latency Secure Mobile Edge Computing Systems, IEEE Wireless Communications Letters 9(4)(2020) 480-484.

[21] C. Ho, C. King, Y. Chang, SCQ: Stage-Based, Context-Aware, QoE-Driven Power Optimization for Interactive Applications on Mobile Devices, 2019 20th IEEE International Conference on Mobile Data Management (MDM), 2019.

[22] Y. Jiang, D. Tsang, Delay-Aware Task Offloading in Shared Fog Networks, IEEE Internet of Things Journal 5(6)(2018) 4945-4956.

[23] J. Li, Y. Han, Optimal Resource Allocation for Packet Delay Minimization in Multi-Layer UAV Networks, IEEE Communications Letters 21(3)(2017) 580-583.

[24] Y. Chen, Y. Mao, H. Liang, S. Yu, Y. Wei, S. Leng, Data Poison Detection Schemes for Distributed Machine Learning, IEEE Access 8(2020) 7442-7454.

[25] Y. Kao, B. Krishnamachari, M. Ra, F. Bai, Hermes: Latency Optimal Task Assignment for Resource-constrained Mobile Computing, IEEE Transactions on Mobile Computing 16(11)(2017) 3056-3069.

[26] S. Mahmoodi, R. Uma, K.P. Subbalakshmi, Optimal Joint Scheduling and Cloud Offloading for Mobile Applications, IEEE Transactions on Cloud Computing 7(2)(2019) 301-313.

[27] H. Li, H. Xu, C. Zhou, X. Lü, Z. Han, Joint Optimization Strategy of Computation Offloading and Resource Allocation in Multi-Access Edge Computing Environment, IEEE Transactions on Vehicular Technology 69(9)(2020) 10214-10226.

[28] J. Kwak, L. Le, H. Kim, X. Wang, Two Time-Scale Edge Caching and BS Association for Power-Delay Tradeoff in Multi-Cell Networks, IEEE Transactions on Communications 67(8)(2019) 5506-5519.

[29] M. Liu, Y. Mao, S. Leng, S. Mao, Full-Duplex Aided User Virtualization for Mobile Edge Computing in 5G Networks, IEEE Access 6(2018) 2996-3007.

[30] H. Guo, J. Liu, J. Zhang, Computation Offloading for Multi-Access Mobile Edge Computing in Ultra-Dense Networks, IEEE Communications Magazine 56(8)(2018) 14-19.

[31] Y. Liu, P. Olmos, D. Mitchell, Generalized LDPC Codes for Ultra Reliable Low Latency Communication in 5G and Beyond, IEEE Access 6(2018) 72002-72014.

[32] M. Chen, Y. Hao, Task Offloading for Mobile Edge Computing in Software Defined Ultra-Dense Network, IEEE Journal on Selected Areas in Communications 36(3)(2018) 587-597.

[33] X. Wang, K. Wang, S. Wu, S. Di, H. Jin, K. Yang, S. Ou, Dynamic Resource Scheduling in Mobile Edge Cloud with Cloud Radio Access Network, IEEE Transactions on Parallel and Distributed Systems 29(11)(2018) 2429-2445.

[34] J. Zhang, X. Zhou, T. Ge, X. Wang, T. Hwang, Joint Task Scheduling and Containerizing for Efficient Edge Computing, IEEE Transactions on Parallel and Distributed Systems 32(8)(2021) 2086-2100.

[35] M. Adhikari, T. Amgoth, Heuristic-based load-balancing algorithm for Iaa S cloud, Future Generation Computer Systems 81(2018) 156-165.

[36] S. Ojha, H. Rai, A. Nazarov, Enhanced Modified HEFT Algorithm for Task Scheduling in Cloud Environment, 2020 2nd International Conference on Advances in Computing, Communication Control and Networking (ICACCCN), 2020.

[37] M. Zeng, W. Hao, O. Dobre, H. Poor, Delay Minimization for Massive MIMO Assisted Mobile Edge Computing, IEEE Transactions on Vehicular Technology 69(6)(2020) 6788-6792.

[38] X. Xiong, K. Zheng, L. Lei, L. Hou, Resource Allocation Based on Deep Reinforcement Learning in IoT Edge Computing, IEEE Journal on Selected Areas in Communications 38(6)(2020) 1133-1146.

[39] S. Pang, S. Wang, Joint Wireless Source Management and Task Offloading in Ultra-Dense Network, IEEE Access 8(2020) 52917-52926.

[40] S. Li, B. Li, W. Zhao, Joint Optimization of Caching and Computation in Multi-Server NOMA-MEC System via Reinforcement Learning, IEEE Access 8(2020) 112762-112771.

[41] L. Cheng, J. Zheng, J. Xiao, A distributed downlink resource allocation algorithm for dense small cell networks, 2015 International Conference on Wireless Communications & Signal Processing (WCSP), 2015.

[42] Y. Chen, Z. Liu, Y. Zhang, Y. Wu, X. Chen, L. Zhao, Deep Reinforcement Learning-Based Dynamic Resource Management for Mobile Edge Computing in Industrial Internet of Things, IEEE Transactions on Industrial Informatics 17(7)(2021) 4925-4934.