electronics

MDPI

Article

Joint Optimization of Computation, Communication and Caching in D2D-Assisted Caching-Enhanced MEC System

Jiaqi Ge [1], Gaochao Xu [1,*], Yang Zhang [2,3,*], Jianchao Lu [3], Haihua Chen [4] and Xiangyu Meng [1]

[1] College of Computer Science and Technology, Jilin University, Changchun 130012, China; jiaqi.ge@mq.edu.au (J.G.); xiangyumeng@jlu.edu.cn (X.M.)
[2] School of Information Management, Wuhan University, Wuhan 430072, China
[3] School of Computing, Macquarie University, Sydney, NSW 2109, Australia; jianchao.lu@hdr.mq.edu.au
[4] Department of Information Science, University of North Texas, Denton, TX 76205, USA; haihua.chen@unt.edu
* Correspondence: xugc@jlu.edu.cn (G.X.); yang.zhang@mq.edu.au (Y.Z.)

Abstract: In the era of intelligent applications, Mobile Edge Computing (MEC) is emerging as a promising technology that provides abundant resources for mobile devices. However, establishing a direct connection to the MEC server is not always feasible for certain devices. This paper introduces a novel Device-to-Device (D2D)-assisted system to address this challenge. The system leverages idle helper devices to execute and offload tasks to the MEC server, thereby enhancing resource utilization and reducing offload time. To further minimize offloading time for latency-sensitive tasks, this paper incorporates edge caching. The problem is formulated by jointly optimizing computation, communication and caching, and a novel *J*oint *M*ultiple *D*ecision *O*ptimization *A*lgorithm (JMDOA) is proposed to solve the minimum-energy-consumption problem. Specifically, the JMDOA algorithm decomposes the integer-mixed non-convex optimization problem into two subproblems based on distinct properties of discrete variables. These subproblems are solved separately and optimized iteratively, ensuring convergence to a suboptimal solution. Simulations demonstrate the effectiveness and superiority of JMDOA, exhibiting lower energy consumption and reduced time compared to other baseline algorithms, approaching the optimum. This work contributes to the field by presenting a novel approach to optimizing resource allocation in MEC systems, with potential implications for the future development of intelligent applications.

Keywords: D2D-assisted; caching-enhanced; MEC system; joint optimization; block coordinate descent

Citation: Ge, J.; Xu, G.; Zhang, Y.; Lu, J.; Chen, H.; Meng, X. Joint Optimization of Computation, Communication and Caching in D2D-Assisted Caching-Enhanced MEC System. *Electronics* **2023**, *12*, 3249. https://doi.org/10.3390/electronics12153249

Academic Editor: Flavio Canavero

Received: 16 June 2023
Revised: 11 July 2023
Accepted: 26 July 2023
Published: 27 July 2023

1. Introduction

The proliferation of mobile terminal equipment and the increasing complexity of the Internet-of-Things (IoT)-device applications have highlighted the limitations of mobile cloud computing [1,2], such as long transmission delays and heavy central server loads. In response, Mobile Edge Computing (MEC) [3,4] has emerged as a promising solution, providing abundant computing, communication and storage resources for mobile devices in close proximity. This technology facilitates task offloading, reduces latency and enables agile mobile services [5].

With the advent of 5th generation (5G) [6] wireless systems and the growing complexity of application programs, there has been a surge in large-scale intelligent devices connecting to edge servers [6]. However, some devices face challenges in establishing a direct connection to the MEC server due to limited network coverage, intermittent connectivity or hardware constraints. Additionally, the execution of latency-sensitive tasks requires faster processing times. Therefore, it is crucial to find efficient ways for these devices to execute tasks with minimal overhead and high speed to drive further advancements in mobile edge computing.

Device-to-Device (D2D) communication has been proposed as a solution to reduce delay and energy consumption through cooperative behavior among mobile devices [7–9]. Integrating D2D with mobile edge computing brings together the advantages of localized computing, reduced latency, improved efficiency and enhanced reliability, offering a promising solution for various mobile applications and services. For example, Pan et al. [10] propose D2D communication integrated with content caching to maximize the offloading gain. Similarly, Yu et al. [11] propose leveraging network-assisted D2D collaboration for wireless distributed computing and result sharing. However, there are still some limitations to address. Firstly, appropriate resource management strategies need to be formulated based on the characteristics of D2D. Secondly, the execution time for latency-sensitive tasks needs to be further reduced.

To tackle these challenges, we propose a D2D-assisted and caching-enhanced MEC system, as depicted in Figure 1, aiming to improve resource utilization and shorten execution time for the offloading of computing-intensive tasks in an energy-efficient manner. The main contributions of this paper can be summarized as follows:

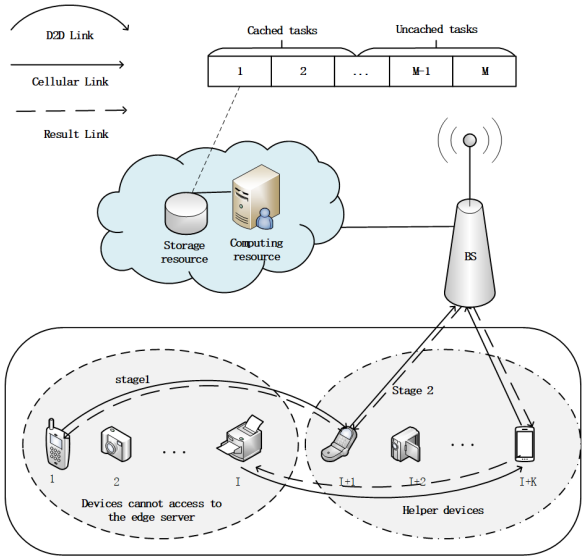

Figure 1. The D2D-assisted and Caching-enhanced System.

- Unlike traditional approaches that directly offload tasks to the MEC server, our proposed D2D-assisted and caching-enhanced MEC system aims to minimize energy consumption with lower latency. By combining edge caching, we upload only uncached tasks from local devices, while executing cached tasks directly on the edge server. This approach allows us to jointly optimize computation, communication and caching, thereby improving resource utilization in the system.
- Additionally, the problem formulated in this paper poses a significant challenge as it involves a mixed-integer non-convex-optimization NP-hard problem with both discrete and continuous variables. Solving this problem directly is difficult. To overcome this, we introduce the Joint Multiple Decision Optimization Algorithm (JMDOA), which utilizes block coordinate descent and convex optimization techniques. Furthermore, we propose a caching-policy-initialization method based on linear weighting, reducing the number of iterations and overall execution time.
- The evaluation of JMDOA demonstrates its effectiveness in optimizing resource allocation and reducing energy consumption compared to other algorithms. The optimal solution achieved via JMDOA closely approximates the exhaustive method, while significantly reducing time complexity. Moreover, our results show that JMDOA successfully reduces execution time to a certain extent.

The rest of this paper is organized as follows: Section 2 introduces the related works. Section 3 proposes a D2D-assisted caching-enhanced MEC system and formalizes the

problem of energy-consumption minimization. In Section 4, the integer-mixed non-convex-optimization NP-hard problem is divided into two parts aiming at energy-consumption minimization and solution, respectively. Section 5 is the experimental simulation part and the conclusion is given in Section 6.

2. Related Work

As for the MEC system, the combination of D2D and MEC enhances the computing ability of the system and shortens transmission distance where the nearby devices provide close support for simple tasks, while computing-intensive tasks can further obtain sufficient computing and storage resources at the cost of long-distance transmission in the MEC server. There are many types of research on the D2D-assisted MEC system. Ouamri et al. [12] primarily focus on the application of Device-to-Device (D2D) communication in the context of Unmanned Aerial Vehicles (UAVs), aiming to improve energy efficiency. Lingjun Pu et al. [13] put forward a new mobile task-unloading framework based on D2D cooperation, which realizes the dynamic sharing of computing and communication resources through the control assistance of network operators. Dan Wu et al. [14] put forward a dynamic distributed-resource-sharing scheme, which is applied to the unified framework of general D2D communication by jointly optimizing mode selection, resource allocation and power allocation. Yinghui He et al. [15] integrate D2D communications with MEC to improve the computation capacity of the cellular networks, aiming to maximize the number of devices supported by the cellular networks with the constraints of both communication and computation resources. Y Dai et al. [16] develop the interlay mode as a unique D2D model to maximize the system sum rate in the non-orthogonal multiple access (NOMA) cellular networks. However, tasks are offloaded to the MEC server or devices nearby as above. Almost none of them consider the circumstance that mobile devices cannot connect to the MEC server directly in the case of no Internet access or poor signal. Therefore, in our D2D-assisted MEC system, mobile devices choose one of the helper devices as a relay to help execute and transmit computation-intensive tasks.

Although the D2D-assisted MEC system solves the problems of limited computing resources and power of mobile devices by reasonable offloading for large-scale and complex tasks, shortening the task-offloading process and reducing the offloading-task data are effective ways to reduce energy consumption further. Therefore, researchers have begun to combine edge caching [17,18] with the MEC system. Reference [19] proposes a D2D-cache strategy using multi-agent reinforcement learning. The D2D-cache problem is formalized as a bandit problem with multiple agents and weapons and Q-learning is used to learn how to coordinate cache decisions. Reference [20] investigates a joint pushing and caching policy in a general mobile-edge-computing (MEC) network with multi-user and multi-cast data based on hierarchical-reinforcement learning. Reference [21] investigates the collaborative-caching problem in the edge-computing environment to minimize the system cost and proposes an online algorithm called CEDC-O to solve this problem. Reference [22] studies edge caching in fog-computing networks, where a capacity-aware edge-caching framework is proposed by considering both the limited fog-cache capacity and the connectivity capacity of base stations. Reference [23] studies the unique problem of caching fairness in edge-computing environments. Reference [24] formulates the Edge Data Caching (EDC) problem as a constrained-optimization problem to minimize the caching-data cost and maximize the reduction in service latency.

Regarding edge caching, most researchers pay more attention to content caching [25,26]. Yu, G et al. [27] propose a content-caching strategy based on mobility prediction and joint-user prefetching. J. Guo et al. [28] propose a novel context-aware object-detection method based on edge-cloud cooperation. Y. Dai et al. [29] integrate DRL and permit blockchain into vehicular networks for intelligent and secure content caching. Wu W et al. [30] propose a content-based D2D cooperative edge-caching strategy. The content can be cached in the user equipment or the surrounding small base stations according to popularity. However, except for content, the task's size, popularity and complexity are also important factors

that deserve consideration. Thus, unlike previous research, we propose a caching-strategy-initialization method based on a linear weighting of complexity, size and popularity.

In addition, the complexity of tasks further accelerates the energy consumption of mobile devices, causing battery consumption to be a critical factor in restricting the development of mobile devices. Meanwhile, executing tasks with low power will prolong the execution time, which cannot meet the quality of service requirements of users. To this end, it is necessary to manage resources reasonably to reduce the energy consumption of mobile devices. Reference [31] focuses on resource allocation in a downlink 5G tri-sectorial cell for non-orthogonal multiple access (NOMA) systems. Reference [32] proposes a proportional fair-based scheduler algorithm that incorporates Signal Interference Noise Ratio (SINR) compensation to address the challenge of resource allocation in cellular networks. Reference [33] studies resource allocation for a multi-user mobile-edge-computation-offloading system under infinite or finite cloud-computation capacity. Reference [34] investigates the latency-minimization problem in a multi-user time-division multiple access mobile-edge-computation-offloading system with joint communication- and computation-resource allocation. Reference [35] studies the problem of joint task offloading and resource allocation to maximize the users' task-offloading gains. Reference [36] proposes an integrated framework for computation offloading and interference management in wireless cellular networks with MEC, which formulates the computation-offloading decision, physical-resource-block (PRB) allocation. Reference [37] designs an iterative heuristic MEC resource-allocation algorithm to make the offloading decision dynamically. Reference [38] optimizes the joint caching, offloading and time-allocation policy to minimize the weighted-sum energy consumption subject to the caching and deadline constraints. Resource management varies in the MEC system, but few of them jointly optimize computing, communication and caching of the offloading process, which is an all-sided optimization.

According to the research above, we propose a D2D-assisted and caching-enhanced MEC system with no Internet access or poor signal for mobile devices to minimize energy consumption. With the help of D2D cooperation, devices that cannot access the Internet choose one of the helper devices as a relay to execute and transmit tasks to the MEC server. Combined with edge caching and jointly optimized resource allocation, the model can improve resource utilization, shorten execution time and reduce energy consumption. To enhance readability, we present a comparative analysis as Table 1.

Table 1. Comparison of the current work with other literature.

Reference	Main Contributions	Limitations
[12–16]	These studies propose various methods for efficient resource allocation and task offloading, integrating D2D communication with mobile edge computing.	These works do not incorporate caching strategies, which could further reduce latency.
[17–24]	These papers present different strategies for caching in mobile D2D networks.	These studies primarily focus on edge caching and do not sufficiently consider the interplay between computation and communication.
[25–30]	These papers primarily focus on content-caching strategies.	These works do not adequately consider the size, popularity and complexity of tasks, which could impact caching efficiency.
[31–38]	These papers propose different optimization strategies for computation, communication and caching.	While these studies consider multiple aspects of the offloading process, few of them offer a comprehensive optimization approach that jointly considers computing, communication and caching.
Our work	We propose a comprehensive optimization approach that jointly considers computing, communication and caching in the offloading process.	—

3. System Model and Problem Formalization

As shown in Figure 1, our study focuses on a D2D-assisted and caching-enhanced MEC system consisting of a single base station and N mobile devices equipped with multiple antennas. In this system, certain local mobile devices are unable to access the MEC server directly (ND) due to lack of Internet access or poor signal. The set of NDs is denoted by $\mathcal{I} = \{1, 2, ..., I\}$. On the other hand, there are idle devices, referred to as helper devices (HDs), denoted by $\mathcal{K} = \{I+1, I+2, ..., I+K\}$. HDs have access to the MEC server, which helps execute and transmit tasks. $\mathcal{I} + \mathcal{K} = N$, where N is the total number of devices in the system.

The base station covering these N devices has deployed an edge server supporting multi-user caching. This edge server provides mobile users with computing and communication resources and caching services. Due to the limited storage space of the MEC server, tasks are cached selectively and only uncached tasks require offloading. Cached tasks can be executed directly at the MEC server without requesting and configuring data. Furthermore, our study considers that the time is divided into multiple time slots with duration T. Each ND is supposed to execute a computation-intensive and latency-sensitive task at the beginning of each slot.

3.1. Task Model

Suppose there is a set of computation-intensive and latency-sensitive tasks, which is described as $\mathcal{M} = \{1, 2, \ldots, M\}$. Each task consists of two elements, $\mathcal{M} = (V_m, C_m)$, where V_m represents the data size of task m (kilobytes) and C_m represents the number of cycles for executing one bit. In addition, each task must be completed within the time slot T. It is worth mentioning that we only consider the delay of sending data while ignoring the transmission delay of data results since the size of the returned result is considerably smaller compared to the entire task data.

At the beginning of each time slot, each ND requests a task from $\mathcal{M}$ randomly, which is requested by multiple NDs concurrently. Additionally, each ND has a unique preference for tasks and we have access to task-popularity information. Suppose the probability of ND i executes the task $M_i; M_i \in \mathcal{M}$ is represented as $P_{M_i}(m)$, where $P_{M_i}(m) = Pr[M_i = m], m \in \mathcal{M}$. All tasks are independent, each ND only executes one task in each time slot, then $\sum_{m \in \mathcal{M}} P_{M_i}(m) = 1$. Suppose the task set of all NDs composes a new state in each time slot, which is denoted by $S = \{M_1, M_2, \ldots, M_I\}$. In this system, we consider B time slot, where $b = \{1, 2, \ldots, B\}$ represents the index of the time slot. In the b-th time slot, we denote the task set as S_b and the set of nodes executing task m as $N_m(b)$. The number of nodes requesting task m is denoted as $\mid N_m(b) \mid$. The offloading process is illustrated in Figure 2 and the variables and symbols used in the model are summarized in Table 2.

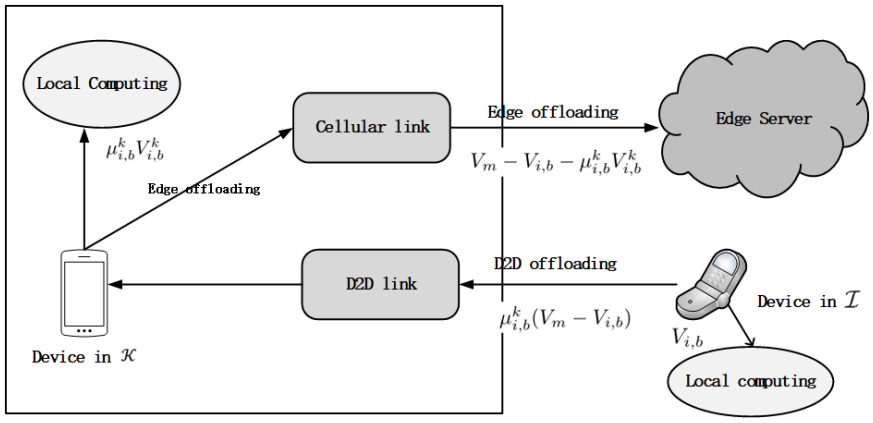

Figure 2. Process of Task Offloading.

Table 2. Units for Magnetic Properties.

Symbol	Description
$\mathcal{I}$	Device set of local devices which cannot access the Internet
$\mathcal{K}$	Device set of helper devices
C_m	The number of cycles for executing one-bit data
$V_{i,b}$	The execution-data size of local device i
$V_{i,b}^k$	The execution-data size of helper device k
$\mu_{i,b}^k$	User association from device i to device k
f_i^l	The CPU computing power of local device i
f_k^l	The CPU computing power of helper device k
p^{wait}	Idle waiting power of the device
$p_{i,b}^k$	Transmission power from device i to k
$p_{k,b}^e$	Transmission power from device k to the edge
a	Cache policy
B_k	Transmission channel bandwidth from device i to k
B_e	Channel bandwidth from device K to edge server
D	Edge-server caching capacity

3.2. Computing Model

Suppose the device $i, i \in \mathcal{I}$, which enables one to directly access the MEC server (ND), requests a computation-intensive and latency-sensitive task. It chooses a helper device (HD) as helper. The computing model includes three parts, the task is executed locally on device i, on the corresponding HD and concurrently on the MEC server. Resource allocation is dynamically redistributed in each time slot. We consider device i executing task m in time slot b, where $m \in \mathcal{M}, i \in N_m(b)$ and $b \in \{1, 2, \ldots, B\}$.

3.2.1. Local Computing Model

Assume that the CPU computing power of the ND i is f_i^l, $V_{i,b}$ denotes the amount of data executed locally and C_m is the number of cycles for executing one-bit data; then, the local computing delay can be expressed as:

$$D_{i,b}^l = \frac{V_{i,b} C_m}{f_i^l}. \tag{1}$$

The local computing energy consumption can be represented as:

$$E_{i,b}^l = \kappa {f_i^l}^2 V_{i,b} C_m, \tag{2}$$

where κ is the effective capacitance related to the CPU core and we default $\kappa = 10^{-26}$ in this paper.

3.2.2. Helper Devices Computing Model

The helper device k helps compute and transmit tasks from ND i, the local computing delay of the helper device k when executing the task offloaded by device i can be expressed as:

$$D_{k,b}^l = \frac{\mu_{i,b}^k V_{i,b}^k C_m}{f_k^l}, \tag{3}$$

where $\mu_{i,b}^k$ represents the user association between ND i and HD k during time slot b. $\mu_{i,b}^k = 1$ indicates the presence of a direct D2D link established from device i to device k; otherwise, there is no link. Once the connection is established, device k provides computing and communication resources for device i. f_k^l is the CPU computing power of helper device k and $V_{i,b}^k$ indicates the amount of data offloaded by the ND i and executed by the helper

device k in the time slot b. The computing energy consumption consumed by helper device k is described as:

$$E^l_{k,b} = \kappa \mu^k_{i,b} {f^l_k}^2 V^k_{i,b} C_m. \tag{4}$$

3.2.3. Edge Computing Model

At the start of each time slot, the MEC server reallocates its resources. Suppose the computing capacity allocated by the edge server to the device k for executing the offloaded task from device i is denoted by $f^e_{i,b}$; then, the edge computing delay is calculated as:

$$D^e_{i,b} = \frac{(V_m - V_{i,b} - \mu^k_{i,b} V^k_{i,b}) C_m}{f^e_{i,b}}. \tag{5}$$

Although the mobile device is idle in the calculation period of the edge server, the energy consumption of the waiting time can be considered proportional to the execution time. Suppose that the waiting power consumed is p^{wait}; then, the power consumption of the device i during the execution of the server can be expressed as:

$$E^e_{i,b} = p^{wait} * D^e_{i,b}. \tag{6}$$

3.3. Communication Model

If ND i requests to perform a task that is not cached in the edge server, it requires assistance from helper device k to upload the task to the MEC server. The task is then executed concurrently locally, on the helper device and the edge server. The transmission of communication includes two stages, transmission from ND i to HD k and transmission from HD k to the edge server.

3.3.1. Transmission Model from NDs to HDs

The ND i selects an HD k as a helper device based on their distance. We assume that a direct D2D link can be established between i and k if they are within a certain threshold range, which we denote as R. Let the transmission power provided by the device i in time slot b be denoted as $p^k_{i,b}$. The communication rate between ND i and the corresponding HD k can be expressed as:

$$r^k_{i,b} = B_k \log_2(1 + \frac{p^k_{i,b} h^k_{i,b}}{\sigma^2}), \tag{7}$$

where σ^2 indicates the noise interference of the channel, B_k is the channel bandwidth and $h^k_{i,b}$ represents the channel gain from the device i to k, which is assumed to be a constant in order to facilitate calculation.

Since the amount of data executed locally is set to $V_{i,b}$, the delay from device i to k can be expressed as:

$$D^k_{i,b} = \frac{\mu^k_{i,b}(V_m - V_{i,b})}{r^k_{i,b}}, \tag{8}$$

then the required energy consumption is:

$$E^k_{i,b} = p^k_{i,b} D^k_{i,b}. \tag{9}$$

Given the limited transmission power of each mobile device, denoted as p^{max}, and the one-to-one user-association between devices, the transmission power within each time slot is subject to the following constraint:

$$p^k_{i,b} \leq p^{max}, \forall i \in \ N_m(b), b \in \{1, 2, \ldots, B\}. \tag{10}$$

3.3.2. Transmission Model from HDs to the MEC Server

In the process of edge transmission, the mobile edge server assigns the same subchannel, denoted as B_e, to each mobile device. Let $p^e_{k,b}$ represent the transmission power allocated from the helper device k to the edge server. $h^e_{k,b}$ denotes the channel gain from mobile device k to the edge server in the b-th time slot. While this value remains constant within each time slot, it varies across different time slots. Therefore, we define the offloading rate from helper device k to the mobile-edge-computing (MEC) server as:

$$r^e_{k,b} = B_e \log_2(1 + \frac{p^e_{k,b} h^e_{k,b}}{\sigma^2}). \tag{11}$$

The offloaded data calculated at the edge server is $V_m - V_{i,b} - \mu^k_{i,b} V^k_{i,b}$, then the transformation delay from mobile device k to the edge server can be expressed as:

$$D^e_{k,b} = \frac{V_m - V_{i,b} - \mu^k_{i,b} V^k_{i,b}}{r^e_{k,b}}. \tag{12}$$

The energy consumption during transmission can be expressed as:

$$E^e_{k,b} = p^e_{k,b} D^e_{k,b}. \tag{13}$$

Due to the resource constraints mobile device k can bring, we set the maximum transmission power as p^{max}, then the constraint of transmission power from the device k to the edge server is restricted as follows:

$$p^e_{k,b} \leq p^{max}, \forall i \in \ N_m(b), b \in \{1, 2, \ldots, B\}. \tag{14}$$

3.4. Cache Model

In our study, we assume that all applications are cached in the edge server, since memory is insufficient for most types of applications. Therefore, the caching strategy primarily focuses on task-data caching. Caching task data offers several advantages, including reduced data transmission, lower latency and decreased energy consumption. When a task is cached, it eliminates the need for data transmission. However, given the limited storage space of the edge server, it becomes crucial to allocate the cache space effectively. This allocation depends on factors such as task popularity, task data size and the complexity of task execution. Considering these factors is vital for optimizing the caching strategy and improving overall system performance.

Let D represent the caching capacity of the edge server. If the task is cached on the server, the returned results are negligible throughout the entire execution process. Therefore, the execution time and energy consumption of the task are equivalent to those of executing the task on the edge server alone, denoted as $D^{cache}_{i,b} = D^e_{i,b}$ and $E^{cache}_{i,b} = E^e_{i,b}$, respectively. Conversely, the task is not currently cached, which requires the collaboration of local computing, helper devices and an edge server for its execution. The offloading process can be divided into two parts. Initially, the task is offloaded to the helper device, where both the local computing of mobile device i and the transmission process to mobile device k are performed simultaneously. Subsequently, the local execution of mobile device k is carried out in parallel with the transmission from device k to the edge server, as depicted in Figure 3. Thus, the time delay depends on the longer execution time among them. The calculation delay can be expressed as:

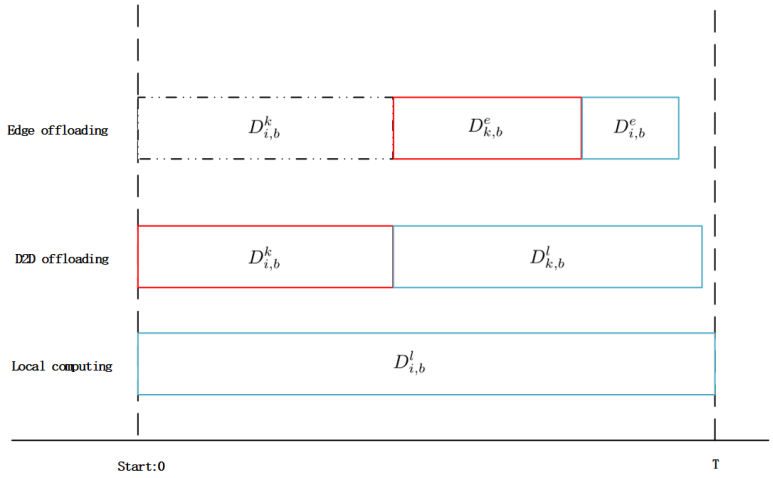

Figure 3. Delay of task offloading.

$$D_{i,b}^{uncache} = \max\{D_{i,b}^{l}, D_{i,b}^{k} + D_{k,b}^{l}, D_{i,b}^{k} + D_{k,b}^{e} + D_{i,b}^{e}\}. \tag{15}$$

The energy consumed by the mobile device during the offloading process without caching can be expressed as follows:

$$E_{i,b}^{uncache} = E_{i,b}^{l} + E_{k,b}^{l} + E_{i,b}^{k} + E_{k,b}^{e} + E_{i,b}^{e}. \tag{16}$$

Let a_m be a binary variable indicating whether task m is cached (a_m = 1) or not (a_m = 0). $a_m E_{i,b}^{cache}$ represents the energy consumed when task m is cached, while the term $(1 - a_m)E_{i,b}^{uncache}$ represents the energy consumed when task m is not cached; then, the total energy consumption of the mobile device i in the b-th slot can be calculated with the following formula:

$$E_{total} = \sum_{i \in N_m(b)} \sum_{m \in \mathcal{M}} a_m E_{i,b}^{cache} + (1 - a_m) E_{i,b}^{uncache}. \tag{17}$$

During the execution of tasks, the interactions between resource allocation and caching strategy play a crucial role. However, caching has a lasting impact on tasks due to its long-term nature, while resource allocation requires real-time updates. To address this, we propose a method that involves utilizing the same caching strategy for consecutive time slots, denoted as B, while updating resource allocation and other variables at the start of each slot. One of the primary constraints affecting task execution is the power consumption of mobile devices. As a result, our objective is to minimize energy consumption by optimizing the transmission power, task partitioning, user association and caching strategy. Let a denote the caching strategy which is a vector of m dimensions. It can be expressed as $a = \{a_1, a_2, \ldots, a_M\}$. V_I and V_K correspond to the execution data of device i and the corresponding helper device k, respectively. Each device has a preference for the requested task. Specifically, V_I is only related to each device and can be represented as a vector of dimension I, denoted as $V_i = V_1, V_2, \ldots, V_I$. On the other hand, V_K is related to device i and is a matrix of dimension $I * K$. p^I represents the transmission power from mobile device I to mobile device K in the b-th time slot. Similarly, p^k represents the transmission power provided by the device k when uploading the task from device I to the edge server. μ represents the user association which is a one-to-one mapping from I to K. All three variables, p^I, p^k and μ, are vector matrices with dimensions $I \times K$. By jointly considering

these factors, we aim to achieve efficient task execution on mobile devices. The problem is formalized as follows:

$$
\begin{aligned}
P_1 : & \min_{a,V_I,V_K,p_I,p_K,\mu} \sum_{b=1}^{B} E_{total} \\
s.t.\ & C1 : \mu_{i,t}^{k} \in \{0,1\} \\
& C2 : \sum_{i \in N_m(b)} \sum_{m \in \mathcal{M}} \mu_{i,t}^{k} \leq 1, \forall k \in \mathcal{K} \\
& C3 : \sum_{k \in \mathcal{K}} \mu_{i,t}^{k} \leq 1, \forall i \in N_m(b), m \in \mathcal{M} \\
& C4 : a_m \in \{0,1\}, \forall m \in \mathcal{M} \\
& C5 : \sum_{m \in \mathcal{M}} a_m V_m \leq D \\
& C6 : a_m D_{i,b}^{cache} + (1 - a_m) D_{i,b}^{uncache} \leq T \\
& C7 : V_m - V_{i,b} - \mu_{i,b}^{k} V_{i,b}^{k} \geq 0, \forall i \in N_m(b), m \in \mathcal{M} \\
& C8 : 0 \leq p_{i,b}^{k}, p_{k,b}^{e} \leq p^{max}, \forall i \in N_m(b), m \in \mathcal{M} \\
& \forall k \in \mathcal{K}, \forall b \in \{1,2,\ldots,B\}.
\end{aligned}
\tag{18}
$$

The objective of this study is to minimize the overall energy consumption of mobile devices, where $a, V_I, V_K, p^I, p^K, \mu$ are optimization variables of the problem. Constraint $C1$ indicates that the user association is represented by a 0-1 integer variable. Constraints $C2$ and $C3$ guarantee a one-to-one mapping between mobile device I and mobile device K for user association. Constraint $C4$ states that the caching strategy is a binary variable. To maintain the capacity limit D, constraint $C5$ ensures that the total amount of task data cached on the server does not exceed this limit. Regardless of caching in the edge server or not, constraint C6 states that the task execution time for all devices must not exceed the time constraint T. Constraint $C7$ denotes the amount of execution data of device i, device k and the edge server are all in the range of 0 to V_m. Constraint $C8$ indicates that the transmission power from mobile device i to device k and the transmission power from mobile device k to the edge server cannot exceed the maximum power p^{max}. Since resources are reallocated at the beginning of each period, all constraints should be met within B periods.

4. Problem Solution

$P1$ is a non-convex NP-hard problem that involves a combination of integer and mixed variables. The problem includes two important variables, caching strategy a and user association μ, both of which can take binary values. Solving this problem is further complicated by the presence of second-order terms in the objective function and constraints, specifically in the form of $x \cdot y$. However, it is worth noting that the caching strategy a remains fixed across consecutive time slots, while user association μ varies in each time slot. This distinction allows us to address these variables separately, simplifying the problem-solving process.

In the case of user association μ, each variable $\mu_{i,b}^{k}$ is a 0-1 integer variable. One approach is to exhaustively enumerate all 2^{I*K} possible solutions. While this method may be feasible for small values of $I * K$, it becomes impractical as $I * K$ increases. Therefore, to facilitate subsequent optimization, we introduce a relaxation technique where we consider the range $0 \leq \mu_{i,b}^{k} \leq 1$ for each variable. By relaxing the discrete nature of $\mu_{i,b}^{k}$, we eliminate constraint $C1$.

Furthermore, to simplify the subsequent solution process, we undertake a variable transformation for $P1$. Given $\delta_{i,b}^k = p_{i,b}^k \frac{\mu_{i,b}^k (V_m - V_{i,b})}{B_k \log_2(1+\frac{p_{i,b}^k h_{i,b}^k}{\sigma^2})}$, then $p_{i,b}^k = \frac{\sigma^2 (2^{\frac{\mu_{i,b}^k (V_m - V_{i,b})}{\delta_{i,b}^k B_k}} - 1)}{h_{i,b}^k}$. Similarly, if we have $\delta_{k,b}^e = \frac{V_m - V_{i,b} - \mu_{i,b}^k V_{i,b}^k}{B_e \log_2(1+\frac{p_{k,b}^e h_{k,b}^e}{\sigma^2})}$, we obtain $p_{k,b}^e = \frac{\sigma^2 (2^{\frac{V_m - V_{i,b} - \mu_{i,b}^k V_{i,b}^k}{\delta_{k,b}^e B_e}} - 1)}{h_{k,b}^e}$. Meanwhile, we can replace constraint $C6$ with $C9$ and constraint $C8$ with $C10$ and $C11$. Consequently, problem $P1$ can be transformed into problem $P2$.

$$
\begin{aligned}
P_2 : & \min_{a, V_I, V_K, \delta_I, \delta_K, \mu} \sum_{b=1}^{B} h(a, V_I, V_K, \delta_I, \delta_K, \mu) \\
& C9 : a_m \frac{V_m C_m}{f_{k,b}^e} + (1 - a_m) \max\{ \frac{V_{i,b} C_m}{f_i^l}, \\
& \delta_{i,b}^k + \frac{\mu_{i,b}^k V_{i,b}^k C_m}{f_k^l}, \delta_{i,b}^k + \delta_{k,b}^e + \frac{(V_m - V_{i,b} - \mu_{i,b}^k V_{i,b}^k) C_m}{f_{k,b}^e} \} \leq T \\
& C10 : 0 \leq \delta_{i,b}^k \leq \frac{\mu_{i,b}^k (V_m - V_{i,b})}{B_k \log_2(1+\frac{p^{max} h_{i,b}^k}{\sigma^2})} \\
& C11 : 0 \leq \delta_{k,t}^e \leq \frac{V_m - V_{i,b} - \mu_{i,b}^k V_{i,b}^k}{B_e \log_2(1+\frac{p^{max} h_{k,b}^e}{\sigma^2})} \\
& C2, C3, C4, C5, C7,
\end{aligned}
\tag{19}
$$

where the objective function h can be represented by:

$$
\begin{aligned}
h = & \sum_{m \in \mathcal{M}} \sum_{i \in N_m(b)} \sum_{k \in \mathcal{K}} a_m p^{wait} \frac{V_m C_m}{f_{k,b}^e} \\
& + (1 - a_m)(\kappa {f_i^l}^2 C_m V_{i,b} + \kappa \mu_{i,b}^k {f_k^l}^2 C_m V_{i,b}^k \\
& + \frac{\sigma^2 (2^{\frac{\mu_{i,b}^k (V_m - V_{i,b})}{\delta_{i,b}^k B_k}} - 1)}{h_{i,b}^k} \delta_{i,b}^k + \frac{\sigma^2 (2^{\frac{V_m - V_{i,b} - \mu_{i,b}^k V_{i,b}^k}{\delta_{k,b}^e B_e}} - 1)}{h_{k,t}^e} \delta_{k,b}^e \\
& + p^{wait} \frac{(V_m - V_{i,b} - \mu_{i,b}^k V_{i,b}^k) C_m}{f_{k,b}^e}).
\end{aligned}
\tag{20}
$$

$P2$ is an integer-mixed non-convex joint optimization problem. To address this problem, we propose a new algorithm Joint Multiple Decision Optimization Algorithm (JMDOA), which is based on iterative block coordinate descent and convex optimization technologies. Specifically, we decompose $P2$ into two subproblems to minimize energy consumption. The first subproblem involves jointly optimizing computing and communication using a predefined caching strategy. The second subproblem focuses on optimizing the caching strategy based on the solution obtained from the first subproblem.

4.1. User Association, Task Partition and Transmission Power Optimization.

In this part, we focus on enhancing computing and communication efficiency by utilizing a specific caching strategy. To achieve this, we aim to optimize the user association, task partitioning and transmission power in a coordinated manner. Firstly, we set the initial

caching policy as $a = a^0$. By doing so, constraint $C9$ can be transformed into $C12$ and we derive the subproblem P_3 by incorporating the initial cache policy into problem P_2:

$$\begin{aligned} P_3 : & \min_{V_I, V_K, \delta_I, \delta_K, \mu} \sum_{b=1}^{B} f(V_I, V_K, \delta_I, \delta_K, \mu) \\ s.t.\ C12 : & a_m^0 \frac{V_m C_m}{f_{k,b}^{edge}} + (1 - a_m^0) \max\{ \frac{V_{i,b} C_m}{f_i^{local}}, \\ & \delta_{i,b}^k + \frac{\mu_{i,b}^k V_{i,b}^k C_m}{f_k^{local}}, \delta_{i,b}^k + \delta_{k,b}^e + \frac{(V_m - V_{i,b} - \mu_{i,b}^k V_{b,t}^k) C_m}{f_{k,b}^{edge}} \} \leq T \\ & C2, C3, C7, C10, C11. \end{aligned} \tag{21}$$

$P3$ is a non-convex problem due to the presence of second-order terms involving the multiplication of variables x and y in both the objective function and multiple constraints. To address this issue, we adopt the Reformulation–Linearization Technique (RLT) [39], which allows us to linearize second-order terms and eliminate all quadratic terms. We present the solution of $V_I, \delta_I, \delta_K, \gamma, \beta$ in Appendix A.

After obtaining the solution $V_I, \delta_I, \delta_K, \gamma, \beta$, the suboptimal solution of problem P_3 is obtained. However, it is important to note that μ represents the solution obtained after relaxation, where the solution vector contains continuous values ranging from 0 to 1. In reality, user association is represented by a binary vector. Therefore, we need a method to convert the relaxed values into binary variables while aiming to approach the optimal solution. Here is the specific procedure for this conversion: if the value of $\mu_{i,b}^k$ is less than 0.3, we set $\mu_{i,b}^k$ to 0. If the value is larger than 0.7, we set $\mu_{i,b}^k$ to 1. Otherwise, we sort the values in descending order. This conversion process helps align the relaxed solution with the optimal solution to the greatest extent possible.

4.2. Optimization of Caching Strategy

With the initial cache policy denoted as $a = a^0$, we obtain the jointly optimized scheme $V_I, V_K, \delta_I, \delta_K, \mu$. By incorporating these results into problem P_2, we can derive a new optimized caching policy. Consequently, the energy consumption can be expressed as:

$$g(a) = \sum_{m \in \mathcal{M}} \sum_{i \in N_m(b)} a_m E_{i,b}^{cache} + (1 - a_m) E_{i,b}^{uncache}. \tag{22}$$

The energy consumption of B time slots can be expressed as:

$$\begin{aligned} P_5 : & \min_{a} \sum_{b=1}^{B} g(a) \\ s.t.\ C17 : & a_m \frac{V_m C_m}{f_{k,b}^e} + (1 - a_m) \\ & \max\{ \frac{V_{i,b}^0 C_m}{f_i^l}, \delta_{i,b}^{k,0} + \frac{\mu_{i,b}^{k,0} V_{i,b}^{k,0} C_m}{f_k^l}, \delta_{i,b}^{k,0} + \delta_{k,b}^{e,0} \\ & + \frac{(V_m - V_{i,b}^0 - \mu_{i,b}^{k,0} V_{i,b}^{k,0}) C_m}{f_{k,b}^e} \} \leq T \\ & C4, C5. \end{aligned} \tag{23}$$

The question P_5 is an integer-programming problem with binary variables. By utilizing the exhaustive method, we can enumerate all possible caching strategies, amounting to a total of 2 to the power of M possibilities. This approach has a time complexity of $O(2^M)$. For small values of M, the time complexity falls within an acceptable range. However, as M gradually increases, the computational cost becomes excessively high, rendering it unsuitable for scenarios involving a large number of tasks. To address this limitation and accommodate general situations, we employ a method based on convex optimization to solve the problem. As a is a discrete variable, P_5 cannot be regarded as a convex

optimization problem; thus, we relax the vector a as $0 \leq a \leq 1$; then, P_5 can be converted into P_6:

$$\begin{aligned} P_6 : & \min_{a} \sum_{t=1}^{P} g^{'}(a) \\ s.t.\ & C18 : 0 \leq a \leq 1 \\ & C5, C17. \end{aligned} \tag{24}$$

It is obvious that P_6 is a convex optimization problem, which can be solved by the conventional convex optimization method.

4.3. Design and Implementation of Global Algorithm

Based on the analysis provided, we present a comprehensive overview of the global algorithm as shown in Algorithm 1. Our primary focus is on problem P_1, which involves a combination of discrete and continuous variables. Both the problem and its constraints feature second-order terms of the form $x \cdot y$. Notably, P_1 is a non-convex NP-hard problem that incorporates integer-mixed elements. To tackle this complexity, we approach the discrete variables μ and a based on their inherent properties. By relaxing the variable μ, we transform P_1 into a new problem denoted as P_2. This conversion allows us to effectively handle the aforementioned challenges and enhance the overall solution. Next, we propose an alternative iterative algorithm to address the optimization problem P_2 in this study. Our algorithm combines block coordinate descent and convex optimization techniques and decomposes P_2 into two subproblems, namely P_3 and P_5. The first subproblem, P_3, focuses on optimizing user association, task partition and transmission power while considering a given caching strategy a. To efficiently solve P_3, we employ the Reformulation–Linearization Technique (RLT) to eliminate all quadratic terms and then solve the resulting problem using the SLSQP method. The second subproblem, P_5, aims to optimize the caching strategy using the variables obtained from the solution of the first subproblem. P_5 is solved directly by relaxing the constraints on a. We iteratively solve P_3 and P_5 until the growth of energy consumption falls below a certain threshold denoted as ε. This iterative process ensures the convergence of the algorithm. Both P_3 and P_5 are convex optimization problems and can be regarded as multi-convex problems, as confirmed in Reference [40]. The overall process is illustrated in Figure 4.

Algorithm 1 JMDOA algorithm for solving global problem

Initialization: Caching strategy $a = a^{(0)}$; Aiming function $f = 0$; Execution times $x = 0$

1: **repeat**
2: solve problem $P4$
3: obtain $V_I^{(x)}, V_K^{(x)}, p_I^{(x)}, p_K^{(x)}, \mu^{(x)}$
4: $f^x \leftarrow$ solve problem P_6 for given $V_I^{(x)}, V_K^{(x)}, p_I^{(x)}, p_K^{(x)}, \mu^{(x)}$
5: compute a^x
6: x=x+1
7: **until** $f^x - f^{x+1} < \varepsilon$

return f,$a, V_I, V_K, p_I, p_K, \mu$

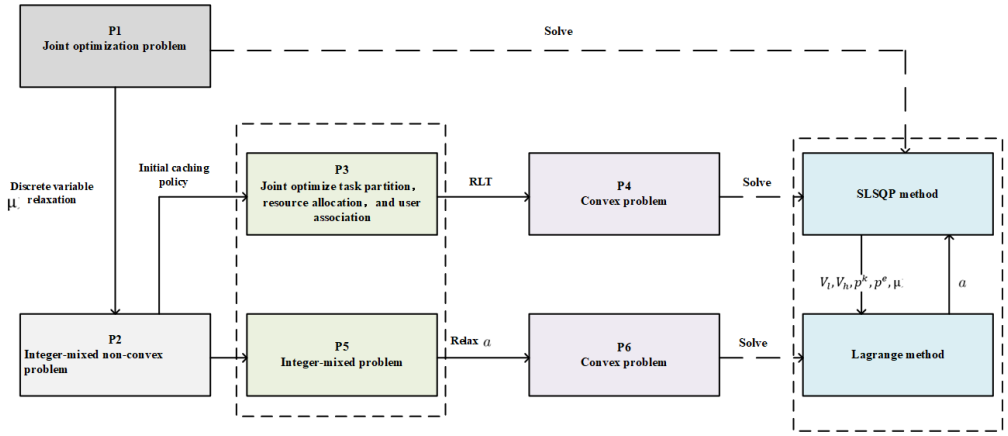

Figure 4. Process of problem transformation and solving.

It is worth noting that the caching strategy obtained via P_6 is represented as a continuous vector, but it actually functions as a binary (0-1) integer variable. Therefore, we need to discretize the continuous variable a_m. The discretization process for the caching policy follows the same approach as variable μ. If the caching strategy value is greater than 0.7, we set a_m to 1; if the value is less than 0.3, a_m is set to 0. For values between 0.3 and 0.7, they are sorted in ascending order and stored based on the available storage space. Our discretization method is determined through experimental testing. Although different schemes have been proposed by researchers for discretizing continuous variables, there is currently no unified approach. Consequently, the precision of our method may not be optimal, but experimental results demonstrate its effective performance and efficiency.

4.4. Initialization of Caching Strategy

The initialization of a caching strategy greatly impacts the time complexity. By employing a suitable initialization method, the number of iterations and time required can be minimized. In this study, we propose a weight-based caching method. To implement this method, we introduce a utility function:

$$f(m) = w_1\lambda(s_m) + w_2\lambda(p_m) + w_3\lambda(c_m), \tag{25}$$

where s_m represents the size of the task m, p_m denotes the popularity and c_m denotes the complexity of the task. w_1,w_2 and w_3 are the weight of three factors above, respectively, which can be obtained via the principal component analysis [41] and $w_1 + w_2 + w_3 = 1$. Since the three factors have varying magnitudes, we normalize them using the max-min normalization method:

$$\lambda(x) = \frac{x - X_{min}}{X_{max} - X_{min}}, \tag{26}$$

where x represents the original value of a current attribute, X represents the set of attributes that require normalization. X_{min} and X_{max} denote the minimum and maximum values within the set of attributes X that require normalization. For instance, we measure task popularity during the total execution times of B slots, which can be represented as $\sum_{b=1}^{B} \mid N_m(b) \mid$. Then, the set of popularities is $X = \{\sum_{b=1}^{B} \mid N_1(b) \mid, \sum_{b=1}^{B} \mid N_2(b) \mid, \ldots, \sum_{b=1}^{B} \mid N_m(b) \mid\}$. By applying a normalization process, we can derive the normalized value $\lambda(p_m)$. Next, after calculating the revenue function for each task, we sort the resulting set $S = f(1), f(2), \ldots f(m)$ in descending order. Subsequently, we cache the first n values based on the available storage space, leading to the determination of the initial caching strategy $a_m^{(0)}$.

5. Simulation Results

This section presents the design of the simulation environment for the D2D-assisted and caching-enhanced MEC system, along with the verification of our proposed JMDOA

algorithm through simulation. Unless specified otherwise, all parameters adhere to the defined values. The system under consideration consists of 16 mobile devices and an edge server, where half of the mobile devices are unable to establish a connection with the edge server. The task list comprises 10 tasks, denoted as $M = 10, N = 16$. The CPU frequency of each device i and device k is randomly distributed as $f_i^l \sim random[0.6, 0.8]$ GHz and $f_k^l \sim random[0.8, 1]$ GHz, respectively. Meanwhile, the CPU frequency of the edge server is set at $f_{k,t}^e = 2$ GHz. The computation task involves input data with a size ranging from $V_m \sim random[800, 1000]$ Mbytes. Executing one-bit data requires a varying number of cycles, denoted as C_m within the range of $[800, 1000]$. The caching space of the edge server is represented by $D = 10^6$ bits and the effective switched capacitance is $\kappa = 10^{-26}$. Regarding the communication model, the bandwidth between local devices and helper devices is assumed to be $B_k = 2$ Mbps, while the bandwidth between helper devices and the edge is set as $B_e = 5$ Mbps. The noise power is defined as $\sigma^2 = 10^{-9}$. Each mobile device has a maximum transmission power of $p^{max} = 1.2$ W and an idle power consumption of $p^{wait} = 0.1$ W. The channel power gain from local devices to helper devices is modelled as $h_{i,t}^k \sim (3, 5) * 10^{-7}$ and $h_{k,t}^e \sim (6, 8) * 10^{-7}$ from helper devices to the edge. To ensure reliable results, we conducted 100 simulations and calculated the average values. The simulations are performed on a standard PC with a CPU speed of $2 * 2.88$ GHz, 8 G of memory, using Python and Pyopt [38] for convex optimization. Additionally, the time slot is set to $T = 10$ s.

5.1. Effect of JMDOA

In this simulation, we verify the effect of our proposed JMDOA compared with the other three comparison schemes from time cost and energy consumption.

Greedy Algorithm (GA): This approach involves updating the caching strategy based on the current optimal value and obtaining results through an iterative solution.

Exhaustive Algorithm (EA): To achieve the optimal solution, we systematically enumerate all possible values of a. Despite the time-intensive nature of this algorithm, it consistently delivers optimal results that serve as valuable references.

Preconditioned Sequential Quadratic Programming (PSQP): This algorithm is a highly efficient algorithm known as the sequential quadratic programming method with a BFGS variable metric update. It is widely recognized for its effectiveness in problem-solving. Our main objective is to apply this algorithm to address problem P_3 in a comparative experiment.

Figures 5 and 6 demonstrate the effectiveness of our proposed JMDOA algorithm in reducing both energy consumption and execution time. In comparison to the efficient PSQP algorithm, JMDOA exhibits superior performance. While PSQP is well suited for convex optimization problems, the problem at hand involves a combination of integer and mixed variables, making it non-convex and NP-hard. JMDOA, designed to handle such complex problems, provides a more efficient solution. Although our algorithm slightly trails the EA in terms of energy consumption, the difference is minimal. However, the EA incurs significantly increased execution time as it systematically enumerates all possible values, which can be extremely time-consuming. In contrast, JMDOA utilizes a more efficient optimization approach, striking a better balance between energy consumption and execution time. Additionally, JMDOA outperforms the GA in initializing the caching strategy, leading to improved performance in both energy conservation and time efficiency. Efficient caching significantly reduces the time and energy required for data retrieval, contributing to overall performance improvement. These results validate the effectiveness of JMDOA in optimizing resource allocation in D2D-assisted mobile-edge-computing systems.

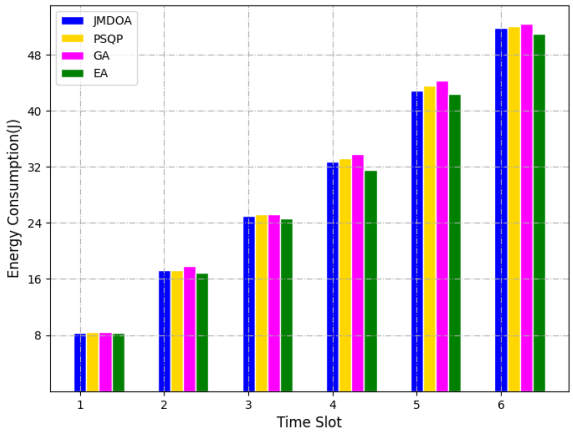

Figure 5. Effect of JMDOA on Energy Consumption.

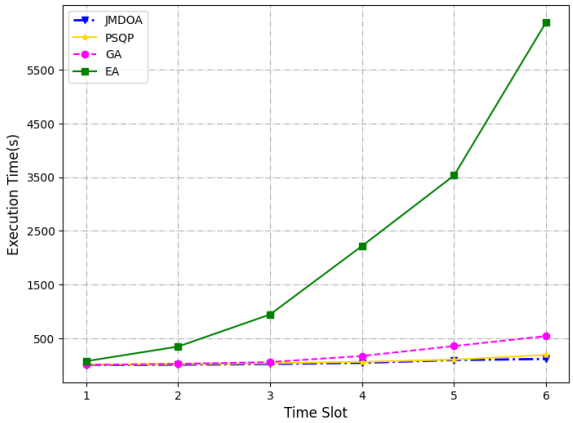

Figure 6. Effect of JMDOA on Excution Time.

5.2. Effect of Joint Resource Allocation

In this simulation, we evaluate the performance of the joint resource-allocation strategy with edge caching, where we give the contrasting tests as follows:

All helper device execution (AH): All tasks not stored in cache are delegated to helper devices for execution.

All edge server execution (AE): All tasks are offloaded to the edge server by the helper devices and computing resources are average allocated to each task.

Joint execution (LHE): The task is divided into three parts, with each part being executed in a different location. One-third of the task is executed locally, one-third is executed at the mapping helper device and the remaining part is offloaded to the edge server.

The result in Figure 7 confirms the effectiveness of JMDOA in joint optimizing resources in terms of energy consumption. From Figure 7a, we observe when the average cycle for executing one bit is small, the energy consumption difference among the four algorithms is minimal. However, as the average cycle increases, AH exhibits the highest energy consumption, while JMDOA consistently has the lowest. This discrepancy can be attributed to the fact that the energy consumption of helper devices primarily depends on their computing ability, whereas the energy consumption of the edge server relies on its communication ability. With a small average cycle for executing one bit, the energy consumption of computing on the helper device and offloading to the edge server may be similar. Nevertheless, as the average cycle for executing one bit increases, the computing ability decreases, leading to lower energy consumption for JMDOA.

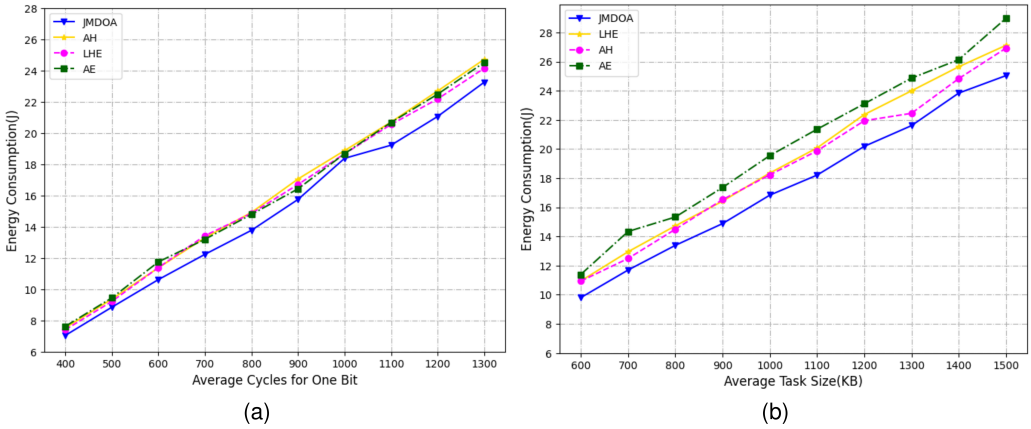

Figure 7. Effect of Joint Resource Allocation with the change of frequency and task size.

In Figure 7b, AE incurs the highest energy consumption, followed by LHE with relatively lower energy consumption and JMDOA has the lowest energy cost. When tasks are offloaded to the edge server, they are first offloaded to the helper devices. This implies that edge caching can accommodate more tasks for caching when the task size is small. The energy consumption gap between AH and AE gradually increases because the communication energy consumption for tasks that are not cached is higher than the computation energy consumption of the helper device. While LHE is somewhat inefficient in resource utilization. On the other hand, the joint resource allocation approach performs well in both scenarios, as it effectively balances computing and communication resources to make optimal choices.

5.3. Effect of Task Caching

In this simulation, we verify the effect of edge caching and demonstrate the superior performance of JMDOA's caching strategy. To achieve this, we conduct three contrasting tests.

Random caching (RC): The initial caching strategy is given randomly.

No caching (NC): All tasks are not cached in the edge server, i.e., $a = \{0, 0, \ldots, 0\}$.

Size base caching (SC): The initial caching strategy is determined by the task size. Tasks are cached in ascending order until the cache space limit is reached.

Figure 8 illustrates the energy consumption for different caching strategies, considering task sizes ranging from 600 KB to 1500 KB and average cycles for executing one bit varying from 400 to 1300. It is evident that caching tasks result in lower energy consumption compared to non-caching tasks, highlighting the significance of edge caching.

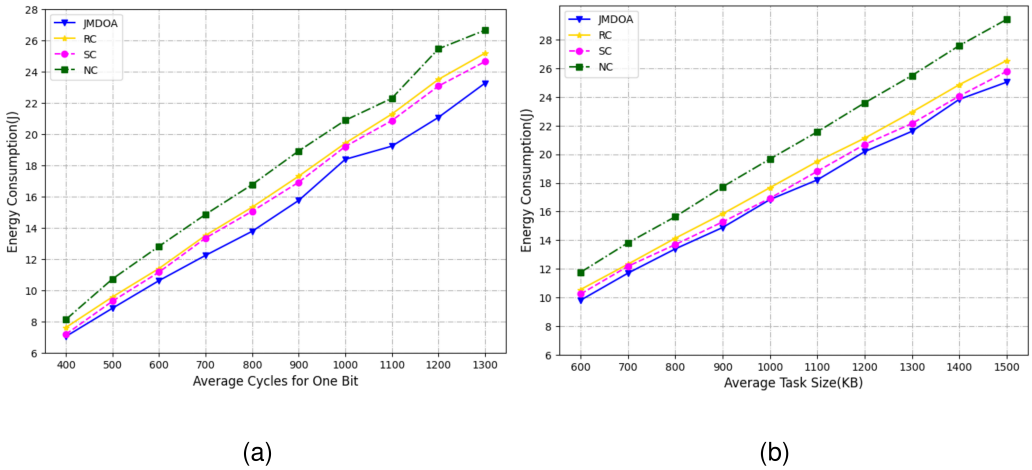

Figure 8. Effect of Task Caching with the change of frequency and task size.

Among the strategies evaluated, the NC strategy exhibits the least efficient performance due to its lack of task caching at the edge server, necessitating task offloading for execution. Conversely, the SC strategy, which considers the size of edge caching, outperforms the RC strategy by strategically caching tasks based on their size. Our proposed JMDOA achieves the lowest energy consumption. This approach not only takes into account factors influencing edge caching but also optimizes user association, task partitioning and transmission power in response to the actual situation.

The disparity in energy-consumption speed between Figure 8a,b can be attributed to the task size. When the task size is small, a larger number of tasks is cached in the edge server, resulting in slower energy consumption. However, as the task size increases, the number of cached tasks decreases, leading to a significant rise in energy consumption. Notably, as shown in Figure 8a, the computing ability has a minimal impact on task caching, further emphasizing the importance of task size in energy consumption.

6. Conclusions

To enhance resource utilization and reduce offload time in the mobile-edge-computing (MEC) system, particularly for devices that cannot directly connect to the MEC server, this paper introduces a novel system that integrates device-to-device (D2D) assistance and caching techniques. The primary objective of this system is to minimize energy consumption while improving task execution and offloading efficiency. By jointly optimizing computation, communication and caching, we formulate the problem as an integer-mixed non-convex problem. To address this challenge, we propose a Joint Multiple Decision Optimization Algorithm (JMDOA). Leveraging block coordinate descent and convex optimization techniques, the JMDOA algorithm provides an efficient and effective solution approach. The experimental results highlight the superior performance of the JMDOA algorithm. Compared to single caching strategies or resource allocation methods, our proposed method achieves lower energy consumption and significant energy savings with minimal time cost. In future research, we plan to conduct experiments involving power measurements on physical devices to validate the practical implementation of our system. Additionally, we will prioritize ensuring the privacy of the transmission process.

Author Contributions: Conceptualization, J.G. and J.L.; methodology, J.G. and G.X.; experiment, J.G. and Y.Z.; writing—original draft preparation, J.G.; writing—review and editing, Y.Z., J.L. and X.M.; visualization, H.C.; supervision, G.X. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the China Scholarship Council (CSC) (Grant No. 202106170092) and Jilin Province Science and Technology Development Plan Project under Grants 20200401076GX.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

We introduce $\beta_{i,b}^k = \mu_{i,b}^k V_{i,b}^k$, where $0 \le \mu_{i,b}^k \le 1$, $0 \le V_{i,b}^k \le V_m$, and then the constraint factor product can be obtained:

$$\begin{cases} \{[\mu_{i,b}^k - 0] \cdot [V_{i,b}^k - 0]\}_{LS} \\ \{[1 - \mu_{i,b}^k] \cdot [V_{i,b}^k - 0]\}_{LS} \\ \{[\mu_{i,b}^k - 0] \cdot [V_m - V_{i,b}^k]\}_{LS} \\ \{[1 - \mu_{i,b}^k] \cdot [V_m - V_{i,b}^k]\}_{LS} \end{cases}, m \in \mathcal{M}, i \in N_m(b), k \in \mathcal{K}, \tag{A1}$$

where $\{\cdot\}_{LS}$ denotes the steps of the linearization step of $\beta_{i,b}^k$=$\mu_{i,b}^k V_{i,b}^k$. Similarly, by introducing $\beta_{i,b}^k$ into Formula (23), we obtain the following expression:

$$\begin{cases} \beta_{i,b}^k \geq 0 \\ V_{i,b}^k - \beta_{i,b}^k \geq 0 \\ \mu_{i,b}^k V_m - \beta_{i,b}^k \geq 0 \\ V_m - \mu_{i,b}^k V_m - V_{i,b}^k + \beta_{i,b}^k \geq 0 \end{cases} ,m \in \mathcal{M}, i \in N_m(b), k \in \mathcal{K}. \tag{A2}$$

In the same way, denote $\gamma_{i,b}^k = \mu_{i,b}^k (V_m - V_{i,b})$, where $0 \leq \mu_{i,b}^k \leq 1, 0 \leq V_{i,b} \leq V_m$; then, the constraint can be obtained as follows:

$$\begin{cases} \gamma_{i,b}^k \geq 0 \\ V_{i,b} - \gamma_{i,b}^k \geq 0 \\ \mu_{i,b}^k V_m - \gamma_{i,b}^k \geq 0 \\ V_m - \mu_{i,b}^k V_m - V_{i,b} + \gamma_{i,b}^k \geq 0 \end{cases} ,m \in \mathcal{M}, i \in N_m(b), k \in \mathcal{K}. \tag{A3}$$

After substituting $\beta_{i,b}^k$ and $\gamma_{i,b}^k$ to the objective function and constraints in P_3, P_3 can be transferred to P_4 as:

$$\begin{aligned} P_4 : & \min_{V_I, \delta_I, \delta_K, \beta, \gamma} \sum_{b=1}^{B} f^{'}(V_I, \delta_I, \delta_K, \beta, \gamma) \\ s.t.\ & C13 : 0 \leq V_m - V_{i,b} - \beta_{i,b}^k \\ & C14 : 0 \leq \delta_{i,b}^k \leq \frac{\gamma_{i,b}^k}{B_k \log_2(1 + \frac{p^{max} h_{i,b}^k}{\sigma^2})} \\ & C15 : 0 \leq \delta_{k,b}^e \leq \frac{V_m - V_{i,b} - \beta_{i,b}^k}{B_k \log_2(1 + \frac{p^{max} h_{k,b}^e}{\sigma^2})} \\ & C16 : a_i^0 \frac{V_m C_m}{f_{k,b}^e} + (1 - a_m^0) \max\{ \frac{V_{i,b} C_m}{f_i^l}, \\ & \delta_{i,b}^k + \frac{\beta_{i,b}^k C_m}{f_k^l}, \delta_{i,b}^k + \delta_{k,b}^e + \frac{(V_m - V_{i,b} - \beta_{i,b}^k) C_m}{f_{k,b}^e} \} \leq T \\ & C17 : (23) \\ & C18 : (24), \end{aligned} \tag{A4}$$

where the function $f^{'}$ is denoted by:

$$\begin{aligned} f^{'} = & \min_{V_I, \delta_I, \delta_K, \beta, \gamma} \sum_{m \in \mathcal{M}} \sum_{i \in N_m(b)} \sum_{k \in \mathcal{K}} a_m^0 p^{wait} \frac{V_m C_m}{f_{k,b}^e} \\ & + (1 - a_m^0)(\kappa {f_i^l}^2 C_m V_{i,b} + \kappa \beta_{i,b}^k {f_k^l}^2 C_m \\ & + \frac{\sigma^2 (2^{\frac{\gamma_{i,b}^k}{\delta_{i,b}^k B_k}} - 1)}{h_{i,b}^k} \delta_{i,b}^k + \frac{\sigma^2 (2^{\frac{V_m - V_{i,b} - \beta_{i,b}^k}{\delta_{k,b}^e B_e}} - 1)}{h_{k,b}^e} \delta_{k,b}^e \\ & + p^{wait} \frac{(V_m - V_{i,b} - \beta_{i,b}^k) C_m}{f_{k,b}^e}). \end{aligned} \tag{A5}$$

The function $f(x) \triangleq A(2^{\frac{B}{x}} - 1)$ is a monotonically increasing convex function for $x > 0$ and $B > 0$. Its perspective function $f(x) \triangleq A(2^{\frac{B}{x}} - 1)x$ is also convex. Therefore,

the objective function and all constraints in problem P_4 are convex, making it a convex optimization problem. To solve this problem, we can employ convex optimization methods, such as the Lagrange Method or Interior Point Method, as outlined in [42]. In this case, we will utilize the Sequential Least-Square Quadratic Programming (SLSQP) method due to its superlinear speed in finding the solution. The solution of SLSQP is achieved through the following steps: Firstly, the nonlinear constraint problem is transformed using the Taylor formula, expanding the function of P_4 at the iterative point. This transformation yields a quadratic programming problem. The algorithm employs a quasi-Newton Hessian approximation with a BFGS update of the B-matrix and an L1-test function in the line search algorithm. To validate the accuracy of the results, the Python tool package pyOpt [43] is utilized.

References

1. Armbrust, M.; Fox, A.; Griffith, R.; Joseph, A.D.; Katz, R.; Konwinski, A.; Lee, G.; Patterson, D.; Rabkin, A.; Stoica, I.; et al. A view of cloud computing. *Commun. ACM* **2010**, *53*, 50–58. [CrossRef]
2. Zhang, Q.; Cheng, L.; Boutaba, R. Cloud computing: State-of-the-art and research challenges. *J. Internet Serv. Appl.* **2010**, *1*, 7–18. [CrossRef]
3. Abbas, N.; Zhang, Y.; Taherkordi, A.; Skeie, T. Mobile edge computing: A survey. *IEEE Internet Things J.* **2017**, *5*, 450–465. [CrossRef]
4. Mao, Y.; You, C.; Zhang, J.; Huang, K.; Letaief, K.B. A survey on mobile edge computing: The communication perspective. *IEEE Commun. Surv. Tutorials* **2017**, *19*, 2322–2358. [CrossRef]
5. Siriwardhana, Y.; Porambage, P.; Liyanage, M.; Ylianttila, M. A survey on mobile augmented reality with 5G mobile edge computing: Architectures, applications and technical aspects. *IEEE Commun. Surv. Tutorials* **2021**, *23*, 1160–1192. [CrossRef]
6. Wijethilaka, S.; Liyanage, M. Survey on network slicing for Internet of Things realization in 5G networks. *IEEE Commun. Surv. Tutorials* **2021**, *23*, 957–994. [CrossRef]
7. Gismalla, M.S.M.; Azmi, A.I.; Salim, M.R.B.; Abdullah, M.F.L.; Iqbal, F.; Mabrouk, W.A.; Othman, M.B.; Ashyap, A.Y.; Supa'at, A.S.M. Survey on device to device (D2D) communication for 5GB/6G networks: Concept, applications, challenges and future directions. *IEEE Access* **2022**, *10*, 30792–30821. [CrossRef]
8. Dai, X.; Xiao, Z.; Jiang, H.; Alazab, M.; Lui, J.C.; Dustdar, S.; Liu, J. Task co-offloading for d2d-assisted mobile edge computing in industrial Internet of things. *IEEE Trans. Ind. Inform.* **2022**, *19*, 480–490. [CrossRef]
9. Omidkar, A.; Khalili, A.; Nguyen, H.H.; Shafiei, H. Reinforcement-Learning-Based Resource Allocation for Energy-Harvesting-Aided D2D Communications in IoT Networks. *IEEE Internet Things J.* **2022**, *9*, 16521–16531. [CrossRef]
10. Pan, Y.; Pan, C.; Yang, Z.; Chen, M.; Wang, J. A caching strategy towards maximal D2D assisted offloading gain. *IEEE Trans. Mob. Comput.* **2019**, *19*, 2489–2504. [CrossRef]
11. Yu, S.; Dab, B.; Movahedi, Z.; Langar, R.; Wang, L. A socially-aware hybrid computation offloading framework for multi-access edge computing. *IEEE Trans. Mob. Comput.* **2019**, *19*, 1247–1259. [CrossRef]
12. Ouamri, M.A.; Barb, G.; Singh, D.; Adam, A.B.; Muthanna, M.; Li, X. Nonlinear Energy-Harvesting for D2D Networks Underlaying UAV with SWIPT Using MADQN. *IEEE Commun. Lett.* **2023**, *27*, 1804–1808. [CrossRef]
13. Pu, L.; Chen, X.; Xu, J.; Fu, X. D2D fogging: An energy-efficient and incentive-aware task offloading framework via network-assisted D2D collaboration. *IEEE J. Sel. Areas Commun.* **2016**, *34*, 3887–3901. [CrossRef]
14. Wu, D.; Cai, Y.; Hu, R.Q.; Qian, Y. Dynamic distributed resource sharing for mobile D2D communications. *IEEE Trans. Wirel. Commun.* **2015**, *14*, 5417–5429. [CrossRef]
15. He, Y.; Ren, J.; Yu, G.; Cai, Y. D2D communications meet mobile edge computing for enhanced computation capacity in cellular networks. *IEEE Trans. Wirel. Commun.* **2019**, *18*, 1750–1763. [CrossRef]
16. Dai, Y.; Sheng, M.; Liu, J.; Cheng, N.; Shen, X.; Yang, Q. Joint mode selection and resource allocation for D2D-enabled NOMA cellular networks. *IEEE Trans. Veh. Technol.* **2019**, *68*, 6721–6733. [CrossRef]
17. Zeydan, E.; Bastug, E.; Bennis, M.; Kader, M.A.; Karatepe, I.A.; Er, A.S.; Debbah, M. Big data caching for networking: Moving from cloud to edge. *IEEE Commun. Mag.* **2016**, *54*, 36–42. [CrossRef]
18. He, S.; Huang, W.; Wang, J.; Ren, J.; Huang, Y.; Zhang, Y. Cache-enabled coordinated mobile edge network: Opportunities and challenges. *IEEE Wirel. Commun.* **2020**, *27*, 204–211. [CrossRef]
19. Jiang, W.; Feng, G.; Qin, S.; Yum, T.S.P.; Cao, G. Multi-agent reinforcement learning for efficient content caching in mobile D2D networks. *IEEE Trans. Wirel. Commun.* **2019**, *18*, 1610–1622. [CrossRef]
20. Qian, Y.; Wang, R.; Wu, J.; Tan, B.; Ren, H. Reinforcement learning-based optimal computing and caching in mobile edge network. *IEEE J. Sel. Areas Commun.* **2020**, *38*, 2343–2355. [CrossRef]
21. Xia, X.; Chen, F.; He, Q.; Grundy, J.; Abdelrazek, M.; Jin, H. Online collaborative data caching in edge computing. *IEEE Trans. Parallel Distrib. Syst.* **2020**, *32*, 281–294. [CrossRef]
22. Li, Q.; Zhang, Y.; Li, Y.; Xiao, Y.; Ge, X. Capacity-aware edge caching in fog computing networks. *IEEE Trans. Veh. Technol.* **2020**, *69*, 9244–9248. [CrossRef]

23. Huang, Y.; Song, X.; Ye, F.; Yang, Y.; Li, X. Fair and efficient caching algorithms and strategies for peer data sharing in pervasive edge computing environments. *IEEE Trans. Mob. Comput.* **2019**, *19*, 852–864. [CrossRef]
24. Xia, X.; Chen, F.; He, Q.; Cui, G.; Lai, P.; Abdelrazek, M.; Grundy, J.; Jin, H. Graph-based data caching optimization for edge computing. *Future Gener. Comput. Syst.* **2020**, *113*, 228–239. [CrossRef]
25. Safavat, S.; Sapavath, N.N.; Rawat, D.B. Recent advances in mobile edge computing and content caching. *Digit. Commun. Netw.* **2020**, *6*, 189–194. [CrossRef]
26. Vigneri, L.; Spyropoulos, T.; Barakat, C. Quality of experience-aware mobile edge caching through a vehicular cloud. In Proceedings of the 20th ACM International Conference on Modelling, Analysis and Simulation of Wireless and Mobile Systems, Miami, FL, USA, 21–25 November 2017; pp. 91–98.
27. Yu, G.; Wu, J. Content caching based on mobility prediction and joint user Prefetch in Mobile edge networks. *Peer-to-Peer Netw. Appl.* **2020**, *13*, 1839–1852. [CrossRef]
28. Guo, J.; Song, B.; Chen, S.; Yu, F.R.; Du, X.; Guizani, M. Context-aware object detection for vehicular networks based on edge-cloud cooperation. *IEEE Internet Things J.* **2019**, *7*, 5783–5791. [CrossRef]
29. Dai, Y.; Xu, D.; Zhang, K.; Maharjan, S.; Zhang, Y. Deep reinforcement learning and permissioned blockchain for content caching in vehicular edge computing and networks. *IEEE Trans. Veh. Technol.* **2020**, *69*, 4312–4324. [CrossRef]
30. Wu, W.; Zhang, N.; Cheng, N.; Tang, Y.; Aldubaikhy, K.; Shen, X. Beef up mmWave dense cellular networks with D2D-assisted cooperative edge caching. *IEEE Trans. Veh. Technol.* **2019**, *68*, 3890–3904. [CrossRef]
31. Alkama, D.; Zenadji, S.; Ouamri, M.A.; Khireddine, A.; Azni, M. Performance of Resource Allocation for Downlink Non-Orthogonal Multiple Access Systems in Tri-Sectorial Cell. In Proceedings of the 2022 IEEE International Conference on Electrical Sciences and Technologies in Maghreb (CISTEM), Tunis, Tunisia, 26–28 October 2022; Volume 4, pp. 1–6.
32. Sylia, Z.; Cédric, G.; Amine, O.M.; Abdelkrim, K. Resource allocation in a multi-carrier cell using scheduler algorithms. In Proceedings of the 2018 4th International Conference on Optimization and Applications (ICOA), Mohammedia, Morocco, 26–27 April 2018; pp. 1–5.
33. You, C.; Huang, K.; Chae, H.; Kim, B.H. Energy-efficient resource allocation for mobile-edge computation offloading. *IEEE Trans. Wirel. Commun.* **2016**, *16*, 1397–1411. [CrossRef]
34. Ren, J.; Yu, G.; Cai, Y.; He, Y. Latency optimization for resource allocation in mobile-edge computation offloading. *IEEE Trans. Wirel. Commun.* **2018**, *17*, 5506–5519. [CrossRef]
35. Tran, T.X.; Pompili, D. Joint task offloading and resource allocation for multi-server mobile-edge computing networks. *IEEE Trans. Veh. Technol.* **2018**, *68*, 856–868. [CrossRef]
36. Wang, C.; Yu, F.R.; Liang, C.; Chen, Q.; Tang, L. Joint computation offloading and interference management in wireless cellular networks with mobile edge computing. *IEEE Trans. Veh. Technol.* **2017**, *66*, 7432–7445. [CrossRef]
37. Ning, Z.; Dong, P.; Kong, X.; Xia, F. A cooperative partial computation offloading scheme for mobile edge computing enabled Internet of Things. *IEEE Internet Things J.* **2018**, *6*, 4804–4814. [CrossRef]
38. Wen, W.; Cui, Y.; Quek, T.Q.; Zheng, F.C.; Jin, S. Joint optimal software caching, computation offloading and communications resource allocation for mobile edge computing. *IEEE Trans. Veh. Technol.* **2020**, *69*, 7879–7894. [CrossRef]
39. Sherali, H.D.; Adams, W.P. *A Reformulation–Linearization Technique for Solving Discrete and Continuous Nonconvex Problems*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2013; Volume 31.
40. Xu, Y.; Yin, W. A block coordinate descent method for regularized multiconvex optimization with applications to nonnegative tensor factorization and completion. *SIAM J. Imaging Sci.* **2013**, *6*, 1758–1789. [CrossRef]
41. Abdi, H.; Williams, L.J. Principal component analysis. *Wiley Interdiscip. Rev. Comput. Stat.* **2010**, *2*, 433–459. [CrossRef]
42. Boyd, S.P.; Vandenberghe, L. *Convex Optimization*; Cambridge University Press: Cambridge, UK, 2004.
43. Perez, R.E.; Jansen, P.W.; Martins, J.R. pyOpt: A Python-based object-oriented framework for nonlinear constrained optimization. *Struct. Multidiscip. Optim.* **2012**, *45*, 101–118. [CrossRef]