Hindawi
Complexity
Volume 2023, Article ID 9604454, 15 pages
https://doi.org/10.1155/2023/9604454

Research Article

An Efficient Algorithm for Resource Allocation in Mobile Edge Computing Based on Convex Optimization and Karush–Kuhn–Tucker Method

Kaijing Wang,[1] **Shelily F. Akhtar**,[2,3] **and Fahad Ahmed Al-Zahrani**[4]

[1]*School of Management and Information, Chuzhou City Vocational College, Chuzhou 239000, Anhui, China*
[2]*Electrical Engineering Department, Asia Pacific University, Dhaka, Bangladesh*
[3]*Islamic University Centre for Scientific Research, The Islamic University, Najaf, Iraq*
[4]*Department of Computer Engineering, Umm Al-Qura University, Mecca 24381, Saudi Arabia*

Correspondence should be addressed to Kaijing Wang; k.wang.cn@hotmail.com and Fahad Ahmed Al-Zahrani; falzahrani.ksa@outlook.com

Received 11 August 2023; Revised 20 November 2023; Accepted 2 December 2023; Published 24 December 2023

Academic Editor: Roberto Natella

Mobile edge computing (MEC) is receiving more attention than centralized cloud computing due to the massive increase in transmission and compute requirements in 5G vehicle networks. It offers a significant amount of processing and storage resources to the edge of networks, offloading applications from vehicle terminals that are computation-intensive and delay-sensitive. For devices with limited resources, it uses edge resources to provide computationally heavy operations while conserving energy. This paper proposes a novel approach for computing offloading in MEC. To effectively optimize the MEC resources, this paper proposes a novel algorithm. First, the joint optimization and service cache decision subproblems were determined from continuous and discrete variables. Then, the near-optimal solution is determined from the subproblems through convex optimization and Karush–Kuhn–Tucker method. Simulation results show that the proposed algorithm has better computational offloading and resource allocation performance as compared to existing algorithms.

1. Introduction

Many computing-intensive and delay-sensitive applications (such as autonomous driving, image identification, and natural language processing) have increasing needs due to the rapid expansion of mobile communications and the Internet of Things (IoT) [1]. One of the major problems that needs to be resolved immediately is how to reduce application processing delays because terminal devices with limited computing capacity would produce substantial task delays when processing such programs [2]. The issue of terminal computing limitations has been resolved by the development of mobile cloud computing (MCC) technology [3, 4]. The purpose of MCC is to increase the potential computing capabilities of terminal devices by extending the abundant computing resources of the cloud to resource-constrained terminal devices [5]. The vast communication distance between the terminal device and the cloud server results in a network delay, making it impossible to rely exclusively on MCC to meet the network needs of latency-sensitive applications. Mobile Edge Computing (MEC) is suggested as a new technology to address the aforementioned issues and support MCC [6, 7]. Base stations and associated edge servers make up the majority of mobile edge computing (MEC) systems. When terminal devices carry out computing tasks, edge nodes made up of base stations and servers can offer computing, communication, and data storage services to these terminals nearby,

lowering task completion latency and enhancing system performance [8, 9]. Currently, related studies on resource allocation based on MEC networks and task offloading have been conducted. In [10–12], the power consumption and system spectrum resources are collaboratively optimized with the goal of maximizing energy efficiency in various application scenarios. Researchers in [13, 14] optimized computation and spectrum resources while minimizing energy usage and delay, respectively. However, in real-world applications, edge servers' processing and storage capabilities are constrained, and excessive computing demands will unavoidably result in high computing and storage demands, which will have an impact on system performance. A fresh approach to resolving the aforementioned issues is offered by the resource allocation and unloading method implemented within the converged network of MCC and MEC. When a terminal device requests a task, it can be executed not only locally but also on an edge server or in the cloud thanks to the integrated network architecture of MCC and MEC. In order to meet various service requirements, the network design can offer terminals more effective and adaptable offloading services. Relevant research has demonstrated that the joint optimization of offload decision-making and resource allocation can enhance network performance under a three-level network design [15, 16].

To sum up, the three-tier network architecture made up of terminals, edge services, and clouds is thoroughly examined in this paper, along with aspects like wireless bandwidth resources, computation resources, and storage resources. A service cache is proposed as a joint optimization algorithm with resource allocation with the aim of minimizing the task delay of all terminal devices and carefully taking into account the impact of storage resources, computing resources, offloading decisions, and task offloading ratios on task delays. Following are the particular innovation points:

(i) A three-layer network architecture made up of cloud servers, edge servers, and multiusers is built. Each user's computing responsibilities can be partially offloaded to the local server using a ground server, edge server, or cloud server. We explain the paradigm for minimizing terminal task latency and discuss how user computing resources, edge service computing resources, storage resources, and bandwidth resources may affect the delay.

(ii) A joint optimization strategy for service caching and resource allocation is proposed. First, the continuous variable and discrete variable of the original problem are decoupled into two subproblems, that is, the service cache decision problem and the joint optimization problem of computing resources and communication resources. Secondly, the two subproblems are alternately iteratively optimized based on the reconstruction linearization technique (RLT), relaxation, and convex optimization methods. Finally, a suboptimal solution to the problem is obtained.

(iii) The simulation results demonstrate that the suggested technique has clear advantages in terms of convergence speed and performance over existing algorithms.

2. Related Work

With the development of 5G, data exchange between terminal devices and remote cloud servers may cause the backhaul network to be paralyzed. It is difficult to achieve millisecond-level computing and communication delays by relying solely on the cloud computing model. In order to solve the delay and energy consumption problems caused by the distance between cloud computing data centers and terminal devices, scholars have proposed to transfer cloud functions to the edge of the network. At the edge of the network, close to the terminal mobile device, MEC has emerged as a new network structure and computing paradigm that provides information technology services and has computing capabilities.

The authors in [6] investigate the energy cost reduction for mission-critical internet of things (IoT) in mobile edge computing (MEC) systems based on convex optimization. In order to lower transmission delay and increase the lifespan of the IoT devices' batteries, brief data packets are sent between the devices and the access points (APs). It is made clear how short-packet transmission affects the distribution of radio resources. We frame the challenge of minimizing energy costs analytically as a mixed-integer nonlinear programming (MINLP) problem, which is hard to solve optimally. To be more precise, the challenge stems mostly from the interplay between resource management across all IoT devices and binary offloading factors. A strategy for resource allocation and job offloading based on MEC is proposed in [7]. To address the issue of small-base station long-term performance optimization, a system model is constructed. In addition, a Lyapunov drift penalty technology-based energy consumption deficit queue is created to enable tiny base stations to meet energy consumption limits throughout the long-term optimization process. The suggested algorithm is capable of computing offloading decisions and iterating based on time slots. The results of the simulation experiment demonstrate that the algorithm clearly optimizes both energy consumption and time delay. Simultaneously, it is confirmed that the algorithm can be balanced by iterations in the computation of offloading, ensuring consistent and coordinated users and a low total cost. It does not, however, take into account work offloading and resource allocation under cloud-side collaboration, nor does it increase the scalability of edge servers and multichannel communication. It also does not integrate cloud computing with MEC. The challenge of minimizing energy usage for MEC systems was examined in [8]. With respect to the latency limitations, it jointly optimized the transmit power, local and edge computation resource allocation, as well as task offloading. The optimization problem concerning energy consumption has been framed as a nonconvex, NP-hard MINLP problem. To address this challenge, we have created a two-loop combined search and

SCA system, in which the outer loop searches for the best offloading options while the inner loop uses fixed offloading decisions to solve the optimization problem. However, this paper does not consider dynamic allocation and requires a large number of iterations.

The authors in [17] cooperatively optimize offloading choices and resources inside the centralized network when MEC and MCC are connected to lessen job delays across all terminals. The authors in [18] optimized the task offloading and resource allocation designs in order to maximize system income. The research mentioned above address the issue of task offloading and resource allocation jointly from various angles; however, they do not consider how task offloading and service cache coupling affect system performance. By storing databases, applications, function libraries, and related codes connected to computing jobs in edge services to support various applications, the term "service cache" is used. Existing MEC research can be split into two groups from the perspective of task offloading: total offloading of computing tasks and partial offloading of computing activities. The full-offload approach is appropriate for highly integrated or relatively basic tasks that cannot be split and can only be carried out by mobile devices or by edge servers as a whole, such as natural language processing, navigation, etc. For divide-and-conquer activities like image identification, augmented reality (AR) applications, etc., the partial offloading approach is appropriate. Such applications allow for the simultaneous or serial execution of tasks that have been broken down into many smaller jobs. The resource allocation based on the partial unloading model is more rational, the resource utilization is more efficient, and the usage rate is also higher than the complete unloading model. There are currently studies being done on resource allocation and service caching. The authors in [19] suggest an integer linear programming and random search approach to optimize the service cache resources between edge nodes in order to reduce the supplier cost. In [20], the placement of edge servers and user service requirements are collaboratively optimized with the goal of reducing the overall cost of service caching. The authors in [21] optimize the content cache of the edge server while minimizing delay and expense. Although task offloading and service cache optimization have enhanced system performance in the aforementioned research, they are all based on resource allocation optimization when tasks are inseparable and do not apply to activities that are separable.

The majority of complicated applications are made up of various execution components. For instance, real-time face recognition tasks must be broken down into 4 subtasks: face detection, image processing and feature extraction, face recognition, and face recognition. These sub-tasks demand various amounts of computational, storage, and connectivity resources. Therefore, it is crucial to figure out how to simultaneously optimize using service caching and partial task offloading. Using an open Markov queuing network model, researchers in [22] construct a service chain model, aim to reduce network response time under the constraints of storage and computing resources, construct an edge node service scheduling model, and examine cache decision-making and computing resources. When tasks are offloaded, joint optimization ignores resources like terminals and cloud servers that are available. According to [23], which is based on the partial offloading model, joint optimization research is conducted on service cache and task offloading under the constraints of computing resources, but it assumes that bandwidth resources are evenly distributed, resulting in tasks with a large amount of computing data getting less bandwidth resources and resulting in relatively low bandwidth resources. In addition, low-bandwidth processes with high delay receive greater bandwidth resources, wasting resources. The storage space of edge nodes is also not constrained by [23], hence in this study, the edge server has unlimited storage space and can cache all pertinent data needed for task execution.

Researchers in [24] proposed a total energy consumption minimization optimization method under time-sharing and computing time constraints and formulated a resource management strategy for heterogeneous input data arrival time and computing deadline. In [25], a hybrid fiber-wireless network is designed to provide support for the coexistence of centralized cloud and multi-access edge computing, and a fiber-wireless access network architecture and offloading scheme are proposed. In order to improve efficiency, researchers in [26–28] focus on the research of binary full offloading strategy to minimize delay or energy consumption. Researchers in [29] comprehensively considered the optimization of offloading decisions, channel allocation, and computing resource allocation schemes. By combining the advantages of genetic algorithms and particle swarm optimization to improve the performance and convergence speed of offloading algorithms, a suboptimal algorithm based on genetic algorithms and particle swarm optimization is proposed. However, these works only focus on the calculation and offloading process of wireless access networks, ignoring the dynamic changes in channel information and interference. Aiming at the offloading problem of multiuser computing, researchers in [30] proposed a D2D resource sharing scheme with optimal comprehensive benefits, which can significantly improve the performance of the offloading network. Researchers in [31] used D2D offloading computing, combined with user behavior and specific network operating conditions to develop a large-scale D2D-supported cellular network ESE evaluation framework. To analyze the SNR distribution and average rate, researchers in [32] proposed a D2D communication framework. In [33], the authors adopted spectrum sharing and provided a comparative analysis of the coverage performance of two different sharing modes.

Some recent works are dedicated to adopting distributed computing offloading schemes. For example, researchers in [33] proposed a game algorithm to jointly determine the computing offloading scheme, transmission scheduling rules, and pricing rules. This algorithm can maximize the benefits of the entire network and realize the game balance between mobile users. Researchers in [34] proposed a noncooperative exact potential game for edge node pair communication and computing resource contention, aiming at maximizing the long-term benefits of offloading.

Researchers in [35] used evolutionary game theory to optimize the subcarrier allocation scheme and allocated more subcarriers to subcarriers with better channel conditions to optimize energy efficiency. Researchers in [36] studied a new computing offloading problem in the MEC network: mobile devices can offload tasks to distributed computing nodes through D2D communication or the MEC server, and they deduced the offloading decision problem of mobile devices as a sequence game. However, it does not consider the service cache; that is, there is no binary file on the distributed computing nodes and edge servers [37], which is dependent on the cache and computing tasks. Due to the limited computing and storage resources of the edge server, it will inevitably affect the quality of service.

3. System Model

3.1. Centralized Mobile Edge Network Model. Figure 1 depicts a centralized mobile edge network. The cloud server, a single service node (made up of edge servers and base stations), and L terminals with computation and communication capabilities make up the centralized mobile edge network, as can be seen in Figure 1. In this context, the term "terminal201d often refers to terminals with computational capabilities, such as mobile phones, tablets, and cars[38, 39]. The formula for the number of terminals is $L = \{1, 2, \ldots, L\}$, $i \in L$. For the terminal to receive dependable communication, computation, and storage services, an optical fiber connection connects the cloud server and the service node [40]. This cloud server has a lot of computing and storage resources because it is mostly made up of cloud server clusters.

In the situation depicted in Figure 1, the service node can act as a coordinator, making storage and computing decisions in response to the task requests from the terminals, in addition to providing communication, computation, and storage resources for the terminal. Here, it is assumed that the edge server storage computing service's maximum storage capacity is C and its maximum processing capacity is F [41, 42]. Databases, data dictionaries, and related application codes make up the majority of the computing service data, where each application stands for a specific kind of computing work [43–74]. In addition, the service cache must be refreshed frequently because storage is a lengthy operation. In this study, the system is believed to run in discrete time intervals with interval lengths that correspond to service caching and task assignment decision update intervals. While ignoring the index of other time periods, this research examines one of them [45]. At the beginning of a time period, each terminal randomly requests a computing task, and one type of computing task corresponds to one type of computing service. It is assumed here that there is a computing service $\mathbb{K} = \{1, 2, \ldots, K\}$, $w_{A_i} = \left\{c_{A_i}^{\text{input}}, s_i\right\}$ means that user i requests the corresponding task of computing service A_i, $A_i \in K$, where $c_{A_i}^{\text{input}}$ represents the terminal i size of the input data for executing the related task of service A_i, and s_i represents the number of cycles required to execute 1 bit data. In order to facilitate the analysis, this paper makes the following reasonable assumptions:

(1) The service node is aware of the channel state information (CSI), the size of each task's input data, and the task's cycle count. An efficient resource allocation and offloading decision can be made based on the aforementioned hypotheses [46–48].

(2) Each application's calculation result data are considerably less than its calculation input data; therefore, the calculation results' backhaul link transmission latency is disregarded.

3.2. Communication Model. The spectrum bandwidth used in this research is B, and the orthogonal frequency division multiplexing method is used to distribute spectrum resources to terminal equipment. The uplink transmission rate of terminal i executing work connected to service A_i can be represented as in [49, 50]. Each terminal device will use uplink spectrum resources when offloading computational duties to service nodes.

$$r_{A_i}^{\text{up}} = \theta_{A_i} \text{Blog}\left[1 + \frac{\left|h_{i,B}\right|^2 P_{i,s} d^{-r}}{\text{BN}_0}\right]. \tag{1}$$

Among them, θ_{A_i} is the percentage of the uplink bandwidth that the terminal uploads data, $P_{i,s}$ is the transmit power of the terminal, $h_{i,B}$ is the channel fading coefficient between the base station and the terminal, d is the distance between the terminal and the base station which indicates the mobility of device, r is the path loss, and BN_0 is the noise power of the channel which is related to σ^2. N_0 is the power spectral density of Gaussian white noise [51].

3.3. Service Cache Model. The service nodes can do relevant computing tasks if the edge server contains computing services. If not, the tasks will be given to the cloud server to complete. The edge server must make judgments using the service data that has been cached because it does not have enough storage to handle all of the terminal devices' compute requests [52]. Here, the storage decision of the server is expressed as $X = \left\{x_{A_1}, x_{A_2}, \ldots, x_{A_i}\right\}$, $x_{A_i} \in \{0, 1\}$, $A_i \in \mathbb{K}, i \in \mathscr{L}$; x_{A_i} indicates whether the server caches the A_i requested by the terminal device, $x_{A_i} = 1$ indicates that the server caches the computing service A_i, $x_{A_i} = 0$ indicates that the server does not cache the corresponding computing service A_i. The required storage size for each service A_i is c_{A_i}, and the edge server caching decision is limited by the storage capacity, i.e., $\sum_{i \in \mathscr{L}} x_{A_i} c_{A_i} \leq C, A_i \in \mathbb{K}$.

3.4. Computational Task Offloading Model. A computational workload may be divided into a number of smaller subtasks via partial task offloading [53]. This article carries out research using the unloading model, which states that programs that partition codes, such as image recognition, are separated into several modules, with the result of one module serving as the input for the next module. This article then partially unloads the model and describes the task execution delay model.

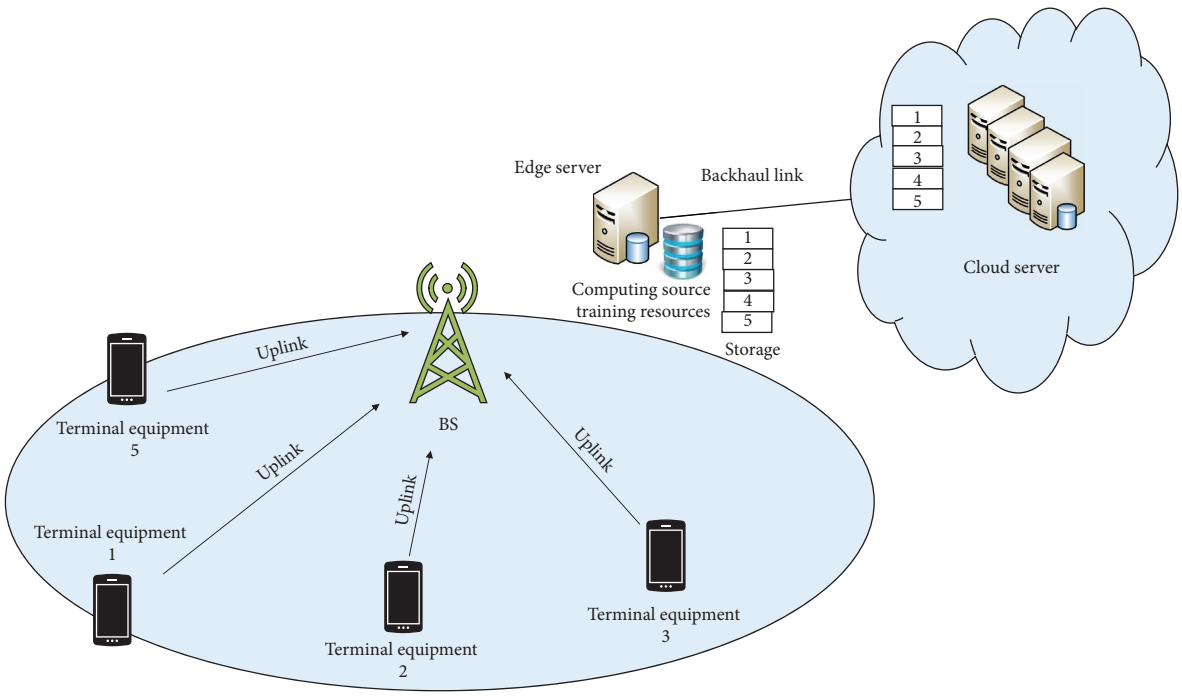

Figure 1: System model.

(1) When the computing service A_i is stored on the edge server, the task can be executed on the edge server. At this time, the task execution delay is analyzed as follows.

If some tasks are offloaded locally, the percentage of task unloading is represented by β_{A_i}, and the local unloading amount of tasks is represented by $c_{A_i}^{\text{local}}$, that is, $c_{A_i}^{\text{local}} = c_{A_i}^{\text{input}}\beta_{A_i}$, $0 \le \beta_{A_i} \le 1$, $\forall i \in \mathscr{L}$, $A_i \in \mathbb{K}$, the local computing capability is f_i^{local} [54, 55], then the local computing delay of the task is expressed as follows:

$$t_{A_i}^{\text{local}} = \frac{c_{A_i}^{\text{local}} s_i}{f_i^{\text{local}}}. \tag{2}$$

If some tasks are offloaded on the service node, when the task is offloaded to the edge server, the task offloading percentage is expressed as $1 - \beta_{A_i}$, and the data size of the corresponding task offloaded to the edge server is expressed as $c_{A_i}^{\text{mec}} = c_{A_i}^{\text{input}}(1 - \beta_{A_i})$ [56]. The computing resource allocated to the terminal device is expressed as f_i^{mec} and the execution delay of the task at the edge server is expressed as follows:

$$t_{A_i}^{\text{mec}} = \frac{c_{A_i}^{\text{mec}} s_i}{f_i^{\text{mec}}}. \tag{3}$$

The terminal device will use the uplink to transmit data after the task has been sent to the edge server; at this point, the uplink's transmission delay is expressed as follows:

$$t_{A_i}^{\text{up}} = \frac{c_{A_i}^{\text{mec}}}{r_{A_i}^{\text{up}}}. \tag{4}$$

In conclusion, the overall task execution latency is represented as follows when the computing service is kept in the service center:

$$U_{A_i}^{\text{local\_mec}} = t_{A_i}^{\text{local}} + t_{A_i}^{\text{up}} + t_{A_i}^{\text{mec}}. \tag{5}$$

(2) When the computing service A_i is not stored in the edge server, the computing tasks will be partially offloaded locally and in the cloud.

If some tasks are offloaded locally, the percentage of task unloading is expressed by α_{A_i}, and the local unloading amount of tasks is expressed by $c_{A_i}^{\text{local\_nocach}}$, that is, $c_{A_i}^{\text{local\_nocach}} = c_{A_i}^{\text{input}}\alpha_{A_i}$, $0 \le \alpha_{A_i} \le 1$, $\forall i \in \mathscr{L}$, $\forall k \in \mathbb{K}$. The local computing power is f_i^{local} [57], so the local computing delay of the task is expressed as follows:

$$t_{A_i}^{\text{local\_nocach}} = \frac{c_{A_i}^{\text{local\_nocach}} s_i}{f_i^{\text{local}}}. \tag{6}$$

If the task is offloaded to the cloud server, the offload percentage is expressed as $1 - \alpha_{A_i}$, where the data size of the task offloaded to the edge server is expressed as cloud $c_{A_i}^{\text{cloud}} = c_{A_i}^{\text{input}}(1 - \alpha_{A_i})$ [58]. A task execution delay is only connected to the transmission delay, specifically the uplink transmission delay and the backhaul link transmission delay, because the cloud server has a large amount of computational resources [59]. As a result, the task's transmission delay is expressed as follows when it is carried out on the cloud server:

$$t_{A_i}^{\text{cloud}} = \frac{c_{A_i}^{\text{cloud}}}{R} + \frac{c_{A_i}^{\text{cloud}}}{r_{A_i}^{\text{up}}}. \tag{7}$$

Among them, R is the transmission rate between the edge service and the cloud server. Therefore, when the computing service is stored in the cloud server, the total delay of task execution is expressed as follows:

$$U_{A_i}^{\text{local\_cloud}} = t_{A_i}^{\text{local\_nocach}} + t_{A_i}^{\text{cloud}}. \tag{8}$$

4. Problem Description

Storage changes are minimal since they are part of a long-term statistical process, but computation and communication resources must be adjusted in real-time to meet the needs of various users [60]. This study takes into account the current time period's joint optimization of computation, communication, and storage resources based on the aforementioned features [61]. The main goal of this paper is to jointly optimize the service cache decision x, uplink spectrum resource θ, task partial offload percentage α and β, and server computing resource f under the constraints of computing, communication, and storage resources to minimize the execution of all terminal devices total delay of the task [62]. The optimization objective can be expressed as follows:

$$\begin{aligned}
\text{P0: } & \min_{x,\theta,\alpha,\beta,f} \sum_{i=1}^{L} x_{A_i} U_{A_i}^{\text{local_mec}} + \left(1 - x_{A_i}\right) U_{A_i}^{\text{local_cloud}} \\
\text{s.t. C1: } & \sum_{i \in \mathscr{L}} x_{A_i} f_i^{\text{mec}} \le F \\
\text{C2: } & \sum_{i \in \mathscr{L}} \theta_{A_i} \le 1 \\
\text{C3: } & 0 \le \alpha_{A_i} \le 1,\ 0 \le \beta_{A_i} \le 1, \forall i \in \mathscr{L} \\
\text{C4: } & x_{A_i} \in \{0,\ 1\}, \forall i \in \mathscr{L} \\
\text{C5: } & \sum_{i \in \mathscr{L}} x_{A_i} c_{A_i} \le C,
\end{aligned} \tag{9}$$

P0 shows that the constraint condition C1 indicates that the total computing resources allotted to terminal devices are less than or equal to the server's maximum computing capacity, and C2 indicates that the total spectrum bandwidth is less than or equal to the frequency allotted to terminal devices. C3 stands for the task offload ratio [63, 64], C4 for the service caching choice, and C5 for the maximum amount of computing services that the edge server may cache. In addition, P0 requires the development of a novel optimization approach because it is a challenging mixed-integer nonlinear programming problem made up of discrete and continuous variables.

5. Algorithm Description

This work suggests a joint optimization method of service caching and resource allocation for task offloading and service caching in order to address the aforementioned issues. This strategy is referred to as the joint optimization strategy of service caching and resource allocation. First, separate the original problem's continuous and discrete variables into two distinct issues: the service cache decision issue and the combined optimization issue for communication and compute resources [65, 66]. Second, the fixed cache choice is used to resolve the problem of allocating communication and computation resources. After receiving the optimal resource allocation variables, optimize the cache decision once more. Finally, loop iteration yields the answer to the original problem.

5.1. Joint Optimization of Computing and Communication Resources. Prior to jointly optimizing compute and communication resources, cache decisions are initialized. P0 becomes a function made up of continuous variables like compute resource, bandwidth resource, and task offload ratio under the specified cache decision condition. The next step is to solve the continuous variables. After the initial cache decision is given as $x^{(0)}$ [67, 68], the multiterminal delay optimization function is expressed as follows:

$$\begin{aligned}
y(\alpha, \beta, \theta, f) = & \sum_{i=1}^{L} x_{A_i}^{(0)} \left[\frac{\left\{ \left(c_{A_i}^{\text{input}} \beta_{A_i} s_i / f_i^{\text{local}} \right) + \left(c_{A_i}^{\text{input}} s_i \left(1 - \beta_{A_i}\right) / f_i^{\text{mec}} \right) \right\} / c_{A_i}^{\text{input}} \left(1 - \beta_{A_i}\right)}{c_{A_i}^{\text{input}} \left(1 - \beta_{A_i}\right) / \theta_{A_i} B \log\left[1 + \left(\left|h_{i,B}\right|^2 P_{i,s} d^{-r} / \sigma^2 \right)\right]} \right] \\
& + \left(1 - x_{A_i}^{(0)}\right) \left[\frac{c_{A_i}^{\text{input}} \alpha_{A_i} s_i}{f_i^{\text{local}}} + \frac{c_{A_i}^{\text{input}} \left(1 - \alpha_{A_i}\right)}{\theta_{A_i} B \log\left[1 + \left(\left|h_{i,B}\right|^2 P_{i,s} d^{-r} / \sigma^2 \right)\right]} + \frac{c_{A_i}^{\text{input}} \left(1 - \alpha_{A_i}\right)}{R} \right].
\end{aligned} \tag{10}$$

The corresponding optimization problem is expressed as follows:

$$
\begin{aligned}
P1:\ & \min_{\alpha,\beta,\theta,f} y(\alpha,\beta,\theta,f)\\
\text{s.t. } C6:\ & \sum_{i\in\mathscr{L}} x_{A_i}^{(0)} f_i^{\text{mec}} \le F\\
C2:\ & \sum_{i\in\mathscr{L}} \theta_{A_i} \le 1\\
C3:\ & 0\le \alpha_{A_i}\le 1,\ 0\le \beta_{A_i}\le 1, \forall i\in\mathscr{L}.
\end{aligned}
\tag{11}
$$

After the caching decision is determined, its corresponding constraint condition changes simultaneously, and its constraint condition will change from C1 to C6. In order to avoid division by zero, constants ε_1 and ε_2 are added to the variables θ_{A_i} and f_i^{mec} [69], which can be expressed as $\overline{\theta}_{A_i} = (\theta_{A_i}+\varepsilon_1)^{-1}$ and $f_i^{-\text{mec}} = (f_i^{\text{mec}}+\varepsilon_2)^{-1}$. Finally, substitute this variable into formula (10) and the corresponding constraints to obtain new functions and constraints [70]. The corresponding question is expressed as follows:

$$
\begin{aligned}
P2:\ & \min_{\alpha,\beta,\theta,f} \sum_{i=1}^{L} x_{A_i}^{(0)}\left[\frac{c_{A_i}^{\text{input}}\beta_{A_i}s_i}{f_i^{\text{local}}} + c_{A_i}^{\text{input}}s_i\left(1-\beta_{A_i}\right)f_i^{-\text{mec}} + \frac{c_{A_i}^{\text{input}}\left(1-\beta_{A_i}\right)\overline{\theta}_{A_i}}{B\log\left[1+\left(\left|h_{i,B}\right|^2P_{i,s}d^{-r}/\sigma^2\right)\right]}\right]\\
&+\left(1-x_{A_i}^{(0)}\right)\left[\frac{c_{A_i}^{\text{input}}\alpha_{A_i}s_i}{f_i^{\text{local}}} + \frac{c_{A_i}^{\text{input}}\left(1-\alpha_{A_i}\right)\overline{\theta}_{A_i}}{B\log\left[1+\left(\left|h_{i,B}\right|^2P_{i,s}d^{-r}/\sigma^2\right)\right]} + \frac{c_{A_i}^{\text{input}}\left(1-\alpha_{A_i}\right)}{R}\right]\\
\text{s.t. } C7:\ & \sum_{i\in\mathscr{L}} \frac{1}{f_i^{-\text{mec}}} - \varepsilon_2 \le F, k\in\mathbb{K}\\
C8:\ & \sum_{i\in\mathscr{L}} \frac{1}{\overline{\theta}_{A_i}} - \varepsilon_1 \le 1, k\in\mathbb{K}\\
C3:\ & 0\le \alpha_{A_i}\le 1,\ 0\le \beta_{A_i}\le 1, \forall i\in\mathscr{L}.
\end{aligned}
\tag{12}
$$

It can be seen from P2 that there is a product term for multiplying two variables in formula (12), so this problem is a nonconvex optimization problem. In order to solve P2, the above problem needs to be transformed again.

5.2. RLT-Based Problem Transformation. Since P2 contains product terms $\alpha_{A_i}\overline{\theta}_{A_i}$, $\beta_{A_i}f_i^{-\text{mec}}$, and $\beta_{A_i}\overline{\theta}_{A_i}$, the above problem needs to be transformed by RLT [71]. First, introduce three auxiliary variables μ_{A_i}, η_{A_i}, χ_{A_i}, where $\mu_{A_i} = \alpha_{A_i}\overline{\theta}_{A_i}$, $\eta_{A_i} = \beta_{A_i}f_i^{-\text{mec}}$, $\chi_{A_i} = \beta_{A_i}\overline{\theta}_{A_i}$. Substituting new variables into formula (12), P2 can be rewritten as follows:

$$
\begin{aligned}
\text{P2:}\ & \min_{\alpha,\beta,\overline{\theta},\overline{f}} \sum_{i\in\mathscr{L}}\left[x_{A_i}^{(0)}\frac{c_{A_i}^{\text{input}}\beta_{A_i}s_{A_i}}{f_i^{\text{local}}} + c_{A_i}^{\text{input}}s_i^m f_i^{-\text{mec}} - s_i\eta_{A_i}\frac{c_{A_i}^{\text{input}}\overline{\theta}_{A_i} - c_{A_i}^{\text{input}}\chi_{A_i}}{B\log\left[1+\left(\left|h_{i,B}\right|^2P_{i,s}d^{-r}/\sigma^2\right)\right]}\right]\\
&+\left(1-x_{A_i}^{(0)}\right)\left[\frac{c_{A_i}^{\text{input}}\alpha_{A_i}s_i}{f_i^{\text{local}}} + \frac{c_{A_i}^{\text{input}}\overline{\theta}_{A_i} - c_{A_i}^{\text{input}}\mu_{A_i}}{B\log\left[1+\left(\left|h_{i,B}\right|^2P_{i,s}d^{-r}/\sigma^2\right)\right]} + \frac{c_{A_i}^{\text{input}}\left(1-\alpha_{A_i}\right)}{R}\right].
\end{aligned}
\tag{13}
$$

For the newly added variables μ_{A_i}, η_{A_i}, χ_{A_i}, and constraints such as $0\le\alpha_{A_i}\le 1, 0\le\beta_{A_i}\le 1$, $1/F+\varepsilon_2\le f_i^{-\text{mec}}\le 1/\varepsilon_2$, $1/1+\varepsilon_1\le\overline{\theta}_{A_i}\le 1/\varepsilon_1$, the RLT factor product constraint condition of μ_{A_i} can be obtained as follows:

$$\begin{cases} \left\{\left[\alpha_{A_i}-0\right]\left[\overline{\theta}_{A_i}-\dfrac{1}{1+\varepsilon_1}\right]\geq 0\right\}_{\mathrm{LS}}, \\ \left\{\left[\alpha_{A_i}-0\right]\left[\dfrac{1}{\varepsilon_1}-\overline{\theta}_{A_i}\right]\geq 0\right\}_{\mathrm{LS}}, \\ \left\{\left[1-\alpha_{A_i}\right]\left[\overline{\theta}_{A_i}-\dfrac{1}{1+\varepsilon_1}\right]\geq 0\right\}_{\mathrm{LS}}, \\ \left\{\left[1-\alpha_{A_i}\right]\left[\dfrac{1}{\varepsilon_1}-\overline{\theta}_{A_i}\right]\geq 0\right\}_{\mathrm{LS}}. \end{cases} \tag{14}$$

Among them, $\{\cdot\}_{\mathrm{LS}}$ represents the linearization step [72]. Substituting $\mu_{A_i}=\alpha_{A_i}\overline{\theta}_{A_i}$ into formula (14), we can get

$$\begin{cases} \mu_{A_i}-\dfrac{\alpha_{A_i}}{1+\varepsilon_1}\geq 0, \\ \dfrac{\alpha_{A_i}}{\varepsilon_1}-\mu_{A_i}\geq 0, \\ \overline{\theta}_{A_i}-\dfrac{1}{1+\varepsilon_1}-\mu_{A_i}+\dfrac{\alpha_{A_i}}{1+\varepsilon_1}\geq 0, \\ \dfrac{1}{\varepsilon_1}-\overline{\theta}_{A_i}-\dfrac{\alpha_{A_i}}{\varepsilon_1}+\mu_{A_i}\geq 0. \end{cases} \tag{15}$$

Among them, $\forall i\in\mathscr{L}$. Similarly, the RLT factor product constraint condition corresponding to η_{A_i} is

$$\begin{cases} \left\{\left[\beta_{A_i}-0\right]\left[f_i^{-\mathrm{mec}}-\dfrac{1}{F+\varepsilon_2}\right]\geq 0\right\}_{\mathrm{LS}}, \\ \left\{\left[\beta_{A_i}-0\right]\left[\dfrac{1}{\varepsilon_2}-f_i^{-\mathrm{mec}}\right]\geq 0\right\}_{\mathrm{LS}}, \\ \left\{\left[1-\beta_{A_i}\right]\left[f_i^{-\mathrm{mec}}-\dfrac{1}{F+\varepsilon_2}\right]\geq 0\right\}_{\mathrm{LS}}, \\ \left\{\left[1-\beta_{A_i}\right]\left[\dfrac{1}{\varepsilon_2}-f_i^{-\mathrm{mec}}\right]\geq 0\right\}_{\mathrm{LS}}. \end{cases} \tag{16}$$

Substituting $\eta_{A_i}=\beta_{A_i}f_i^{-\mathrm{mec}}$ into formula (16), we can get

$$\begin{cases} \eta_{A_i}-\dfrac{\beta_{A_i}}{F+\varepsilon_2}\geq 0, \\ \dfrac{\beta_{A_i}}{\varepsilon_2}-\eta_{A_i}\geq 0, \\ f_i^{-\mathrm{mec}}-\dfrac{1}{F+\varepsilon_2}-\eta_{A_i}+\dfrac{\beta_{A_i}}{F+\varepsilon_2}\geq 0, \\ \dfrac{1}{\varepsilon_2}-f_i^{-\mathrm{mec}}-\dfrac{\beta_{A_i}}{\varepsilon_2}+\eta_{A_i}\geq 0. \end{cases} \tag{17}$$

Similarly, the RLT factor product constraint condition corresponding to χ_{A_i} is

$$\begin{cases} \left\{\left[\beta_{A_i}-0\right]\left[\overline{\theta}_{A_i}-\dfrac{1}{1+\varepsilon_1}\right]\geq 0\right\}_{\mathrm{LS}}, \\ \left\{\left[\beta_{A_i}-0\right]\left[\dfrac{1}{\varepsilon_1}-\overline{\theta}_{A_i}\right]\geq 0\right\}_{\mathrm{LS}}, \\ \left\{\left[1-\beta_{A_i}\right]\left[\overline{\theta}_{A_i}-\dfrac{1}{1+\varepsilon_1}\right]\geq 0\right\}_{\mathrm{LS}}, \\ \left\{\left[1-\beta_{A_i}\right]\left[\dfrac{1}{\varepsilon_1}-\overline{\theta}_{A_i}\right]\geq 0\right\}_{\mathrm{LS}}. \end{cases} \tag{18}$$

Substituting $\chi_{A_i}=\beta_{A_i}\overline{\theta}_{A_i}$ into formula (18), we can get

$$\begin{cases} \chi_{A_i}-\dfrac{\beta_{A_i}}{1+\varepsilon_1}\geq 0, \\ \dfrac{\beta_{A_i}}{\varepsilon_1}-\chi_{A_i}\geq 0, \\ \overline{\theta}_{A_i}-\dfrac{1}{1+\varepsilon_1}-\chi_{A_i}+\dfrac{\beta_{A_i}}{1+\varepsilon_1}\geq 0, \\ \dfrac{1}{\varepsilon_1}-\overline{\theta}_{A_i}-\dfrac{\beta_{A_i}}{\varepsilon_1}+\chi_{A_i}\geq 0. \end{cases} \tag{19}$$

After the above transformation, the new optimization problem P3 is obtained as

$$\begin{aligned} P3: &\min_{\alpha,\beta,\overline{\theta},\overline{f},\mu,\eta,\chi} y''(\alpha,\beta,\overline{\theta},\overline{f},\mu,\eta,\chi) \\ &\text{s.t. } C3, C7, C8 \\ &C9: \text{formula (15)} \\ &C10: \text{formula (17)} \\ &C11: \text{formula (19)}, \end{aligned} \tag{20}$$

P3 is a convex optimization problem, so it can be solved by Lagrangian multiplier method as well as gradient method [73, 74]. After substituting the obtained solutions $\overline{\theta}$ and $\overline{f}$ into $\overline{\theta}_{A_i} = (\theta_{A_i} + \varepsilon_1)^{-1}$ and $f_i^{-\text{mec}} = (f_i^{\text{mec}} + \varepsilon_2)^{-1}$, the solution of the original variables can be obtained.

5.3. Solution to Cache Optimization Problem. After obtaining the solution of P1, the cache decision problem is solved below. First, substituting the obtained resource allocation scheme $\alpha^{(0)}, \beta^{(0)}, \theta^{(0)}, f^{(0)}$ into P0, the cache decision problem P4 is obtained as follows:

$$\begin{aligned} P4: &\min_{x} \sum_{i=1}^{L} x_{A_i} U_{A_i}^{\text{local\_mec}} + \left(1 - x_{A_i}\right) U_{A_i}^{\text{local\_cloud}} \\ &\text{s.t. } C4: x_{A_i} \in \{0, 1\}, \forall i \in \mathscr{L} \\ &C5: \sum_{i \in \mathscr{L}} x_{A_i} c_{A_i} \le C, \end{aligned} \tag{21}$$

P4 is a 0-1 problem, which can be solved by the branch and bound method. However, the branch and bound method requires 2^k operations in the worst case, and its complexity is acceptable when the amount of input data is small. It becomes more challenging to find an approximation optimal solution quickly as the amount of input data increases [75]. In order to reduce the time complexity, this paper relaxes the integer variable x_{A_i} into a continuous variable of $0 \le x_{A_i} \le 1$. At this point, P4 becomes a convex optimization problem, which can be solved by the interior point method and the KKT (Karush–Kuhn–Tucker) condition [76]. The continuous variable is then restored by rounding, and the threshold is simply set to 0.5.

5.4. Pseudocode of Algorithm. Based on the aforementioned analysis, it is clear that the edge server executes the optimization strategy, obtains the entirety of the information for the current access terminal in stages [77], and then sends the cache decision and resource allocation results to the terminal device, minimizing the execution delay of every device task. The detailed procedures are described in Algorithm 1.

6. Simulation Results

This section runs simulation tests on MATLAB 2017b and evaluates how well the suggested optimization algorithm performs. Additionally, this work mimics the following baseline techniques for performance comparison.

Complete offload cloud (COC) strategy: The cloud server houses all of the service-related task data. The process has now been completed on the cloud server, and each terminal device has received an equal share of the uplink bandwidth resources [36].

Random resource allocation and caching (RCRA) strategy Edge servers with limited storage employ random caching techniques to maximize the amount of service material they can cache and to maximize computational and bandwidth resources based on [55].

Local offload completely (LOC) strategy: Task-related service data is presumptively stored at the local terminal, and all tasks are processed locally at the terminal [57].

6.1. Simulation Parameters. In the simulation experiment, the number of terminal devices is 10, which are randomly distributed in a square area with a side length of 500 m. $h_{i,B}$ obeys the exponential distribution with a mean value of 1 [58], the computing task data size is denoted as c_i and obeys the uniform distribution of [100, 200] kB, and s_i is set to 1500 cycle/bit. The simulation parameter settings are shown in Table 1.

6.2. Algorithm Complexity and Convergence Analysis. Since the original problem P0 is a mixed integer nonlinear programming problem, the complexity of solving the problem by the exhaustive method is $O(2^n)$, showing exponential growth. The proposed algorithm converts the original problem P0 into P3 and P4, and P3 is a standard convex optimization problem [78], which can be solved by the interior point method, and the complexity of the interior point method is $O(n^3)$ [79–82]. P4 is a 0-1 discrete variable convex optimization problem after relaxation, and its algorithm complexity is $O(n^3)$. In addition, the complexity of the number of alternate iterations is $O(n)$ [79]. Therefore, the overall complexity of the proposed algorithm is the product of interior point method and that of the number of alternate iterations, i.e., $O(n^4)$. Compared with the exhaustive method, the complexity of the proposed algorithm is greatly reduced.

Figure 2 depicts the algorithm convergence process. The value produced using the exhaustive procedure is used as the ideal answer in this paper. The exhaustive technique is typically utilized in small-scale applications because it takes a long time. Figure 2 shows that the suggested algorithm swiftly converges and approaches the ideal solution after just 25 iterations, demonstrating that its performance is very similar to that of the exhaustive technique and that it is capable of obtaining an approximate optimal solution. This validates that the proposed algorithm can be tuned to the desired convergence value by configuring the specific suboptimal value of iterations and other parameters.

6.3. Simulation Results and Analysis. The simulation results in this section were realized with MATLAB 2017b on a desktop computer equipped with an Intel Core i5 9400F 2.9 GHz processor and 16 GB RAM. Under the conditions of $N = 10$, $C = 300$ GB, and $R = 1$, the influence of the computing power and storage capacity of the edge server on the task execution delay is shown in Figures 3 and 4, respectively.

Initialize: Caching strategy of edge services $x_{A_i}^{(0)}$, $\forall i \in \mathscr{L}$, terminal transmit power $p_{i,s}$, task input data size input $c_{A_i}^{\text{input}}$, calculation required for each bit of input data quantity s_i, the maximum number of iterations $t_{\max}$, and the algorithm termination condition ξ

(1) Convert the original issue into a continuous nonlinear issue to produce the new issue P2 based on the established service caching scheme.

(2) Transform the nonlinear problem into convex optimization problem P3 by relaxing P2 with RLT

(3) Use the interior point method to solve P3, and obtain the resource allocation scheme $\alpha, \beta, \overline{\theta}, \overline{f}, \mu, \eta, \chi$ under the condition of known cache decision $x_{A_i}^{(0)}$

(4) Substitute the solution obtained in step 3 into P4 and obtain the corresponding unloading decision x_{A_i} through relaxation method and convex optimization method

(5) Obtain the difference between the objective function in step 3 and step 4, if the difference is less than ξ or the number of iterations reaches the maximum value $t_{\max}$, stop the calculation. Otherwise, perform loop iterations on steps 3 and 4.

Algorithm 1: Joint resource allocation and service caching.

Table 1: Simulation parameters.

Parameter	Value
Number of terminal devices	10
Base station communication range side length d	500 m
Computing task input data $c_{A_i}^{\text{input}}$	U[100, 200] kB
Calculation service type	10 K
Uplink bandwidth B	20 MHz
Mobile terminal transmit power $P_{i,s}$	23 dBm
Path loss r	4
Noise power σ^2	−174 dBm/Hz
Computing power of the edge server F	30000 Megacycle
Computing power of the terminal f_i^{local}	N(500, 50) Megacycle
Service cache required to execute a task c_{A_i}	U[30,80] GB
Maximum storage capacity of the edge server C	500 GB
Transmission rate between the server and the remote cloud R	1 Mbps
Number of terminal equipment L	10

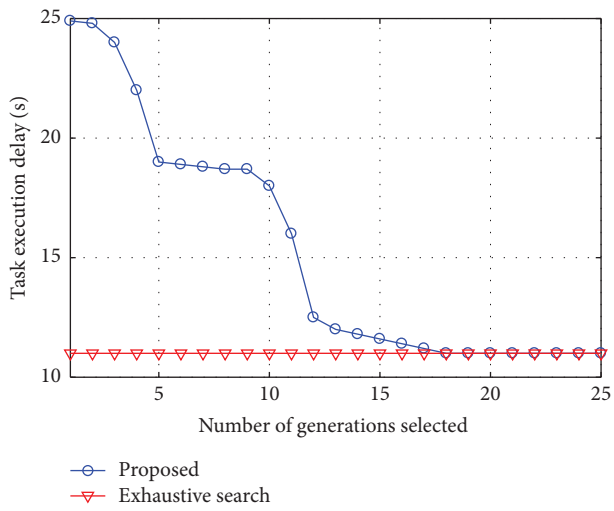

Figure 2: Algorithm convergence process.

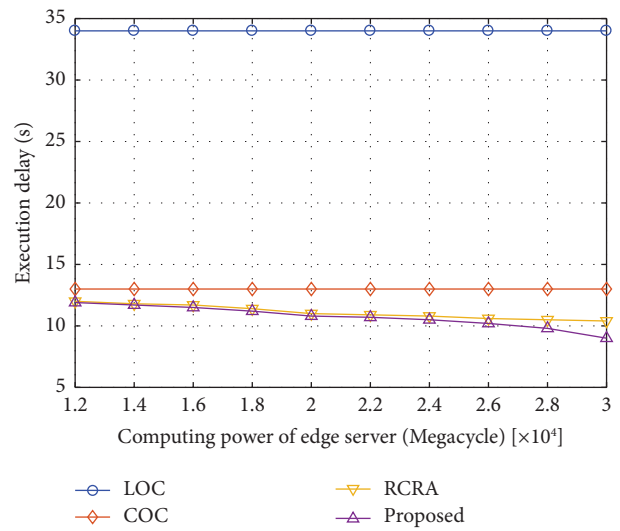

Figure 3: The influence of the computing power of the edge server on the task execution delay.

As observed in Figure 3, the task execution delays of the proposed optimization approach and the RCRA strategy gradually reduce as the processing capacity of the edge server gradually increases, whereas the task execution delays of the LOC strategy and the COC strategy remain constant. This is so that all jobs in the LOC strategy continue to execute with the same delay because the task delay of the LOC strategy solely depends on the local computing power, and the task execution delay has nothing to do with the resources controlled by the edge server. When a task is carried out using the COC strategy, the task execution delay is solely determined by the available wireless bandwidth and the backhaul link's transmission rate. As a result, the task

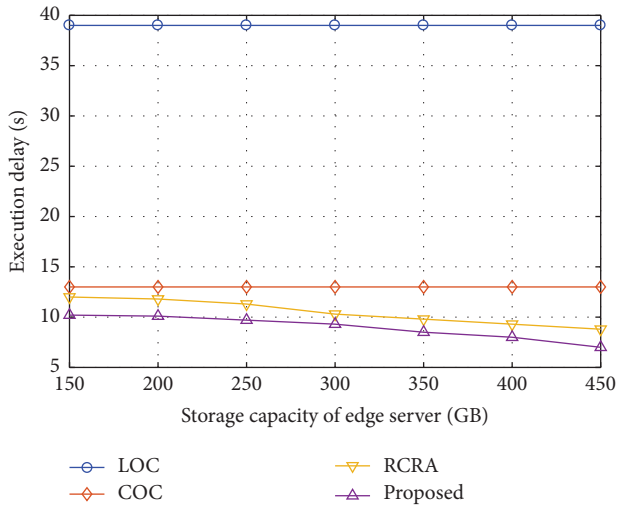

Figure 4: Effect of storage capacity of the edge server on task execution delay.

execution delay of the COC strategy does not change as the computing power of the edge server increases. Furthermore, it is also evident from Figure 3 that the suggested optimization strategy performs marginally better than the RCRA strategy in terms of speed. This is because the JSCRA optimization strategy reasonably optimizes the service cache, allowing for the execution of more tasks on the edge server.

Figure 4 shows that the task execution delays of the proposed approach and the RCRA method steadily reduce as the storage capacity of the edge server increases, whereas the task execution delays of the LOC strategy and the COC strategy remain constant. The proposed technique has a 10% reduction in job execution latency compared to the RCRA strategy and a 30% reduction in task execution delay compared to the COC strategy. The COC strategy's task execution time is tied to bandwidth resources and backhaul connection transmission rates, whereas the LOC strategy's job execution delay is only related to local computing capacity. As a result, neither the LOC strategy nor the COC strategy's job execution delay is affected by the edge server's storage capacity. In addition, Figure 4 demonstrates that the performance advantages of the proposed method and the RCRA technique rise when the storage capacity of the edge server increases in comparison to the COC strategy. This is due to the fact that when an edge server's storage capacity is limited, only a portion of its computational services may be cached, and the remainder must be saved on a distant cloud server. Since the majority of jobs will currently be carried out via cloud services, the proposed approach and RCRA method perform marginally better in terms of task execution delay than the COC strategy. However, as storage capacity rises, the amount of service content that the edge server can cache grows, and the percentage of operations that must be executed remotely gradually declines. Since the edge server's computational capacity is currently being used to its full potential, the performance of the suggested approach and the RCRA strategy outperforms the COC algorithm by a wide margin.

Figure 5 depicts the effect of task input data size on task execution delay. Figure 5 demonstrates that the performance of the suggested technique outperforms previous strategies and that task execution delays increase with task input data size. This is due to the fact that as the amount of input data increases, the task will require more processing and communication resources and all techniques will execute tasks more slowly. In addition, the combined optimization of processing, communication, and storage resources makes the suggested technique perform better than existing solutions.

Figure 6 depicts the effect of the edge server's and the remote cloud's transmission rates on the task execution delay. Figure 6 shows that the total length of all jobs for the proposed method, the RCRA strategy, and the COC strategy steadily lowers as the transmission rate between the edge server and the remote cloud gradually rises. When $R = 5$, the above three strategies performance is the same. This is so that task offloading to the cloud server takes longer than task execution on the edge server when the transmission rate between the edge server and the remote cloud is low. The proposed technique performs very well when compared to RCRA and COC strategies at this moment because the majority of work will be carried out on the edge server. The performance difference between the proposed, RCRA, and COC algorithms will become smaller and smaller as the transmission rate increases because the execution delays of the task in the cloud and the task on the edge server are getting closer and closer. This also demonstrates that the proposed strategy can achieve higher performance gains when the transmission rate is low. In addition, the LOC policy task's execution latency remains unchanged because the task's local delay is determined by the computational capabilities of the terminal and the edge server, not by the transmission rate between clouds.

Figure 7 compares the cache hit ratio of the algorithms under different cache sizes. As can be seen from Figure 7, the cache hit ratio of the proposed algorithm is higher than that of LOC, COC, and RCRA algorithms which makes it superior and effective for the cache utilization of MEC.

Figure 8 compares the throughput of the algorithms under different number of mobile devices. As can be seen from Figure 8, the throughput of the proposed algorithm is higher than existing algorithms which further validates its effectiveness.

To further validate the effectiveness of the proposed algorithm, Figure 9 compares the execution delay of the proposed algorithm and existing convex optimization algorithms (References [6–8]) under transmission rate between the edge server and remote cloud. As can be seen from Figure 9, the execution delay of the proposed algorithm is lower than that of existing algorithms. This further proved the superiority and importance of the proposed algorithm.

Figure 10 compares the execution delay of the proposed and references [80–82] algorithms under increasing storage capacity of the edge server. As can be seen from Figure 10, the execution delay of the proposed algorithm is lower as compared with competing alternatives which validates its effectiveness.

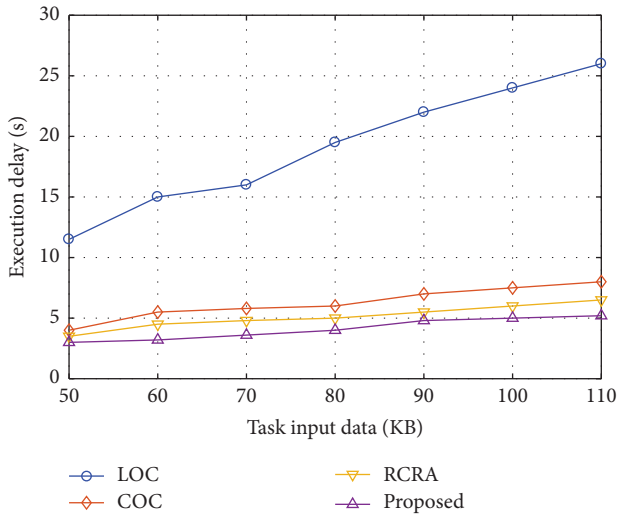

Figure 5: Effect of task input data size on task execution delay.

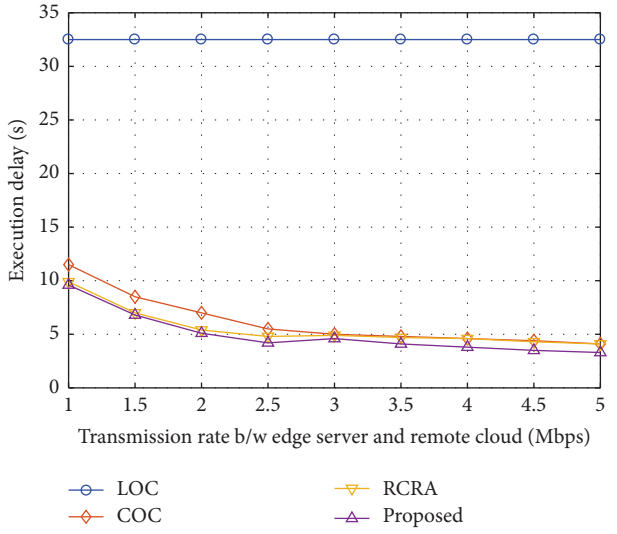

Figure 6: Effect of transmission rate between edge server and remote cloud on task execution delay.

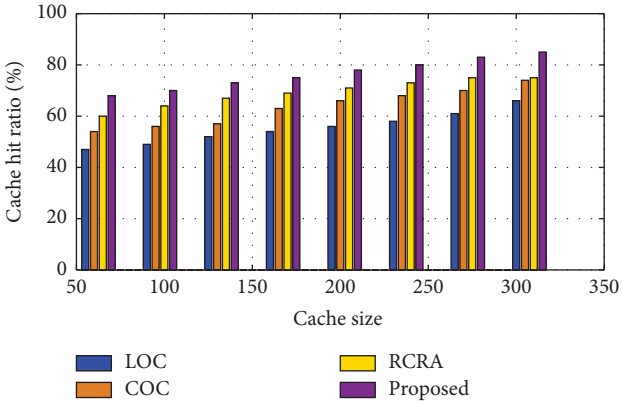

Figure 7: Cache utilization comparison of algorithms.

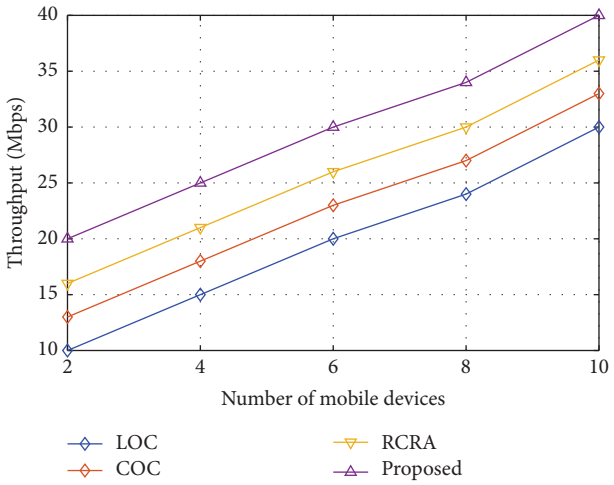

Figure 8: Throughput comparison of algorithms.

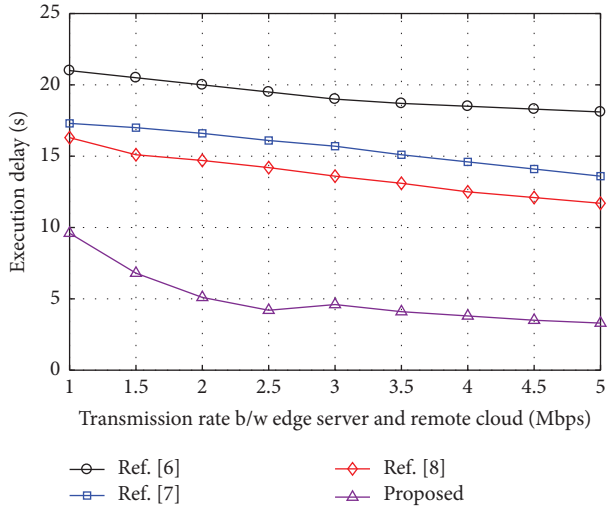

Figure 9: Comparison of execution delays of the proposed and existing convex optimization algorithms.

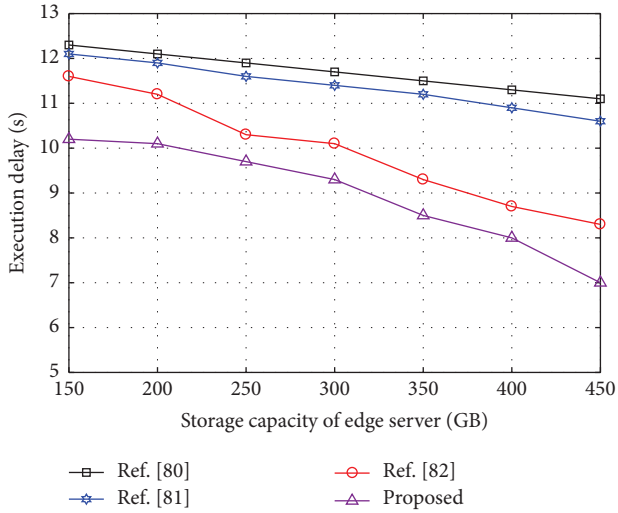

Figure 10: Execution delay vs storage capacity comparison of proposed and existing state-of-the-art algorithms.

7. Conclusion

This study suggests a service cache for the effect of task offloading and service cache coupling on task latency in a centralized mobile edge network made up of multiple terminals, single service nodes, and cloud servers. The combined optimization issue of service caching and resource allocation is first established under the restrictions of computation, communication, and storage capacity. The original problem is then divided into two smaller issues and iteratively improved using the convex optimization approach, relaxation, and reconstructive linearization techniques. According to the simulation results, the proposed method effectively minimizes the system delay and has good convergence when compared to the RCRA strategy, LOC strategy, and COC strategy.

Data Availability

The data used for this study are available within this article.

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Acknowledgments

The authors extend their appreciation to the Deputyship for Research & Innovation, Ministry of Education in Saudi Arabia for funding this research work through the project number: IFP22UQU4170008DSR093.

References

[1] W. Zhou and F. Zhou, "Priority-aware resource scheduling for uav-mounted mobile edge computing networks," *IEEE Transactions on Vehicular Technology*, vol. 72, 2023.

[2] L. Chen, L. Fan, X. Lei, T. Q. Duong, A. Nallanathan, and G. K. Karagiannidis, "Relay-assisted federated edge learning: performance analysis and system optimization," *IEEE Transactions on Communications*, vol. 71, no. 6, pp. 3387–3401, 2023.

[3] W. Zhou and F. Zhou, "Profit maximization for cache-enabled vehicular mobile edge computing networks," *IEEE Transactions on Vehicular Technology*, vol. 72, 2023.

[4] L. He, L. Fan, X. Lei, X. Tang, P. Fan, and A. Nallanathan, "Learning-based MIMO detection with dynamic spatial modulation," *IEEE ransactions on Cognitive Communications and Networking*, vol. 9, no. 6, pp. 1489–1502, 2023.

[5] J. Xia, L. Fan, W. Xu et al., "Secure cache-aided multi-relay networks in the presence of multiple eavesdroppers," *IEEE Transactions on Communications*, vol. 67, no. 11, pp. 7672–7685, 2019.

[6] Y. Fu, X. Yang, P. Yang et al., "Energy-efficient offloading and resource allocation for mobile edge computing enabled mission-critical internet-of-things systems," *European Association for Signal Processing Journal on Wireless Communications and Networking*, vol. 2021, pp. 26–14, 2021.

[7] J. Liu, "Task offloading and resource allocation algorithm based on mobile edge computing in internet of things environment," *Journal of Engineering*, vol. 2021, no. 9, pp. 500–509, 2021.

[8] A. Meng, G. Wei, Y. Zhao, X. Gao, and Z. Yang, "Green resource allocation for mobile edge computing," *Digital Communications and Networks*, vol. 9, no. 5, pp. 1190–1199, 2023.

[9] G. Zhou, R. Zhang, and S. Huang, "Generalized buffering algorithm," *IEEE Access*, vol. 9, pp. 27140–27157, 2021.

[10] H. Tian, N. Huang, Z. Niu, Y. Qin, J. Pei, and J. Wang, "Mapping winter crops in China with multi-source satellite imagery and phenology-based algorithm," *Remote Sensing*, vol. 11, no. 7, p. 820, 2019.

[11] K. Ma, Z. Li, P. Liu et al., "Reliability-constrained throughput optimization of industrial wireless sensor networks with energy harvesting relay," *IEEE Internet of Things Journal*, vol. 8, no. 17, pp. 13343–13354, 2021.

[12] K. Cao, B. Wang, H. Ding et al., "Achieving reliable and secure communications in wireless-powered NOMA systems," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 2, pp. 1978–1983, 2021.

[13] G. Liu, "Data collection in MI-assisted wireless powered underground sensor networks: directions, recent advances, and challenges," *IEEE Communications Magazine*, vol. 59, no. 4, pp. 132–138, 2021.

[14] F. Guo, W. Zhou, Q. Lu, and C. Zhang, "Path extension similarity link prediction method based on matrix algebra in directed networks," *Computer Communications*, vol. 187, pp. 83–92, 2022.

[15] X. Xie and Y. Sun, "A piecewise probabilistic harmonic power flow approach in unbalanced residential distribution systems," *International Journal of Electrical Power & Energy Systems*, vol. 141, Article ID 108114, 2022.

[16] H. Wu, S. Jin, and W. Yue, "Pricing policy for a dynamic spectrum allocation scheme with batch requests and impatient packets in cognitive radio networks," *Journal of Systems Science and Systems Engineering*, vol. 31, no. 2, pp. 133–149, 2022.

[17] Y. Wang, X. Han, and S. Jin, "MAP based modeling method and performance study of a task offloading scheme with time-correlated traffic and VM repair in MEC systems," *Wireless Networks*, vol. 29, no. 1, pp. 47–68, 2022.

[18] J. Zhang, Y. Tang, H. Wang, and K. Xu, "ASRO-DIO: active subspace random optimization based depth inertial odometry," *IEEE Transactions on Robotics*, vol. 39, no. 2, pp. 1496–1508, 2023.

[19] Q. Ni, J. Guo, W. Wu, and H. Wang, "Influence-based community partition with sandwich method for social networks," *IEEE Transactions on Computational Social Systems*, vol. 10, no. 2, pp. 819–830, 2023.

[20] Y. Jiang and X. Li, "Broadband cancellation method in an adaptive co-site interference cancellation system," *International Journal of Electronics*, vol. 109, no. 5, pp. 854–874, 2022.

[21] D. Li, H. Yu, K. P. Tee et al., "On time-synchronized stability and control," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 52, no. 4, pp. 2450–2463, 2022.

[22] J. Li, Y. Deng, W. Sun et al., "Resource orchestration of cloud-edge–based smart grid fault detection," *Association for Computing Machinery Transactions on Sensor Networks*, vol. 18, no. 3, pp. 1–26, 2022.

[23] Y. Mao, Y. Zhu, Z. Tang, and Z. Chen, "A novel airspace planning algorithm for cooperative target localization," *Electronics*, vol. 11, no. 18, p. 2950, 2022.

[24] G. Liu, "A Q-Learning-based distributed routing protocol for frequency-switchable magnetic induction-based wireless

underground sensor networks," *Future Generation Computer Systems*, vol. 139, pp. 253–266, 2023.

[25] T. Li, Y. Li, M. A. Hoque et al., "To what extent we repeat ourselves? Discovering daily activity patterns across mobile app usage," *IEEE Transactions on Mobile Computing*, vol. 21, no. 4, pp. 1492–1507, 2022.

[26] C. Min, Y. Pan, W. Dai et al., "Trajectory optimization of an electric vehicle with minimum energy consumption using inverse dynamics model and servo constraints," *Mechanism and Machine Theory*, vol. 181, Article ID 105185, 2023.

[27] X. Zhang, W. Pan, R. Scattolini, S. Yu, and X. Xu, "Robust tube-based model predictive control with Koopman operators," *Automatica*, vol. 137, Article ID 110114, 2022.

[28] X. Zhang, S. Wen, L. Yan, J. Feng, and Y. Xia, "A hybrid-convolution spatial–temporal recurrent network for traffic flow prediction," *The Computer Journal*, vol. 171, p. c171, 2022.

[29] X. Li and Y. Sun, "Stock intelligent investment strategy based on support vector machine parameter optimization algorithm," *Neural Computing and Applications*, vol. 32, no. 6, pp. 1765–1775, 2020.

[30] X. Li and Y. Sun, "Application of RBF neural network optimal segmentation algorithm in credit rating," *Neural Computing and Applications*, vol. 33, no. 14, pp. 8227–8235, 2021.

[31] Q. Li, H. Lin, X. Tan, and S. Du, "H∞ consensus for multiagent-based supply chain systems under switching topology and uncertain demands," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 50, no. 12, pp. 4905–4918, 2020.

[32] F. Huang, Z. Wang, X. Huang, Y. Qian, Z. Li, and H. Chen, "Aligning distillation for cold-start item recommendation," *Paper presented at the SIGIR '23*, USAfrom, New York, NY, USA, 2023.

[33] X. Liu, T. Shi, G. Zhou et al., "Emotion classification for short texts: an improved multi-label method," *Humanities and Social Sciences Communications*, vol. 10, no. 1, p. 306, 2023.

[34] Y. Yao, J. Zhao, Z. Li, X. Cheng, and L. Wu, "Jamming and eavesdropping defense scheme based on deep reinforcement learning in autonomous vehicle networks," *IEEE Transactions on Information Forensics and Security*, vol. 18, pp. 1211–1224, 2023.

[35] W. Fan, L. Yang, and N. Bouguila, "Unsupervised grouped axial data modeling via hierarchical bayesian nonparametric models with watson distributions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 12, pp. 9654–9668, 2022.

[36] Y. Yao, F. Shu, Z. Li, X. Cheng, and L. Wu, "Secure transmission scheme based on joint radar and communication in mobile vehicular networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 9, pp. 10027–10037, 2023.

[37] Y. Zheng, X. Lv, L. Qian, and X. Liu, "An optimal BP neural network track prediction method based on a GA–ACO hybrid algorithm," *Journal of Marine Science and Engineering*, vol. 10, no. 10, p. 1399, 2022.

[38] Y. Zheng, P. Liu, L. Qian et al., "Recognition and depth estimation of ships based on binocular stereo vision," *Journal of Marine Science and Engineering*, vol. 10, no. 8, p. 1153, 2022.

[39] L. Qian, Y. Zheng, L. Li, Y. Ma, C. Zhou, and D. Zhang, "A new method of inland water ship trajectory prediction based on long short-term memory network optimized by genetic algorithm," *Applied Sciences*, vol. 12, no. 8, p. 4073, 2022.

[40] Y. Zhu, R. Huang, Z. Wu, S. Song, L. Cheng, and R. Zhu, "Deep learning-based predictive identification of neural stem cell differentiation," *Nature Communications*, vol. 12, no. 1, p. 2614, 2021.

[41] G. Zhou, X. Zhou, J. Chen, G. Jia, and Q. Zhu, "LiDAR echo Gaussian decomposition algorithm for FPGA implementation," *Sensors*, vol. 22, no. 12, p. 4628, 2022.

[42] H. Wang, W. Sun, D. Jiang, and R. Qu, "A MTPA and flux-weakening curve identification method based on physics-informed network without calibration," *IEEE Transactions on Power Electronics*, vol. 38, no. 10, pp. 12370–12375, 2023.

[43] B. Cheng, D. Zhu, S. Zhao, and J. Chen, "Situation-aware IoT service coordination using the event-driven SOA paradigm," *IEEE Transactions on Network and Service Management*, vol. 13, no. 2, pp. 349–361, 2016.

[44] X. Liu, J. He, M. Liu, Z. Yin, L. Yin, and W. Zheng, "A scenario-generic neural machine translation data augmentation method," *Electronics*, vol. 12, no. 10, p. 2320, 2023.

[45] S. Lu, M. Liu, L. Yin et al., "The multi-modal fusion in visual question answering: a review of attention mechanisms," *PeerJ Computer Science*, vol. 9, Article ID e1400, 2023.

[46] B. Wang, Y. Shen, N. Li, Y. Zhang, and Z. Gao, "An adaptive sliding mode fault-tolerant control of a quadrotor unmanned aerial vehicle with actuator faults and model uncertainties," *International Journal of Robust and Nonlinear Control*, vol. 33, no. 17, pp. 10182–10198, 2023.

[47] B. Wang, D. Zhu, L. Han, H. Gao, Z. Gao, and Y. Zhang, "Adaptive Fault-tolerant control of a hybrid canard rotor/wing UAV under transition flight subject to actuator faults and model uncertainties," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 59, no. 4, pp. 4559–4574, 2023.

[48] X. Zhang, Y. Wang, X. Yuan, Y. Shen, Z. Lu, and Z. Wang, "Adaptive dynamic surface control with disturbance observers for battery/supercapacitor-based hybrid energy sources in electric vehicles," *IEEE Transactions on Transportation Electrification*, vol. 9, p. 1, 2022.

[49] S. Yang, Q. Li, W. Li, X. Li, and A. Liu, "Dual-level representation enhancement on characteristic and context for image-text retrieval," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 11, pp. 8037–8050, 2022.

[50] K. Zhao, Z. Jia, F. Jia, and H. Shao, "Multi-scale integrated deep self-attention network for predicting remaining useful life of aero-engine," *Engineering Applications of Artificial Intelligence*, vol. 120, Article ID 105860, 2023.

[51] Y. Song, R. Xin, P. Chen, R. Zhang, J. Chen, and Z. Zhao, "Identifying performance anomalies in fluctuating cloud environments: a robust correlative-GNN-based explainable approach," *Future Generation Computer Systems*, vol. 145, pp. 77–86, 2023.

[52] X. Zhou and L. Zhang, "SA-FPN: an effective feature pyramid network for crowded human detection," *Applied Intelligence*, vol. 52, no. 11, pp. 12556–12568, 2022.

[53] X. Dai, Z. Xiao, H. Jiang et al., "Task Co-offloading for d2d-assisted mobile edge computing in industrial internet of things," *IEEE Transactions on Industrial Informatics*, vol. 19, no. 1, pp. 480–490, 2023.

[54] H. Jiang, X. Dai, Z. Xiao, and A. K. Iyengar, "Joint task offloading and resource allocation for energy-constrained mobile edge computing," *IEEE Transactions on Mobile Computing*, vol. 22, no. 7, pp. 4000–4015, 2023.

[55] X. Dai, Z. Xiao, H. Jiang, and J. C. S. Lui, "UAV-assisted task offloading in vehicular edge computing networks," *IEEE Transactions on Mobile Computing*, vol. 23, pp. 1–15, 2023.

[56] Y. Duan, Y. Zhao, and J. Hu, "An initialization-free distributed algorithm for dynamic economic dispatch problems in microgrid: modeling, optimization and analysis,"

Sustainable Energy, Grids and Networks, vol. 34, Article ID 101004, 2023.

[57] S. Wang, H. Sheng, Y. Zhang, D. Yang, J. Shen, and R. Chen, "Blockchain-empowered distributed multi-camera multi-target tracking in edge computing," *IEEE Transactions on Industrial Informatics*, vol. 19, pp. 1–10, 2023.

[58] X. Liang, Z. Huang, S. Yang, and L. Qiu, "Device-Free motion and trajectory detection via RFID," *Association for Computing Machinery Transactions on Embedded Computing Systems*, vol. 17, no. 4, pp. 1–27, 2018.

[59] Y. Xie, X. Wang, Z. Shen, Y. Sheng, and G. Wu, "A two-stage estimation of distribution algorithm with heuristics for energy-aware cloud workflow scheduling," *IEEE Transactions on Services Computing*, vol. 16, pp. 1–14, 2023.

[60] C. Liu, T. Wu, Z. Li, T. Ma, and J. Huang, "Robust online tensor completion for IoT streaming data recovery," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 12, pp. 10178–10192, 2023.

[61] B. Cao, S. Fan, J. Zhao et al., "Large-scale many-objective deployment optimization of edge servers," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 6, pp. 3841–3849, 2021.

[62] B. Cao, J. Zhao, Z. Lv, and P. Yang, "Diversified personalized recommendation optimization based on mobile data," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 4, pp. 2133–2139, 2021.

[63] B. Cao, M. Li, X. Liu, J. Zhao, W. Cao, and Z. Lv, "Many-objective deployment optimization for a drone-assisted camera network," *IEEE transactions on network science and engineering*, vol. 8, no. 4, pp. 2756–2764, 2021.

[64] B. Cao, W. Zhang, X. Wang, J. Zhao, Y. Gu, and Y. Zhang, "A memetic algorithm based on two_Arch2 for multi-depot heterogeneous-vehicle capacitated arc routing problem," *Swarm and Evolutionary Computation*, vol. 63, Article ID 100864, 2021.

[65] B. Cao, Z. Sun, J. Zhang, and Y. Gu, "Resource allocation in 5G IoV architecture based on SDN and fog-cloud computing," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 6, pp. 3832–3840, 2021.

[66] B. Cao, J. Zhang, X. Liu et al., "Edge–cloud resource scheduling in space–air–ground-integrated networks for internet of vehicles," *IEEE Internet of Things Journal*, vol. 9, no. 8, pp. 5765–5772, 2022.

[67] B. Cao, Y. Gu, Z. Lv, S. Yang, J. Zhao, and Y. Li, "RFID reader anticollision based on distributed parallel particle swarm optimization," *IEEE Internet of Things Journal*, vol. 8, no. 5, pp. 3099–3107, 2021.

[68] Q. Zhong, S. Han, K. Shi, S. Zhong, and O. Kwon, "Co-design of adaptive memory event-triggered mechanism and aperiodic intermittent controller for nonlinear networked control systems," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 69, no. 12, pp. 4979–4983, 2022.

[69] Z. Wang, D. Zhao, and Y. Guan, "Flexible-constrained time-variant hybrid reliability-based design optimization," *Structural and Multidisciplinary Optimization*, vol. 66, no. 4, p. 89, 2023.

[70] B. Cao, J. Zhao, Y. Gu, Y. Ling, and X. Ma, "Applying graph-based differential grouping for multiobjective large-scale optimization," *Swarm and Evolutionary Computation*, vol. 53, Article ID 100626, 2020.

[71] B. Cao, J. Zhao, P. Yang et al., "Multiobjective 3-D topology optimization of next-generation wireless data center network," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 5, pp. 3597–3605, 2020.

[72] Z. Xuemin, R. Ying, X. Zenggang, D. Haitao, X. Fang, and L. Yuan, "Resource-constrained and socially selfish-based incentive algorithm for socially aware networks," *Journal of Signal Processing Systems*, vol. 19, 2023.

[73] Q. Wu, J. Fang, J. Zeng, J. Wen, and F. Luo, "Monte Carlo simulation-based robust workflow scheduling for spot instances in cloud environments," *Tsinghua Science and Technology*, vol. 29, no. 1, pp. 112–126, 2024.

[74] X. Ma, Z. Dong, W. Quan, Y. Dong, and Y. Tan, "Real-time assessment of asphalt pavement moduli and traffic loads using monitoring data from Built-in Sensors: optimal sensor placement and identification algorithm," *Mechanical Systems and Signal Processing*, vol. 187, Article ID 109930, 2023.

[75] F. Zhao, H. Wu, S. Zhu et al., "Material stock analysis of urban road from nighttime light data based on a bottom-up approach," *Environmental Research*, vol. 228, Article ID 115902, 2023.

[76] A. Nemirovski, "Interior point polynomial time methods in convex programming," 1996, https://www2.isye.gatech.edu/nemirovs/Lect_IPM.pdf.

[77] X. Gao, D. Niyato, K. Yang, and J. An, "Cooperative scheme for backscatter-aided passive relay communications in wireless-powered D2D networks," *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 152–164, 2022.

[78] X. Gao, P. Wang, D. Niyato, K. Yang, and J. An, "Auction-based time scheduling for backscatter-aided RF-powered cognitive radio networks," *IEEE Transactions on Wireless Communications*, vol. 18, no. 3, pp. 1684–1697, 2019.

[79] S. Han, D. Ma, C. Kang, W. Huang, C. Lin, and C. Tian, "Optimization of mobile edge computing offloading model for distributed wireless sensor devices," *Journal of Sensors*, vol. 2022, Article ID 9047737, pp. 1–9, 2022.

[80] J. Shuja, A. Gani, K. Ko et al., "SIMDOM: a framework for SIMD instruction translation and offloading in heterogeneous mobile architectures," *Transactions on Emerging Telecommunications Technologies*, vol. 29, no. 4, p. e3174, 2018.

[81] S. K. U. Zaman, A. I. Jehangiri, T. Maqsood et al., "LiMPO: lightweight mobility prediction and offloading framework using machine learning for mobile edge computing," *Cluster Computing*, vol. 26, no. 1, pp. 99–117, 2023.

[82] E. Mustafa, J. Shuja, K. Bilal et al., "Reinforcement learning for intelligent online computation offloading in wireless powered edge networks," *Cluster Computing*, vol. 26, no. 2, pp. 1053–1062, 2023.