electronics

Article

Collaborative Computation Offloading and Resource Allocation in Cache-Aided Hierarchical Edge-Cloud Systems

Yanwen Lan [1], Xiaoxiang Wang [1,*], Chong Wang [2], Dongyu Wang [1,*] and Qi Li [1]

[1] The Key Laboratory of Universal Wireless Communications, Ministry of Education, Beijing University of Posts and Telecommunications, Beijing 100876, China; yanwen@bupt.edu.cn (Y.L.); liqi1287345989@bupt.edu.cn (Q.L.)

[2] The Internet of Things laboratory, Bohai Vocational and Technical College, Tianjin 300402, China; 18567005002@163.com

* Correspondence: cpwang@bupt.edu.cn (X.W.); dy_wang@bupt.edu.cn (D.W.)

Received: 16 October 2019; Accepted: 26 November 2019; Published: 30 November 2019

Abstract: The hierarchical edge-cloud enabled paradigm has recently been proposed to provide abundant resources for 5G wireless networks. However, the computation and communication capabilities are heterogeneous which makes the potential advantages difficult to be fully explored. Besides, previous works on mobile edge computing (MEC) focused on server caching and offloading, ignoring the computational and caching gains brought by the proximity of user equipments (UEs). In this paper, we investigate the computation offloading in a three-tier cache-assisted hierarchical edge-cloud system. In this system, UEs cache tasks and can offload their workloads to edge servers or adjoining UEs by device-to-device (D2D) for collaborative processing. A cost minimization problem is proposed by the tradeoff between service delay and energy consumption. In this problem, the offloading decision, the computational resources and the offloading ratio are jointly optimized in each offloading mode. Then, we formulate this problem as a mixed-integer nonlinear optimization problem (MINLP) which is non-convex. To solve it, we propose a joint computation offloading and resource allocation optimization (JORA) scheme. Primarily, in this scheme, we decompose the original problem into three independent subproblems and analyze their convexity. After that, we transform them into solvable forms (e.g., convex optimization problem or linear optimization problem). Then, an iteration-based algorithm with the Lagrange multiplier method and a distributed joint optimization algorithm with the adoption of game theory are proposed to solve these problems. Finally, the simulation results show the performance of our proposed scheme compared with other existing benchmark schemes.

Keywords: MEC; computation offloading; resource allocation; cache; D2D

1. Introduction

With the drastic development of sensors and wireless communication techniques, there is explosive growth in the number of mobile devices accessing wireless networks. It is foreseen that the mobile data traffic will increase even more significantly in the coming years [1]. Furthermore, user quipments(UEs) will be smarter and their accompanied novel, sophisticated applications will be more ubiquitous and pervasive, such as face recognition, interactive game and augmented reality [2]. However, these emerging applications and services need not only extensive computing capabilities and infer vast battery consumption but also high data rate, which throws out a challenge to UEs as their computing power and battery capability are constrained. The previous signal processing and transmission techniques applied in the conventional cellular networks may not be efficient to meet UEs' requirements of high throughput and adequate computational power. To improve the delay performance and operational costs of services in fifth-generation (5G) wireless networks, future communication networks not only need to support seamless wireless access but also to offer the provisioning of computational offloading for UEs [3,4].

To meet with such a challenge of limited computational capability of UEs, a typical paradigm of mobile edge computing (MEC) is proposed which combines wireless network service and cloud computing at the edge of the small cell networks (SCNs) [5]. In an MEC system, a huge number of heterogeneous ubiquitous and decentralized devices are enabled to communicate, potentially cooperate and perform computation offloading by uploading their computational tasks to the MEC server via access ratio networks [6,7]. UEs no longer need to offload all of their tasks (e.g., high-quality video streaming, mobile gaming, etc.) to the central and remote cloud. Thus their requirement can be satisfied at any time and anywhere. However, the limited computational capabilities of edge servers may not be sufficient when there exists competition for resources by a large number of devices. The hierarchical edge-cloud framework is effective in overcoming this problem. Compared with traditional MEC frameworks, the central cloudlets is set in the hierarchical edge-cloud framework are seen as supplements of the computing resources [8,9]. This architecture allows a chance of offloading the workload to high-tier servers and partially relieves the competition among UEs who are requesting delay-sensitive services. However, offloading to higher-tier servers will incur extra overheads of delay and energy consumption in the uplink wireless transmission. Therefore, the central cloud server is indispensable for tasks with a large amount of computation workload and with a small data size [10]. As the UEs' tasks are heterogeneous, the competition of resources for other type of tasks (e.g., tasks with a large data size) is still universal in the RAN, thus how to further relieve the competition is an urgent problem to be solved [11].

Device-to-device (D2D) communication and caching are proposed as technological components to tackle the data-rate problem. D2D enables devices to communicate directly and obtain the proximity gain, reuse gain, and hop gain [12–14]. Moreover, since requests of UEs are not simultaneous in the time domain, so in each period of task requesting, a large number of idle UEs would exist. The idle UEs can enable the requesting UEs to offload part of computation tasks to neighbor devices via D2D communication links. Applying caching to heterogeneous network nodes can satisfy the request of UEs in the local networks. By proactively predicting and caching the most popular contents in the storage facilities, the distances between the UEs and contents are shortened, which improves the performance of content delivering.

From now on, several recent works have investigated computation offloading and resource allocation strategies in the mobile edge systems [15–20]. The authors of [15–17] focused on the computation offloading strategy. By the optimization of offloading mode (e.g., local computing or edge offloading mode), the service delay and energy consumption would be largely saved. Some researchers investigated the communication resources allocation [18], computational resource allocation [19,20] in MEC-enabled systems to improve the resource efficiency. As wireless channel conditions of UEs have significant differences, the communication resource allocation and computational resource allocation can improve communication and computing efficiency, respectively. Some works addressed the joint allocation of radio and server resource allocation algorithms in various MEC systems [21–26]. The joint computation and communication resources can balance the delay cost and computation cost according to the UEs' servicing delay and energy cost. Recently, the authors of [27–29] integrated caching technology with computation offloading to optimize the offloading policy in MEC systems. By task caching, the transmission cost can be largely saved, the offloading cost can be largely saved and UEs' experience can be further improved.

Although the benefits of D2D communication and caching have received much attention in the MEC computing networks, the two technologies are considered separately in most works. Moreover, in their works about task offloading, server offloading mode and local offloading mode are mainly two considered modes with the assumption that the hardware and software resources can support all computing tasks. This assumption is impractical, because computing power of edge servers and UEs are both limited, which may result in poor performance, especially in a scenario of resource competition among a large number of UEs. Furthermore, the advantages of task caching in small base stations (SBSs) have been investigated in some works, but the potential benefits of task caching in UEs and how much gain will be obtained by cache-enabled D2D communication are still not explored in the scenario of MEC systems.

Therefore, this research focuses on the computation offloading and resource allocation in a cache-aided hierarchical edge-cloud system, featured by its SCN architecture. We consider tasks which can be partially offloaded and processed. Specially, four offloading modes are considered in this paper named cloud partial offloading, MEC partial offloading, cache-matched D2D partial offloading and cache-mismatched D2D partial offloading. Our aim to minimize the system cost with a trade-off between the energy consumption and serving delay while meeting the computation constraints of servers. The offloading strategy, CPU-cycle assignment in the cloud server and MEC server, and task offloading ratio are jointly optimized.

The contributions of this paper are listed as follows.

(1) We present the considered hierarchical edge-cloud models, which include cloud partial offloading, MEC partial offloading, cache-matched D2D partial offloading, and cache-mismatched D2D partial offloading. A joint optimization problem is established to minimize the total offloading cost. Particularly, the offloading cost is defined as a linear combination of the total energy consumption and the total offloading latency. We focus on the joint design of CPU cycle frequency, the offloading decision, and the offloading ration allocation with the constraints of the computational resources.

(2) From the observation of the joint optimization problem, it is mixed-integer nonlinear optimization problem (MINLP) and non-convex. To solve it, we decouple it into the offloading ratio allocation problem (ORAP), resource allocation problem (RAP) and offloading mode selection problem (OMSP), according to each offloading mode.

(3) We design an efficient distributed joint computation offloading and resource allocation optimization (JORA) scheme to solve the original optimization problem. Especially for the resource allocation problems in MEC and cloud partial offloading modes, the Lagrange multiplier method is adopted. With the solution of resource allocation problems, an alternative optimization algorithm is proposed to obtain an optimal solution to the offloading ratio problem. For the two D2D offloading modes, we use reformulation linearization-technique (RLT) to reformulate the ratio allocation problems into four linear optimization problems which can be solved in polynomial time. At last, we formulate the offloading optimization problem as a potential game and propose a distributed algorithm to obtain the optimal solution.

(4) We investigate the performance of the proposed scheme through extensive numerical experiments. Simulation results show that the proposed scheme outperforms other existing benchmark schemes.

The paper is organized as follows. In Section 2, we review the related works. The system model is presented in Section 3. The problem formulation and decomposition are given in Section 4. Section 5 provides the formulation of task offloading and resource optimization problem. Meanwhile, the solution to the problem is also given. Simulation results are described in Section 6. Finally, Section 6 concludes this paper.

2. Related Work

The MEC paradigm has attracted dramatic attention in both academia and industry over the past several years. From the perspective of users, most services need data transmission and resource allocation. Thus various works jointly optimize computation offloading and resource allocation in recent years. To the best of our knowledge, their research can be focussed on the following aspects: (i) cache-based data offloading. (ii) joint computation offloading and resource allocation (iii) joint caching, computation offloading and resource allocation.

2.1. Cache-Based Data Offloading

The Cache-based data offloading is known as caching the popular contents at the caching entities (e.g., users' equipment, MEC server, etc.) located at the edge network, with which the delay and energy consumption of end UEs who are requesting contents would be largely decreased. Liu et al. [30] proposed a mobility-aware coded probabilistic caching scheme in MEC-enabled SCNs. In their scheme, user mobility and distributed storage are jointly considered with the aim of throughput maximization. Hou et al. [31] proposed a proactive caching mechanism named Learning-based cooperative caching strategy based on MEC architecture to reduce transmission cost while improving UEs' QoE. He et al. [32] presented a social trust scheme to enhance the security of MSNs. They apply a novel deep reinforcement learning approach to optimally allocating the network resources. Sun et al. [33] presented a MEC-based mobile VR delivery framework to minimize the average required transmission rate. The main concern of task offloading strategy in their work is what, how and where to offload UEs' tasks with the current network conditions. Xu et al. [34] proposed an enhanced adaptive bitrate video delivery scheme with joint cache and radio resource allocation. A JCRA algorithm to solve the matching problem to make cooperation between cache and radio resources. The authors of [35] proposed a novel D2D caching policy for device caching with the adoption of stochastic-geometry in mmWave-Cellular Networks. A random caching policy is proposed and the offloading gain and the distribution of the content retrieval delay are optimized in their works. The works in [36] investigated the probabilistic caching placement in stochastic wireless D2D caching networks. A closed-form approximation of cache-aided throughput was obtained.

2.2. Joint Computation Offloading and Resource Allocation

The main concern of computation offloading strategy is how and where to offload UEs' tasks under current network conditions. Numerous works have been done to achieve an optimal offloading policy. Meanwhile, as the channel conditions and the requested tasks of UEs are heterogeneity, a joint resource allocation strategy includes channel allocation policy, transmit power allocation policy, and computational resource allocation policy are critical to the ultimate QoS of UEs. Some works consider a joint optimization of computation and resource. Zhao et al. [10] studied the scheduling of heterogeneous cloud to maximize the probability that tasks can have the delay requirements met. He et al. [14] integrated the D2D communications with MEC to further improve the computation capacity of the cellular networks. Yang et al. [16] considered small-cell network architecture for task offloading and optimized the offloading policies in order to achieve energy efficiency. Al-Shuwaili et al. [19] jointly optimized communication and computational resources for AR mobile applications with the consideration of inherent collaborative properties of data in the uplink, and data delivery in the downlink. The authors of [20] aimed to maximize the energy efficiency in UAV Based MEC System for IoT Devices. El Haber et al. [24] jointed optimization of computational cost and devices energy for task offloading in multi-tier edge-clouds while respecting the devices' latency requirement. In [26], the authors proposed a jointly optimization policy in heterogeneous networks in order to minimize the cost of system respect to the energy consumption, computation and transmission cost. The authors of [37] investigated the task allocation problem in MEC environment with mmWave technology. In their works, the backhaul bandwidth and the edge server resource allocation are jointly optimized to minimize the total task serving time. In [38], the authors presented a NFV-enabled architecture and proposed orchestration algorithms for VNF onboarding and scheduling.

2.3. Joint Caching, Computation Offloading and Resource Allocation

Noticing that content offloading can largely save the transmission resource and decrease the service delay, some works try to integrate caching with the computation offloading and resource allocation to further the performance in the MEC systems. Tan et al. [13] studied the virtual resource allocation for heterogeneous services which include data service and computation service.

Liu et al. [34] furthered the work in [13] and studied the joint offloading and resource allocation problem in blockchain-based video streaming systems. In their work, D2D technology was adopted to enhance the sharing of computing power to end users. But both the work in [13,39] ignored the competition among UEs. Hao et al. in [29] presented the task caching and resource allocation scheme for mobile edge computing to improve the energy efficiency of the system. But the tasks are caching in MEC server, which is far away from the request users compared with the other nearby idle users, which may exacerbate the transmission delay and the resource competition in MEC server. The authors of [40] exploited the computation-communication tradeoff in a multi-user multi-server MEC network to speed up the downloading phase with the transmission cooperation of multiple computing nodes. Liu et al. in [28] optimized the energy by designing a resource allocation policy in a cache-enabled MEC system. Unfortunately, they considered task caching was arranged in MEC server as the work of [29].

3. System Model

In this section, we introduce a cache-aided hierarchical edge-cloud system. In such a system, the SBSs installed with an MEC server can provide a set of UEs seamless access and an abundance of computing resources in their proximity. The cloud is seen as a supplement to MEC as it has sufficient computing resources and provides service via a fiber backhaul link with the relay of SBSs.

3.1. Network Model

We assume there are several SBSs that connecting the remote cloud server via a fiber backhaul link. The remote cloud center has sufficient computing resources but is far away from UEs, while the MEC server close to the UEs, has limited computing power. We model the cloud server as a large number of virtual machines. Each of them has a dedicated processing capacity of f_C (in cycles per time unit) [41]. Similarly, we model each of the MEC servers as a virtual machine with a processing power of f_M (in cycles per time unit). The SBSs help UEs relay their tasks to cloud server or MEC servers for processing. If there is more than one UE access to the same MEC server, the processing power of the associated server will be shared. Without loss of generality, we focus on a typical small cell and its associated UEs, as shown in Figure 1. Assume there are K UEs in the range of the typical cell, the set of UEs is denoted by $\mathcal{K} = \{1, 2, \cdots, K\}$.

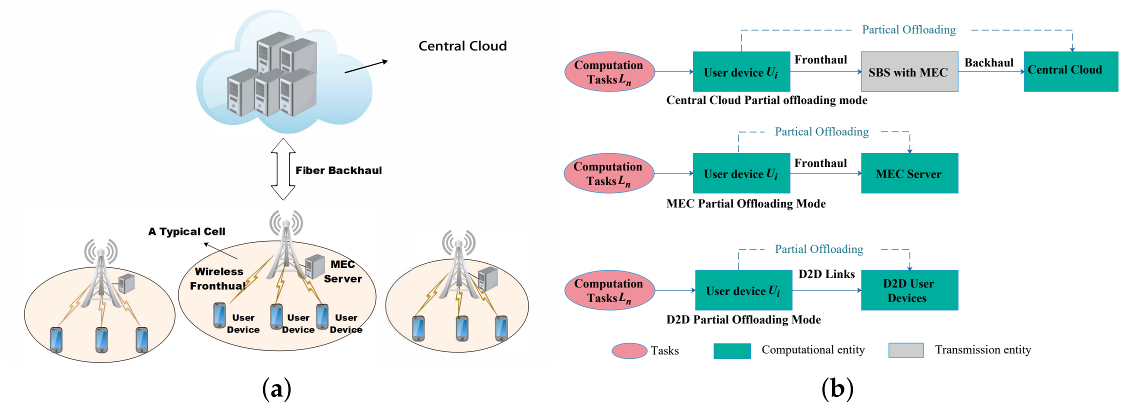


Figure 1. The considered device-to-device (D2D)-aided hierarchical edge-cloud system. (**a**) An overview of a hierarchical mobile edge computing (MEC) system. (**b**) Illustration of different computation offloading modes.

Assume each UE has a virtual reality application task (e.g., rendering scenes, or tracking objects) from a determinant task library need to be processed, and the requesting tasks are are heterogeneous in data size and computation capacity requirement. In our setting, UEs have a caching ability. A random caching scheme is adopted to keep the heterogeneity of tasks among UEs (e.g., uniformly probability random caching), which means all the UEs random cache tasks according to a determined probability distribution until the caching spaces are filled up. Task caching refers to caching the completed task applications and their related data onto UEs' local caching entities. In each offloading period, the UEs can choose to offload their tasks to MEC server, remote cloud, or perform them in the D2D networks. We call these UEs who have a task to be processed active UEs and treat the other UEs as idle UEs. Figure 1a illustrates an instant of such a cache-aided edge-cloud system with D2D communications.

The direct discovery strategy is considered in D2D networks [42]. In the requesting period, UEs participated in the device discovery process, periodically and synchronously transmit/receive discovery signals. In a device discovery period, an active UE would transmit discovery signals that may be detected by other UEs. The information in the discovery signals should include identity and application-related information (e.g., cache state). Thus, the active UEs would accurately establish connections with the most proper cache-matched or mismatched idle UEs. It should be mentioned that the device discovery period and the connection establishment period are all under control of UEs' serving SBS.A quasi-static scenario is considered where the set of UEs remains unchanged during a computation offloading period.

For each UE, there are four offloading modes that can choose, as shown in Figures 1b and 2:

(1) MEC partial offloading (mode 1)

In general, if an active UE cannot find adequate computing resources from the idle UEs nearby, no matter they have cached the requested task or not, the UE may be associated with Mode 1. After the allocation of offloading ratio, the offloading part of the task would be relayed and processed in the MEC server while the left part will be performed locally. It should be emphasized that the offloading and local processing are implemented simultaneously. To ensure the active UEs to experience a short delay and low energy consumption, the offloading ratio should be optimized with the consideration of competition between other active UEs in the same offloading mode. Moreover, the allocated CPU cycles should also be carefully designed.

(2) Cloud partial offloading (mode 2)

Similar to mode 1, for each UE with mode 2, they would additionally experience the transmission delay of backhaul link between SBSs and cloud center, but needn't consider the competition from other UEs compared with mode 1. The offloading part and the left part of the task are also be processed simultaneously. But different from mode 1, there is no competition for resources as there is no share of resources in the virtual machine of cloud centers. UEs are preferred to be associated with this mode if they cannot find adequate computing resources and the size of their request tasks is generally small, which results in a tiny transmit cost.

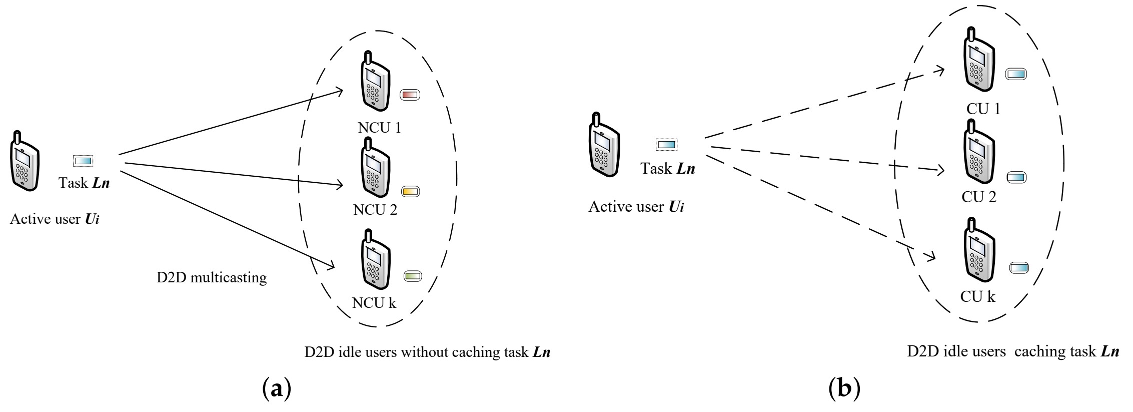


Figure 2. Illustration of two D2D partial offloading modes. (**a**) Cache-mismatched D2D partial offloading. (**b**) Cache-matched D2D partial offloading mode.

(3) Cache-mismatched D2D partial offloading (mode 3)

In this mode, the active UE would requisition all the computing resources of idle UEs nearby. In this case, as the allocated idle UEs have not cached the requesting task, the transmission from the active UEs to the occupied idle UEs for the offloading part should be involved in. For an active UEs, all the idle UEs have not cached the requesting task would formulate a potential collaborative offloading group. After the offloading ratio allocation, the idle UEs with nonzero offloading ratio would keep in the original group and the others would be released. The multicast scheme is adopted for offloading in this paper with a consideration of spectral efficiency. The size of tasks is considerable and the transmission cost should not be ignored which is different from that in the cache-matched D2D offloading mode. It should be emphasized that the whole data of task(e.g., the task-related database for computing) and the computing correlative parameters should be delivered to the hired UEs. The correlative computing parameters may include the input parameters and the computing ratio allocation decision for each hired UE. When the cache-mismatched UEs receive the data, they will decode the signal by a carefully designed decoding function. After the decoding period, they can get their workloads and perform processing.

(4) Cache-matched D2D partial offloading mode (mode 4)

As in this mode, cache-matched idle UEs have already cached the requesting tasks (e.g., task-related database). Thus the active UEs need only to transmit the related computation parameters as in mode 3. The parameters have a tiny data size compared with the whole task. We ignored the transmission delay and transmit energy consumption of UEs as they are far less than that of sending and processing the whole task [29]. So we only need to focus on the computing delay and energy consumption of each processing UEs. It needs to state that in all offloading modes if the offloading ratio is zero, that means the task will be processed locally.

We define the set of offloading strategies each UE could be associated with by $\mathcal{O} = \{DC, DN, M, C\}$. Specially, The indicators DC, DN, M and C mean cache-matched D2D partial offloading mode, cache-mismatched D2D partial offloading mode, MEC partial offloading mode and cloud partial offloading mode, respectively. In this paper, we don't consider the simultaneously offloading in different offloading modes as it will increase the complexity of signal scheduling, and the spectrum resources are limited which may not be enough to support simultaneous and multi-mode access for a large number of UEs.

3.2. Task Caching Model

Assume there is a task library consists of N computation tasks denoted by $\mathcal{N} = \{1, 2, \cdots, N\}$, and all the tasks have different task size and computation requirements. Tasks can be divided and be partially offloaded. Let $L_n(D_n, S_n)$ presents nth of the tasks, where D_n denotes the total number of instructions to be executed (in CPU cycles). Furthermore, S_n means the data sizes in bits of the task L_n. As the assumption in many works, we do not consider the transmission delay of task results and packer loss as the data size after processing. In this paper, the caching capacities of UEs are normalized as the number of caching tasks. Moreover, the caching capacities of all UEs are equally denoted by M.

The distribution of many Internet services was proven to follow Zipf's law. Similar to Internet services, the distribution of computing services also follows Zipf's law[43]. As the assumption in [29], in this paper, we assume that the requests of UEs follow the same Zipf distribution. The popularity set of tasks is denoted by $\mathcal{P} = (P_1, P_2, \cdots, P_N)$, in which the popularity is ranked in descending order. The popularity of tasks can be calculated by

$$P_n = \frac{n^{-\gamma}}{\sum\limits_{j=1}^{N} j^{-r}}, \forall n \in \mathcal{N}, \tag{1}$$

where the exponent γ is the popularity distribution parameter, which reflects the skewness of popularity.

3.3. The Communication and the Computation Models

There are two communication modes, including D2D communication and cellular communication. In this paper, all the UEs occupy the orthogonal spectrum in both D2D links and cellular links, and the cell reuses the bandwidth resources of another cell. That ensures the intercell interference among UEs can be removed no matter which mode they are associated with.

3.3.1. Mode 1: MEC Partial Offloading Mode

If an active UE is associated with MEC partial offloading mode, the UE would firstly offload its task to the SBS and then process it in MEC server. In this case, cellular communication will be applied. For a typical UE i in this case with the task L_n, the transmit rate R_i^T can be calculated by

$$R_i^T = B^C \log_2\left(1 + \frac{p_i H_i^M}{\sigma^2 + I_c}\right), \tag{2}$$

where B^C represents the bandwidth in cellular communication for each UEs, p_i is the transmit power of UE i, σ^2 is the additive white Gaussian noise. I_c denotes the interference coming from other adjacent cells. For the sake of simplicity, we regard I_c as a constant.

According to the Formula (2), the delay for UE i offloading task L_n to MEC server can be calculated by

$$t_{i,n}^{MT} = \theta_i^M S_n \Big/ R_i^T, \tag{3}$$

where $\theta_i^M \in [0,1]$ is the offloading ratio of UE i in MEC offloading mode. Specially, $\theta_i^M = 0 (\theta_i^M = 1)$ means task will be totally processed locally (in the server).

The energy consumption of transmission can be represented as

$$E_{i,n}^{MT} = p_i t_{i,n}^{MT}. \tag{4}$$

Let f_i^M denote the allocated computational resources in MEC server for UE i, the processing time can be expressed as

$$t_{i,n}^{MC} = \theta_i^M D_n \Big/ f_i^M. \tag{5}$$

According the works in [26], we can also get the energy consumption of partial processing task L_n in MEC server which can be calculated by

$$E_{i,n}^{MC} = \eta_m \theta_i^M D_n (f_i^M)^2, \tag{6}$$

where η_m is the energy effective switched capacitance of MEC server.

Similarly, let f_i^L denote the computing power of UE i, the computation delay and the energy consumption for processing local part of tasks can be expressed as

$$t_{i,n}^{ML} = (1 - \theta_i^M) D_n \Big/ f_i^L \tag{7}$$

$$E_{i,n}^{ML} = \eta_u (1 - \theta_i^M) D_n (f_i^L)^2, \tag{8}$$

where $t_{i,n}^{ML}$ and $E_{i,n}^{ML}$ denote the computing delay and energy consumption by local computing, η_u denotes the energy effective switched capacitance of UE i.

3.3.2. Mode 2: Cloud Partial Offloading Mode

The offloading procedure in this mode can be divided into two phases named as the access phase and the backhaul phase, respectively. The active UEs in this mode should firstly access the serving SBS and transmit their offloading part by wireless access front link, then relay them to cloud center for processing by fiber backhaul link. Therefore, the communication latency includes two parts, fronthaul transmission delay and backhaul transmission delay.

Similar to the Formula (3), let $\theta_i^C \in [0,1]$ denotes the offloading ratio of i in cloud offloading mode, the fronthaul transmission delay and energy consumption which are denoted as $t_{i,n}^{CT1}$, $E_{i,n}^{CT1}$, can be defined as the same as (3) and (4) by following the same notations.

For the backhaul link transmission, we consider the delay in backhaul from local access networks to the cloud center as a constant value. Moreover, the transmission energy consumption in backhaul link can be neglected as it is insignificant compared with the energy consumption of computation. Let T^C denote the backhaul link transmission delay for all tasks. So the total offloading delay can be expressed as $t_{i,n}^{CT} = t_{i,n}^{CT1} + T^C$.

The computation delays for processing the offloading part and the rest part of task L_n are denoted by $t_{i,n}^{CC}$ and $t_{i,n}^{CL}$ which can be expressed as

$$t_{i,n}^{CC} = \theta_i^C D_i / f_i^C \tag{9}$$

$$t_{i,n}^{CL} = (1-\theta_i^C) D_i / f_i^L, \tag{10}$$

where f_i^C represents the allocated computational power in cloud center for UE i.

Similar to the Formulas (6) and (8), we can get the computation energy consumption for the offloading part and the rest part of task L_n

$$E_{i,n}^{CC} = \theta_i^C \eta_c D_n (f_i^C)^2 \tag{11}$$

$$E_{i,n}^{CL} = (1-\theta_i^C) \eta_u \eta_u D_n (f_i^L)^2, \tag{12}$$

where $E_{i,n}^{CC}$ and $E_{i,n}^{CL}$ denote the energy consumption for cloud processing and local processing, η_c denotes the energy effective switched capacitance of cloud servers.

3.3.3. Mode 3: Cache-Mismatched D2D Partial Offloading Mode

For an active UE in cache-mismatched D2D partial computing mode, the UE will offload part of its task to nearby idle UEs who have not cached the requesting task. The multicast scheme will be adopted for transmission. So the transmission rate is depending on the UE with the worst channel state.

Furthermore, some idle UEs may satisfy more than one active UE's demands because the D2D coverage of active UEs may overlap. To simplify, in this paper, if an idle UE can be selected by one more active UE, it would associate to the nearest active UE.

For an active UE i in mode 3, let V_i^{NC} denote the set of potential idle UEs nearby who are within the D2D coverage of i without caching the requesting task L_n. So the multicasting transmit rate can be expressed as

$$R_i^D = B^D \log_2\left(1 + \frac{p_i h_{i,j}^{NC}}{\sigma^2 + I_d}\right), j = argmin\{h_{i,o}^{NC}\}_{o \in V_i^{NC}}, \tag{13}$$

where $h_{i,j}^{NC}$ denotes the channel gain between UE i and the idle UE j, I_d is the interfere from other D2D links with the same channel in other cells. B^D is the bandwidth for each active UE in the D2D offloading modes.

Let $\theta_{i,j}^{NC} \in [0,1]$ denote the allocated offloading ratio of the task requested by i to the UE j. Especially, the local process ratio of the task can be repressed as $\theta_{i,i}^{NC}$. Then we can get the multicasting delay of i which can be calculated by

$$t_{i,n}^{DT} = (1 - \theta_{i,i}^{NC})S_n / R_i^D. \tag{14}$$

Let $E_{i,n}^{DT}$ denotes the energy consumption of multicasting. Thus we have

$$E_{i,n}^{DT} = p_i t_{i,n}^{DT}. \tag{15}$$

The computation delay of UE j for helping process the task L_n can be calculated by

$$t_{i,j,n}^{DC} = \theta_{i,j}^{NC} D_n / f_j^L, j \in V_i^{NC} \cup i. \tag{16}$$

The energy consumption of UE j for processing a part of task L_n can be expressed as

$$E_{i,j,n}^{DC} = \theta_{i,j}^{NC} \eta_u D_n (f_j^L)^2, j \in V_i^{NC} \cup i. \tag{17}$$

3.3.4. Mode 4: Cache-matched D2D Partial Offloading Mode

In cache-matched D2D partial computing mode, the offloading and processing procedure are similar to that of the cache-mismatched D2D partial offloading mode. The only difference is that the multicasting of tasks is not needed because the hired UEs have already cached the request tasks of the active UEs. Thus, the transmission delay can be neglected. Let V_i^{DC} denote the set of idle UEs cached the requesting task L_n, the computation delay and energy cost in this mode can be expressed as the same formulations as Formulas (16) and (17) respectively by replacing V_i^{NC} by V_i^{DC}.

4. Problem Formulation and Solutions

In this section, we formulate the joint offloading and resource allocation problem. Considering the non-convexity of the joint optimization problem, we decompose it into multiple subproblems and propose corresponding algorithms to solve these problems. Then, a distributed joint optimization algorithm is presented to solve the joint offloading and resource allocation problem.

4.1. Problem Formation

Our aim is minimizing the system cost with a trade-off between task processing delay and energy consumption. Considering the fact that the different properties and performance criterions of delay and energy consumption, we use a weighted cost of delay and energy consumption of the system for task processing. We let $\rho_t \in (0,1]$ and $\rho_e \in [0,1]$ denote the normalized weight coefficients of the total process latency and the total energy consumption($\rho_e + \rho_t = 1$). They are used to characterize the impact of delay and energy consumption on the total cost. The values of these parameters are depend on the type of tasks. For example, for a delay-sensitive application, the system may set $\rho_t = 1$ and $\rho_e = 0$.

As introduced previously, a computation task can be offloaded in different modes. Thus for a typical active UE i, we denote its offloading strategies by $\mathbf{x}_i^o \in \{x_i^M, x_i^C, x_i^{NC}, x_i^{DC}\}(i \in \mathcal{K}, o \in \mathcal{O})$. Especially, $\mathbf{x}_i^o = 1, o \in \mathcal{O}$ means UE i is associated with offloading mode o.

For an active UE i, as its task can be computed at multiple computing entities (e.g., MEC server, cloud center or other idle UEs nearby), the total delay for processing tasks should be the maximum one of the handling delay in the involved entities. Thus, we have the total processing delay in different modes.

$$\begin{cases} t_{i,n}^{DC} = \max\left\{t_{i,j,n}^{DC}\right\}_{j\in V_i^{DC}\cup i}, if\ x_i^{DC}=1 \\ t_{i,n}^{NC} = \max\left\{t_{i,i,n}^{NC}, t_{i,n}^{NT}+t_{i,j,n}^{NC}\right\}_{j\in V_i^{NC}}, if\ x_i^{NC}=1 \\ t_{i,n}^{M} = \max\left\{t_{i,n}^{ML}, t_{i,n}^{MT}+t_{i,n}^{MC}\right\}, if\ x_i^{M}=1 \\ t_{i,n}^{C} = \max\left\{t_{i,n}^{CL}, t_{i,n}^{CT}+t_{i,n}^{CC}\right\}, else \end{cases} \tag{18}$$

Similarly, the total energy consumption can be expressed as

$$\begin{cases} E_{i,n}^{DC} = \sum\limits_{j\in V_i^{DC}\cup i} E_{i,j,n}^{DC}, if\ x_i^{DC}=1 \\ \mathrm{E}_{i,n}^{NC} = E_{i,n}^{DT} + \sum\limits_{j\in V_i^{DC}\cup i} E_{i,j,n}^{NC}, if\ x_i^{NC}=1 \\ E_{i,n}^{M} = E_{i,n}^{ML}+E_{i,n}^{MT}+E_{i,n}^{MC}, if\ x_i^{M}=1 \\ E_{i,n}^{C} = E_{i,n}^{CL}+E_{i,n}^{CT}+E_{i,n}^{CC}, else \end{cases} \tag{19}$$

In (18) and (19), the indicators $t_{i,n}^{DC}$, $t_{i,n}^{NC}$, $t_{i,n}^{M}$, $t_{i,n}^{C}$ present the total processing delay in cache-matched D2D partial offloading mode, cache-mismatched D2D partial offloading mode, MEC partial offloading mode and cloud partial offloading mode, respectively. Indicators $E_{i,n}^{DC}$, $E_{i,n}^{NC}$, $E_{i,n}^{M}$, $E_{i,n}^{C}$ present the total processing energy consumption in such four partial offloading modes, respectively.

Let Φ_i^{DC} and Φ_i^{NC} denote the sets of allocated offloading ratio to idle UEs of UE i in mode 3 and mode 4. To minimize the total cost of computation offloading in such system, the following optimization problem can be established

$$\min_{\substack{X_i^{DC}X_i^{NC}X_i^{C}X_i^{M}, \\ \Phi_i^{DC},\Phi_i^{NC}\theta_i^{M}\theta_i^{C},f_i^{M},f_i^{C}}} \quad C = \sum_{i\in\mathcal{K}}\sum_{o\in\mathcal{O}} x_i^o\left(\rho_t t_{i,n}^o + \rho_e E_{i,n}^o\right) \tag{20a}$$

$$\text{Subject to: } \sum_{o\in\mathcal{O}} x_i^o = 1, \forall i\in\mathcal{K} \tag{20b}$$

$$x_i^o \in \{0,1\}, \forall i\in\mathcal{K} \tag{20c}$$

$$0 \le f_i^M \le f_M, \forall i\in\mathcal{K} \tag{20d}$$

$$0 \le \sum_{i\in K} f_i^M \le f_M, \forall i\in\mathcal{K} \tag{20e}$$

$$0 \le f_i^C \le f_c, \forall i\in\mathcal{K} \tag{20f}$$

$$0 \le \theta_i^M \le 1, \forall i\in\mathcal{K} \tag{20g}$$

$$0 \le \theta_i^C \le 1, \forall i\in\mathcal{K} \tag{20h}$$

$$0 \le \theta_{i,j}^{DC} \le 1, \forall i\in\mathcal{K}, \theta_{i,j}^{DC} \in \Phi_i^{DC} \tag{20i}$$

$$0 \le \theta_{i,j}^{NC} \le 1, \forall i\in\mathcal{K}, \theta_{i,j}^{NC} \in \Phi_i^{NC} \tag{20j}$$

$$\theta_{i,i}^{DC} + \sum_{j\in V_i^{DC}} \theta_{i,j}^{DC} = 1, \forall i\in\mathcal{K} \tag{20k}$$

$$\theta_{i,i}^{NC} + \sum_{j\in V_i^{NC}} \theta_{i,j}^{NC} = 1, \forall i\in\mathcal{K}, \tag{20l}$$

where C denotes the value of the total cost. Constraints (20b) and (20c) ensure each UE can only choose one offloading mode in each task processing period. Constraints (20d) and (20f) mean the allocated computational capacity in MEC server and cloud server for each UE should be bounded within zero and its maximum computational power. Similarly, constraint (20e) bound the total allocated computational power for all UEs in MEC particle offloading mode. Constraints (20g) and (20h) state that offloading ratio in MEC partial offloading mode and cloud partial offloading mode should be limited between 0 and 1. Constraints (20i) and (20j) are proposed to guarantee the offloading ratio of each entitie for each active UE in cache-matched D2D offloading mode or cache-mismatched D2D offloading mode are limited between 0 and 1. Futhermore, constraints (20k) and (20l) are proposed to limit the total offloading ratio.

From the observation of task offloading optimization problem, we can see that $x_i^o \in \{0,1\}$ is binary, resulting in the nonconvexity of objective function. These attributes makes (20) become a mixed-integer nonlinear optimization problem (MINLP). Moreover, joint consideration of offloading ratio allocation, offloading mode selection and resource allocation makes centralized algorithms commonly with a high computation complexity. Furthermore, as the adoption of game theory, a centralized algorithm needs a large number of iterations to obtain the NE solution, which is impractical. So a distributed mechanism is extremely urgent to be designed to ensure the efficiency of the solution.

To solve the problem (20), We propose a distributed joint computation offloading and resource allocation optimization (JORA) scheme. In this scheme, we decompose it into three subproblems, which mean offloading ratio allocation problem (ORAP), resource allocation problem (RAP) and offloading mode selection problem (OMSP).

4.2. Resource Allocation Problem (RAP) in JORA

Recalling (20), the optimization of the CPU-cycles frequency of servers for each UE is related to the offloading ratio allocation strategies.

Now we discuss the resource allocation problem in JORA. Since the constraints given by (20b), (20c) and (20g)–(20i) are not related to the CPU-cycle frequency, so they can be overlooked in this subproblem. So the problem of computing resource optimization is equivalent to the following problem when $x_{i,n}^o, \theta_{i,n}^M, \theta_{i,n}^C$ are fixed:

$$\min_{f_i^M, f_i^C} C = \sum_{i \in K} x_i^o (\rho_t t_{i,n}^o + \rho_e E_{i,n}^o) \tag{21a}$$

$$\text{Subject to:} \quad (20d), (20e), (20f). \tag{21b}$$

It is obvious that if the offloading strategies of active UEs are fixed, the problem (20) is equivalent to minimize the cost accumulation of active UEs in their respective modes.

4.2.1. RAP in MEC Partial Offloading Mode

Now we discuss the resource allocation problem of the UEs in MEC offloading mode. By separating the cost of the system in MEC offloading mode from (22), the resource allocation problem of UEs in MEC partial offloading mode can be expressed as

$$\begin{gathered}\min_{f_i^M} C^M = \sum_{i \in \mathcal{K}} \rho_t \max\{t_{i,n}^{ML}, t_{i,n}^{MT} + t_{i,n}^{MC}\} + \rho_e E_{i,n}^M \\ \text{Subject to:} \quad (20d), (20e).\end{gathered} \tag{22}$$

The problem of (22) is continuous and non-differentiable due to the function of $\max\{\cdot\}$. The value of $\max\{\cdot\}$ is respected to the offloading ratio $\{\theta_i^M\}_{i \in \mathcal{K}}$. For a typical UE i, the total delay is related to the numerical relationship between local computing delay and offloading delay decided by the value of θ_i^M.

In order to solve the problem (22), we present a lemma as follows

Lemma 1. *The problem (22) is a convex optimization problem with respect to $\{f_i^M\}_{i\in\mathcal{K}}$ and it is equivalent to the following optimization problem for all UEs with $\theta_i^M \neq 0, \forall i \in \mathcal{K}$.*

$$\begin{aligned} &\min_{f_i^M} C^M = \sum_{i\in\mathcal{K}} \rho_t (t_{i,n}^{MT} + t_{i,n}^{MC}) + \rho_e E_{i,n}^M \\ &\textit{Subject to:} \quad (20d), (20e) \end{aligned} \tag{23}$$

Proof. Please refer to Appendix A. □

For the problem (23), we adopt Lagrange multiplier method to solve it. By applying the Karush–Kuhn–Tucker(KKT) conditions, the Lagrangian of problem of (23) can be expressed as

$$\begin{aligned} &\min_F L(F) + \mu \left(\sum_{i\in\mathcal{K},\theta_i^M\neq 0} f_i^M - f^M \right) \\ &\text{subject to:} \quad (20d), (20e), (20f). \end{aligned} \tag{24}$$

where $F = \{f_i^M\}_{i\in\mathcal{K}}$ and μ are values larger than 0.

For $\forall i \in \mathcal{K}$ and $\theta_i^M \neq 0$, the KKT conditions are as follows

$$\begin{cases} \nabla_{f_i^M} L(F) = -\dfrac{2\rho_t D_n \theta_i^M}{(f_i^M)^2} + 2\rho_e \kappa_M \theta_i^M D_n f_i^M = 0 \\ \mu \displaystyle\sum_{i\in U^M,\theta_i^M\neq 0} \left(f_i^M - f_M \right) = 0. \end{cases} \tag{25}$$

Let $[y]^+ = max\{y, 0\}$, combined with the condition (25), the Lagrange multipliers update as below.

$$\mu(t+1) = \mu(t) + \delta(t) \left[\sum_{i\in\mathcal{K},\theta_i^M\neq 0} f_i^M - f^M \right]^+ \tag{26}$$

where t is the current times of iteration, $\delta(t)$ represents the step of t-th iteration. By utilizing the KKT conditions, the optimal resource allocation solution can be found.

4.2.2. RAP in Cloud Partial Offloading Mode

Similar to RAP in MEC partial offloading, the resource allocation in cloud partial offloading mode can be separated from problem (20), and can be expressed as

$$\begin{aligned} &\min_{f_i^C} C^C = \sum_{i\in\mathcal{K}} \rho_t \max\{t_{i,n}^{CL}, t_{i,n}^{CT} + t_{i,n}^{CC}\} + \rho_e E_{i,n}^C \\ &\text{Subject to:} \quad (20f). \end{aligned} \tag{27}$$

By recalling Lemma 1, for the RAP in cloud partial offloading mode, we have a similar conclusion.

Lemma 2. *The problem (27) is a convex optimization problem with respect to $\{f_i^C\}$ and it is equivalent to the following optimization problem for all UEs with $\theta_i^C \neq 0$, $\forall i \in \mathcal{K}$.*

$$\begin{aligned} &\min_{f_i^C} C^C = \sum_{i\in K} \rho_t \left\{ t_{i,n}^{CT} + t_{i,n}^{CC} \right\} + \rho_e E_{i,n}^C \\ &\textit{Subject to:} \quad (20f). \end{aligned} \tag{28}$$

Proof. Please refer to the proof of Lemma 1. □

Then the following optimum solution can be obtained by solving the following equation:

$$\nabla_{f_i^C} C^C(f_i^C) = -\frac{2\rho_t D_n \theta_i^C}{\left(f_i^C\right)^2} + 2\rho_e \kappa_C \theta_i^C D_n f_i^C = 0, \forall i \in \mathcal{K}. \tag{29}$$

By solving the Equation (29), we can get:

$$f_i^{C*} = \sqrt[3]{\frac{\rho_t}{2\rho_e \kappa_c}}. \tag{30}$$

Based on (30), a closed-form solution of (27) can be expressed as follows:

$$f_i^{Copt} = \begin{cases} f_i^{C*}, f_i^{C*} < f_C \\ f_C, f_i^{C*} \geq f_C \end{cases} \tag{31}$$

4.3. *Offloading Ratio Allocation Problem* ***(ORAP)***

In order to trade off the delay cost and energy cost, the offloading ratio allocation in different offloading modes should be carefully optimized.

4.3.1. ORAP in MEC and Cloud Partial Offloading Modes

As discussed above, the cost of the system in different offloading modes are independent. For all active UEs in mode 1 and mode 2, if the offloading strategies $\{x_i^o\}_{o\in\mathcal{O}}, f_i^M$ and resource allocation policies f_i^C are fixed, the problem (20) in such two types of offloading modes can be equivalently seemed as solving the following offloading ratio allocation problem.

$$\begin{aligned} \min_{\theta_i^M, \theta_i^M} C = \sum_{i\in\mathcal{K}} x_i^M \left[\rho_t \max\left\{t_{i,n}^{ML}, t_{i,n}^{MT} + t_{i,n}^{MC}\right\} + \rho_e E_{i,n}^M\right] + x_i^C \left[\rho_t \max\left\{t_{i,n}^{CL}, t_{i,n}^{CT} + t_{i,n}^{CC}\right\} + \rho_e E_{i,n}^C\right] \\ \text{Subject to:} \quad (20g), (20h) \end{aligned} \tag{32}$$

As the offloading proportions of UEs $\{\theta_i^M\}_{i\in\mathcal{K}}$ are independent, the (32) is equivalent to minimizing the cost of each UE (e.g., *minimize* $C_i^M, \forall i \in \mathcal{K}$).

Assume for any active UE $i \in \mathcal{K}$ associated with MEC partial offloading mode. If the optimal solution have reached, it must fall into two cases as follows.

Case 1: $t_{i,n}^{ML} \geq t_{i,n}^{MT} + t_{i,n}^{MC}$

In this case, the local processing delay is larger than the offloading delay. We can get the constraint on θ_i^M as follows

$$\theta_i^{min} \geq \theta_i^M \geq 0 \tag{33}$$

where $\theta_i^{min} = \frac{D_n R_i^M f_i^M}{f_i^L S_n f_i^M + f_i^L R_i^M D_n + D_n R_i^M f_i^M}$ which meets that the value that $t_{i,n}^{ML} = t_{i,n}^{MT} + t_{i,n}^{MC}$.

The problem (32) can be rewritten as

$$\begin{aligned} \min_{\theta_i^M} C_i^{M1} =& \rho_t t_{i,n}^{ML} + \rho_e (E_{i,n}^{ML} + E_{i,n}^{MT} + E_{i,n}^{MC}) \\ \text{subject to:} \quad & 0 \leq \theta_i^M \leq min\{\theta_i^{min}, 1\}. \end{aligned} \tag{34}$$

By discussing the first derivative of the cost C_i^M to θ_i^M, $\nabla_{\theta_i^M} C_i^M(\theta_i^M) = 0$, we can get the optimal solution of offloading ratio as follows

$$\theta_i^{Mopt} = \begin{cases} 0, \ if \ \nabla_{\theta_i^M} C_i^{M1}(\theta_i^M) > 0 \\ \theta_i^{min}, \ if \ \nabla_{\theta_i^M} C_i^{M1}(\theta_i^M) \leq 0 \end{cases} \tag{35}$$

Case 2: $t_{i,n}^{ML} \leq t_{i,n}^{MT} + t_{i,n}^{MC}$

In this case, the local processing delay is smaller than the offloading delay. We can get the constraint on θ_i^M as follows

$$1 > \theta_i^M \geq \theta_i^{min}. \tag{36}$$

The problem (33) can be rewritten as

$$\begin{aligned} &\min_{f_i^M} C_i^{M2} = \sum_{i \in \mathcal{K}} \rho_t \left(t_{i,n}^{MT} + t_{i,n}^{MC} \right) + \rho_e (E_{i,n}^{ML} + E_{i,n}^{MT} + E_{i,n}^{MC}) \\ &\text{Subject to:} \quad \min\{\theta_i^{min}, 1\} \leq \theta_i^M \leq 1. \end{aligned} \tag{37}$$

As same as case 1, we can get the optimal solution by discussing the first derivative. The optimal solution in this case can be expressed as follows

$$\theta_i^{Mopt} = \begin{cases} \theta_i^{min}, \ if \ \nabla_{\theta_i^M} C_i^{M2}(\theta_i^M) \geq 0 \\ 1, \ if \ \nabla_{\theta_i^M} C_i^{M2}(\theta_i^M) < 0. \end{cases} \tag{38}$$

By summarizing the solutions in case 1 and case 2, the optimal offloading ratio θ_i^{Mopt} can be obtained by

$$\theta_i^{Mopt} = \arg\min \left\{ C_i^M(0), C_i^M(\theta_i^{min}), C_i^M(1) \right\}, \forall i \in \ \mathcal{K}. \tag{39}$$

The process of solving ORAP in cloud partial offloading mode is similar to that in MEC edge partial offloading mode. We can get a similar conclusion about the optimal offloading ratio, which is expressed as follows

$$\theta_i^{Copt} = \arg\min \left\{ C_i^C(0), C_i^C(\theta_i^{M*}), C_i^C(1) \right\}, \forall i \in \ \mathcal{K}. \tag{40}$$

4.3.2. ORAP in D2D Partial Offloading Modes

For all active UEs in D2D offloading modes, the computing resources are fixed, we only consider the ratio allocation to minimize the total cost in such modes.

Firstly, we discuss the D2D cache-mismatched partial offloading mode, as offloading delay is not needed in D2D cache-matched partial offloading mode which is the only difference.

For each UE in this mode, their cost are independent. Thus, the problem (20) in such offloading mode can be equivalently seemed as solving the offloading ratio allocation for each UE. For a UE i requesting task L_n, the offloading ratio allocation problem can be equivalently expressed as

$$\begin{aligned} &\min_{\Phi_i^{NC}} C_i^{DC} = \rho_t \max \left\{ t_{i,i,n}^{DL}, t_{i,n}^{DT} + \left\{ t_{i,j,n}^{DL} \right\}_{j \in V_i^{NC}} \right\} + \rho_e E_{i,n}^{NC}, \forall i \in \mathcal{K}, \forall n \in \mathcal{N} \\ &\qquad \text{Subject to:} \quad (20j), (20k) \end{aligned} \tag{41}$$

Assume the problem Φ_i^{NC*} is an optimal solution of the offloading ratio policy of (41).

Next, we analyze the cache-matched idle UE candidate with the maximum processing delay when the offloading ratio problem reaches the optimal solution.

Theorem 1. *For an active UE i, $(i \in \mathcal{K})$, when the optimal solution to the problem (41) is reached, the idle UEs with the worst computing power would experience the maximum processing delay compared with other idle UEs in the same group.*

Proof. Please refer to Appendix B. □

Assume the idle UE j is the UE with a maximum processing delay in the offloading group of i (e.g., $j = \arg\max\{t_{i,g,n}\}_{g \in V_i^{NC}}$). Under this condition we can further discuss the optimal solution to the problem (41). The problem (41) can be rewritten as

$$\min_{\Phi_i^{NC}} C_i^{DC} = \rho_t \max\left\{t_{i,i,n}^{DL}, t_{i,n}^{DT} + t_{i,j,n}^{DL}\right\} + \rho_e E_{i,n}^{NC}, \forall i \in \mathcal{K}, \forall n \in \mathcal{N} \tag{42a}$$

$$\text{subject to:} \quad \theta_{i,j}^{DC} D_n / f_j^L \geq \theta_{i,g}^{DC} D_n / f_g^L, \forall g \in V_i^{NC}, (20j), (20k), \tag{42b}$$

where (42b) ensures j is the cache-mismatched idle UE with the maximum processing delay which is the necessary condition of the optimal solution.

Similar to the UEs in cloud and MEC offloading modes, we analyze the cache-mismatched D2D ratio offloading mode in two cases. The optimal cost will be the smaller one of the costs in these two cases.

Case 1: $t_{i,i,n}^{DL} \geq t_{i,n}^{DT} + t_{i,j,n}^{DL}$

In this case, the problem (41) can be rewritten as:

$$\min_{\Phi_i^{NC}} C_i^{DC} = \rho_t t_{i,i,n}^{DL} + \rho_e E_{i,n}^{NC}, \forall i \in \mathcal{K}, \forall n \in \mathcal{N} \tag{43a}$$

$$\text{Subject to:} \quad \theta_{i,i}^{DC} D_n / f_i^L \geq \theta_{i,j}^{NC} D_n / f_i^L, \forall j \in v_i^{NC} \tag{43b}$$

(20j), (20k), (42b).

The problem (43) is a Linear programming problem(LP), which is solvable in polynomial time. We denote the optimal offloading ratio allocation policy by Φ_i^{NC1}.

Case 2: $t_{i,i,n}^{DL} < t_{i,n}^{DT} + t_{i,j,n}^{DL}$

The problem (42) can be rewritten as:

$$\min_{\Phi_i^{NC}} C_i^{DC} = \rho_t \left(t_{i,n}^{DT} + t_{i,j,n}^{DL}\right) + \rho_e E_{i,n}^{NC}, \forall i \in \mathcal{K}, \forall n \in \mathcal{N} \tag{44a}$$

$$\text{subject to:} \quad \theta_{i,i}^{DC} D_n / f_i^L \leq \theta_{i,j}^{NC} D_n / f_i^L, \forall j \in V_i^{NC} \tag{44b}$$

(20j), (20k), (42b).

The problem (44) is also a Linear programming problem(LP), which is solvable in polynomial time. We denote the optimal offloading ratio allocation policy by Φ_i^{NC2}.

In summary, the optimal solution of problem (41) can be got by comparing the cost obtained in two cases:

$$\Phi_i^{NC} = \arg\min\left\{C_i^{NC}\left(\Phi_i^{NC1}\right), C_i^{NC}\left(\Phi_i^{NC2}\right)\right\}, \forall i \in \mathcal{K}. \tag{45}$$

We can get the similar solution to the D2D cache-matched offloading mode:

$$\Phi_i^{DC} = \arg\min\left\{C_i^{DC}\left(\Phi_i^{DC1}\right), C_i^{DC}\left(\Phi_i^{DC2}\right)\right\}, \forall i \in \mathcal{K}, \tag{46}$$

where $C_i^{DC}(\Phi_i^{DC1})$ is the cost in case 1 when the active local processing delay is larger than the offloading delay, $C_i^{DC}(\Phi_i^{DC2})$ is the cost in case 2 when the active local processing delay is smaller than the offloading delay.

For the cache-matched D2D partial offloading mode, we can get the same conclusion as the only difference between the two modes is that the transmission delay is omitted in cache-matched D2D partial offloading mode.

The optimal allocated offloading ratio and optimal allocated computing resources are coupled in MEC and cloud partial offloading modes. An iteration-based algorithm is presented to find the optimal solution, which is shown in Algorithm 1.

Algorithm 1 Iteration-based algorithm for offloading ratio and resource allocation.

```
Require:  $\mathcal{K}$,$\mathcal{X}^O$
Ensure:   Current cost $\{C_i\}_{i\in\mathcal{K}}$.
 1: Initialize the offloading ratio for each UEs.
 2: for l = 1:L do
 3:   for i = 1 to K do
 4:     if $x_i^M = 1$ then
 5:       Calculate optimal resource allocation policy according to Formula (25) for all UEs.
 6:       Obtain the optimal value of $\theta_i$ for $\forall i \in \mathcal{K}$ according to (39).
 7:     else if $x_i^C = 1$ then
 8:       Calculate optimal resource allocation policy according to Formula (31) for all UEs.
 9:       Obtain the optimal value of $\theta_i$ for $\forall i \in \mathcal{K}$ according to (40).
10:     else if $x_i^{DC} = 1$ then
11:       Obtain the optimal value of $\Phi_i^{DC}$ for $\forall i \in \mathcal{K}$ according to (46).
12:     else if $x_i^{NC} = 1$ then
13:       Obtain the optimal value of $\theta_i$ for $\forall i \in \mathcal{K}$ according to (45).
14:     end if
15:   end for
16:   Update the value of $\mu(t+1)$ according to Formulas (25) and (26)
17: end for
18: return current cost $\{C_i\}_{i\in\mathcal{K}}$ according to the Formula (21b)
```

In Algorithm 1, the optimization results of the CPU-cycle frequency, the offloading ratio are updated alternately. In particular, during each iteration, the solutions of CPU-cycle frequency optimization can be obtained straightforwardly based on (25), (26) and (31). The offloading ratio allocation is optimized according to (39), (40), (45) and (46).

4.4. Offloading Mode Selection Problem (OMSP)

When the allocated offloading ratio and allocated computing resources given by (20) are fixed, the offloading mode selection problem can be written as

$$\begin{aligned} &\min_{x_i^{DC} x_i^{NC} x_i^C x_i^M} C = \sum_{i\in K} C_i \\ &\text{Subject to:} \quad (20b), (20c). \end{aligned} \tag{47}$$

As shown in (20), although the offloading ratio and the allocated resources are independent with the offloading decision $\{x_i^o\}_{i\in\mathcal{K}}$, the competition of computing resources in MEC server and D2D idle UEs lead to an interaction between active UEs when they are making the offloading decisions.

To determine which tasks should be offloaded, we formulate the interactions between the UE users as a strategic game. In each iteration process of the game, every active UE would make their current best offloading decision according to the current UEs' offloading decisions state.

We define game $g = (\mathcal{K}, \prod_{i\in\mathcal{K}} x_i^o, \{C_i\}_{i\in\mathcal{K}})$, where $\mathcal{K}$ is the set of players and x_i^o is the feasible strategy space of player i, C_i is the player i's cost with consideration of delay and energy consumption, which decides its payoff achieved from using computing services.

In the offloading game, each UE is one player and would select the optimal mode to minimize its cost (e.g., C_i) in response to the other UEs' strategies.

To obtain the strategies of all UEs, we first introduce the concept of the best response strategy.

Let matrix $\mathbf{X}^o_{-i}$ denote the offloading strategies of all UEs, excluding UE i. Each row of $\mathbf{X}^o_{-i}$ represents the offloading vector of a UE. According to the definition of Game Theory [44], the best response strategy for each UE can be expressed as

$$x_i^* = \arg\max C_i(\mathbf{x}_i^o, \mathbf{X}^o_{-i}) \tag{48a}$$

$$\text{Subject to:} \quad x_i^o \in \{0,1\}, i \in \mathcal{K}, \sum_{o\in\mathcal{O}} x_i^o = 1, o \in \mathcal{O}. \tag{48b}$$

We now introduce the NE as follows.

Definition 1. *(NE [44]) An offloading strategy profile $\{x_i^o*\}_{i\in\mathcal{K}}$ is a NE of game g if no player can further decrease the cost by unilaterally altering its strategy, i.e., for all $x_i^o{}' \in X_i, i \in \mathcal{K}$*

$$C(x_i^o{}', \mathbf{X}^{o'}_{-i}) \leq C(x_i^o, X^o_{-i}). \tag{49}$$

We next show that there exists an NE.

Definition 2. *A game is called a potential game if it exists a potential function Q such that for $\forall i \in \mathcal{K}$ and offloading vectors that satisfy:*

$$C_i(x_i^o, \mathbf{X}^o_{-i}) - C_i(x_i^{o'}, \mathbf{X}^{o'}_{-i}) = Q_i(x_i^o, \mathbf{X}^o_{-i}) - Q_i^o(x_i^o{}', \mathbf{X}^{o'}_{-i}). \tag{50}$$

The Nash has self-stability properties which makes the UEs in the game derive mutually satisfactory solution at the equilibrium. At the equilibrium, no one can improve their utilities by changing the offloading policy since the UEs are selfish to act in their interests in the non-cooperative offloading problem.

Propotion 1. *The following function is a potential function and $\mathfrak{g}$ are potential games for all UEs*

$$Q(\mathbf{x}_i^o, \mathbf{X}^o_{-i}) = (1 - x_i^M)\Big(\sum_{j\in\mathbf{K}, j\neq i} c_i^M + x_i^C C_i^C + x_i^{NC} C_i^{NC} + x_i^{DC} C_i^{DC}\Big) + x_i^M \sum_{j\in\mathcal{K}} C_j^M. \tag{51}$$

Proof. For each UE, it can choose four offloading modes for task processing. By substituting the value of x_i^o into the Formula (51), we can get the cost variation by changing the offloading decision.

We first discuss the case that switching offloading decision between cloud offloading mode and MEC offloading mode. Assume the initial choice of UE i is MEC partial offloading mode and then change the offloading mode to cloud offloading partial mode, (e.g., $\mathbf{x}_i^o = \{1,0,0,0\} \to \mathbf{x}_i^{o'} = \{0,1,0,0\}$.

In this case, we have:

$$Q(\mathbf{x}_i^o, \mathcal{X}^o_{-i}) = \sum_{i=1}^{K} C_i^M \tag{52}$$

$$Q(\mathbf{x}_i^{o'}, \mathcal{X}^o_{-i}) = a_{i,m} \sum_{j=1, j\neq i}^{K} C_j^M + C_i^C \tag{53}$$

By subtracting the Formulas (52) and (53), we can achieve that

$$Q(\mathbf{x}_i^o, \mathbf{X}^o_{-i}) - Q(\mathbf{x}_i^{o'}, \mathbf{X}^o_{-i}) = C_i^M - C_i^C = C\left(\mathbf{x}_i^o, \mathbf{X}^o_{-i}\right) - C(\mathbf{x}_i^{o'}, \mathbf{X}^o_{-i}). \tag{54}$$

From (54) we can find that the variation of (51) is equal to that of the cost function. Similarly, we can get that when switch the offloading decisions, same conclusion would be obtained, which proves that proposition 1 is established and the game must existence NE. When the iteration in this game increase to a certain number, the solution will reach the optimal solution. □

The distributed joint optimization algorithm is presented in the Algorithm 2. Now we analyze the complexity of Algorithms. Based on (39), Algorithm 1 can approach the optimal solution of resource and offload ratio allocation problem via L iterations, thus the computational complexity of Algorithm 1 can be estimated as O(LK). As the Algorithm 2 is established by the iteration of Algorithm 1. The computational complexity of Algorithm 2 can be analyzed as O(TLK^2+LK).

Algorithm 2 The distributed joint optimization algorithm.

```
Require: $C_i$, $\mathbf{X}^O$
Ensure: Optimal offloading policy $\mathbf{X}^{O*}$
1: for i = 1 to K do
2:     Each UEs switch the offloading modes.
3:     Calculate the current cost of the system by Algorithm 1 and update the offloading mode to the optimal current mode $x_i^{o'}$.
4: end for
5: while $l <= T$ or $\mathbf{X}^{\mathbf{O}} \neq \mathbf{X}^{\mathbf{O}'}$ do
6:     for i = 1 to K do
7:         Set $x_i^M = 1$, calculate the cost by Algorithm 1 and update the offloading mode with minimum cost $x_i^{o'}$.
8:     end for
9: end while
10: Output the optimal offloading policy $\mathbf{X}^{O*}$ and cost $\{C_i^*\}_{i \in \mathcal{K}}$;
```

5. Simulation Results and Discussions

In this section, we use computer simulation software MATLAB to evaluate the performance of our algorithms. We firstly give the simulation setup and some of the other schemes for comparison in this paper. Then we compare these schemes and analyze the cost performance which is the aim of our optimization.

5.1. Simulation Setup

The simulation results of the proposed JORA are presented in comparison with other algorithms. The simulation was conducted on simulator Matlab 2018. The simulation parameters are described as follows. We considered the system consists of 80 UEs and one SBS. In the simulation, there was an orth-hexagonal region, which was covered by an SBS located at the center, with 300 m in diameter [14]. The UEs were uniformly distributed in the SBS coverage area. According to the reference [13] where 40 UEs compete for a total 20 MB bandwidth, the wireless bandwidth for each UE in D2D and cellular links in this paper was set to 1 MHz. As the assumption in [16], the background noise is set to -100 dbm. The wireless channel gain H_i is modeled as $H_i = 127 + 30 \times \log d$ [29], where d is the distance between UE i and computational entities other than UEs themselves. For UEs, we assumed that the CPU clock speeds are randomly distributed among 800–900 MHz. Without loss of generality, we assumed that the computational resources of each virtual machine of the cloud server and the MEC server are 10 GHz and 20 GHz, separately. The average data size of tasks was randomly distributed between 0.5 MB and 2.5 MB. Meanwhile, the required computational resource for each task is randomly distributed between 1 Ghz and 3 Ghz. The other main parameters in the simulations are summarized in Table 1.

Table 1. Summary of other default simulation parameters.

Parameters	Values
Transmit power of UEs (p_u)	0.1 w
Coefficient per unit of energy (ρ_e)	0.5
Revenue coefficient per unit of delay (ρ_t)	0.5
The number of tasks in the library (N)	20
The number of UEs (K)	80

In the simulations, the cost of the system is presented in units. To evaluate the impact of different parameters, the proposed JORA was compared with three other methods in their respective frameworks.

- DCOA [15]. The DCOA scheme only focuses on offloading strategy in mobile cloud computing (MCC) adopting a distributed potential game. But dynamic resource allocation and collaborative calculation among users are not considered in this scheme.
- RND, which means all UEs randomly select the four offloading strategies. Moreover, the resources and offloading ratio are also randomly allocated.
- MEC W. Local [29]. We use computing systems and algorithms in [29]. In their strategy, UEs partially offload and process tasks in MEC. Moreover, the computing resources and offloading ratio are optimized by an alternating iterative algorithm.
- MEC W. D2D [39]. In this scheme, a group D2D UEs are seemed as the supplement to the MEC server. UEs can choose MEC partial offloading mode or D2D partial offloading mode. The offloading strategy and offloading ratio are optimized, but the computational resources allocation is not considered.

5.2. Performance Evaluation of Distributed JORA Scheme

5.2.1. The Impact of the Number of UEs to the Total Cost of the System

Considering that our aim is minimizing the total cost of the system, so, we illustrate the cost performance of our proposed JORA scheme and discuss the effects of UEs. The number of UEs can reflect the resource competition in edge servers and D2D networks, and it is one of the most important factors to the total cost. As shown in Figure 3, the ratio of number of active UEs to that of idle UEs was fixed to 1, the delay from SBS to cloud center T^C was set to 0.1 s, the cache sizes of UEs were equally fixed to 3. Observing Figure 3, we can find that as the number of UEs increased, total costs increased in all schemes. But compared with the other schemes, our proposed scheme has the best performance. Moreover, through the observation of increasing rates of all schemes, we can find that the proposed scheme had the lowest rate while the RND scheme experienced the worst performance. The reasons were that with the consideration of cached-D2D and the cloud center, the competition in MEC was largely alleviated in the proposed scheme. Moreover, as the number of active UEs and idle UEs increased, schemes without considering the ubiquitous and idle computing resources of UEs (e.g., RND) experienced a higher cost, whether the computing cost or the transmission cost. However, the schemes with D2D (e.g., JORA and MEC w. D2D) will provided more computing resources for active UEs. It can also be verified by the cost increasing rate of scheme MEC W. D2D, which was smaller than the other three schemes without considering the UEs' cooperation computing. We can also find that the scheme MEC w. Local had a better performance than the other three schemes. The reason was rooted in its carefully designed offloading ratio allocation policy and resource allocation policy, while these policies are partial missed in schemes MEC W. D2D and RND. We can also find in Figure 3 that there were small gaps between the costs in proposed JORA with different γ. The reasons can be concluded that as the parameter γ influences the concentration of tasks' popularity. When changing the values of γ, the amount of available idle computing resources nearby the active UEs in D2D networks will be changed but with low increase because of the density of UEs. Moreover, the caching scheme was fixed and the cache sizes of UEs were small, which also makes the gaps small compared with the total costs of UEs in all four offloading modes. The variation of costs in different γ is not so obvious, but we can also find that with the increasing of γ, the total cost decreases from the local amplification of Figure 3. It is because the increasing of γ can increase the concentration of tasks' popularity, which can enhance the hit probability of caching and further decrease the total cost.

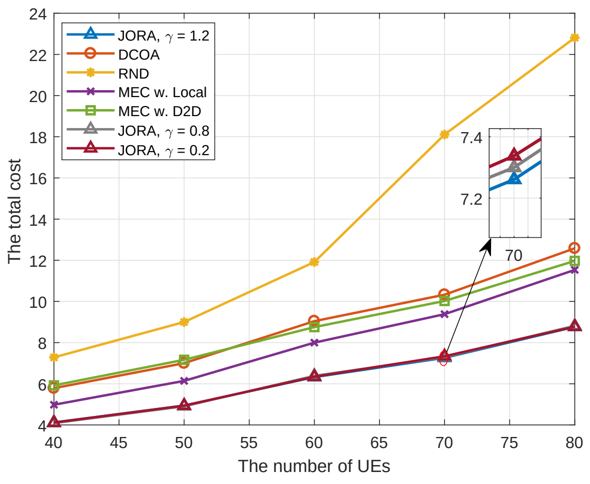


Figure 3. The total cost of the system over the average required computational resources of tasks.

5.2.2. The Effect of Weights among Impact Factors on the Serving Delay and Energy Consumption

The total cost formulated with a trade-off between delay and energy consumption. It is reflected by the delay weight and energy weight. Thus, we set different weights with the increasing number of UEs to find the influence of weights on the delay and energy consumption. In Figure 4, the delay and energy consumption obtained by our proposed JORA with varying number of UEs are evaluated.

In Figure 4a, we can find that as the number of UEs increases, the average processing delays of all schemes decrease. Besides, the average delay and its decreasing rate get higher with the increasing of the delay weight. The reason lies in: as the delay weight becomes larger, UEs will be allocated more proportion of tasks to the cloud center or MEC server in the proposed JORA scheme, which can cause a larger energy consumption but can largely save the processing delay. As the delay weight increases to a certain amount, the advantage of computing in servers is more obvious due to the huge computing resources compared with UEs' cooperative computing. Moreover, the number of serving idle UEs is fixed that results in the ultra-low computational delay.

In Figure 4b, we can find that as the number of UEs increases, the average energy consumption of all schemes decreases. Moreover, the average energy consumption and its increasing rate increase with the increase of energy weight. As the energy weight becomes larger, the energy consumption and its decreasing rate decline do too, no matter how the number of active UEs changes. The reasons can be explained that when there are very few active UEs, the computational resources in D2D networks and MEC server are adequate, which results in a lower average processing delay and energy consumption of active UEs. Moreover, as the number of active UEs increases, more UEs choose to offload their tasks or offload more proportion of their tasks to the servers smaller energy switch coefficient, which results in a larger decreasing rate of energy consumption. Furthermore, as the number of active UEs increases, the competition in the MEC server and in D2D networks was exacerbated. The cloud center can compensate for this shortcoming and bring about performance improvement with a small computation energy consumption compared with cooperative computing among UEs.

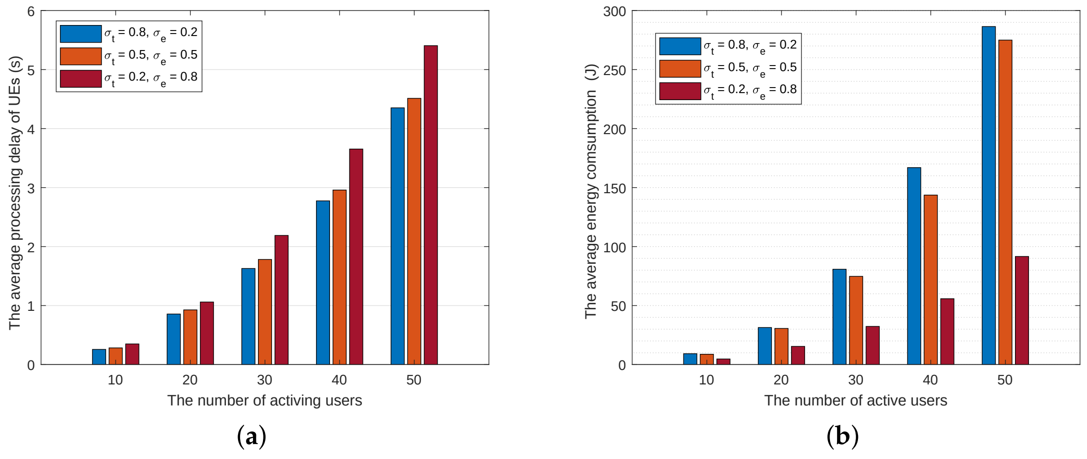


Figure 4. The influence of weights to the processing delay and energy consumption. (**a**) The average processing delay of UEs to the different weights. (**b**) The average energy consumption to the different weights.

5.2.3. The Effect of Cache Size of UEs to the Total Cost of the System

In our proposed scheme, cache-matched D2D offloading mode is deemed as a key offloading mode, which can relieve the resource competition in the servers and decrease the transmission cost. The performance of cache-match D2D offloading is largely affected by cache capacity. So, we evaluate the impact of the cache size of UEs on the average number of UEs, and analyze the influence to the offloading decision with different cache size of UEs. Figure 5 shows the impact of the cache size of UEs on the average number of UEs in each mode. In this simulation, we set the number of active UEs and the number of idle UEs to 50. The delay from SBS to cloud center T^C is fixed to 0.2 s.

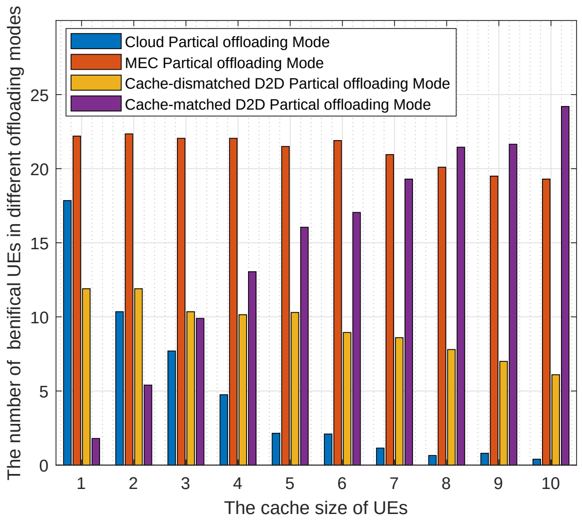


Figure 5. The beneficial UEs in different offloading modes to the average required computational resources of tasks.

We can observe from Figure 5 that the number of UEs associated with the cache-matched D2D partial offloading mode increases while the number of UEs in other offloading modes decreases in different degrees. The reason is that as the cache size of UEs increases, more active UEs have the chance to find the cache-matched idle UEs for task processing. The active UEs can use computational resources of the cache-matched idle UEs while task data transmission is not required, which can largely save the total processing delay and energy consumption caused by data transmission. Thus, comparing with the other three offloading modes, UEs in cache-matched offloading modes will experience a tiny transmission cost. By which reason, more active UEs which have chosen other modes further change their offloading decisions and participate in D2D cache-matched offloading modes.

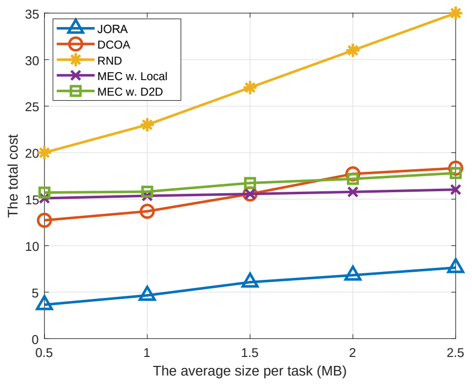
As shown in Figure 5, the number of UEs in MEC partial offloading mode and D2D cache-mismatched partial offloading modes slightly reduced while the number of UEs in cloud partial offloading modes decreases sharply. The reasons can be concluded as follows: (i) there exists a competition of active UEs between the cache-matched D2D offloading mode and the cache-mismatched offloading modes as the number of idle UEs is fixed. The increasing number of beneficial UEs in D2D cache-matched offloading mode certainly would influence the number of UEs in cache-mismatched D2D partial offloading modes. (ii) Comparing with the cache-mismatched D2D offloading mode, the cache-matched offloading mode has obvious advantages because of a smaller transmission cost, which would seize the UEs who originally chose the cache-mismatched offloading mode. (iii) Comparing to costs of cloud partial offloading mode, UEs in the MEC partial offloading mode wouldn't experience backhaul delay. Thus the decreasing of the beneficial number of UEs in edge computing modes are largely coming from the cloud offloading mode.

5.2.4. Effect of Task Size and Workload to the Total Cost

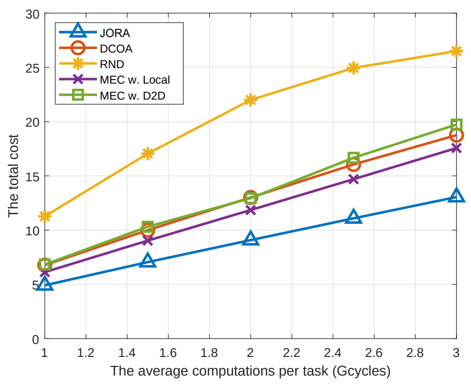
The nature of the task itself determines the cost of transmission or computation, so, we show the influence of the average data size of tasks and average of the workload of tasks to the total cost.

The influence of the average data size of tasks to the total cost is illustrated in Figure 6a. We fix the average computation workload of tasks to $1.2 Gcycles$ and the number of UEs to 30. We can see from Figure 6a, as the average data size of tasks increases, the total costs in all schemes increase too, but our scheme has the best performance compared with other schemes. The reasons can be concluded as follows. Firstly, it is obvious that more average required computational resources of tasks means more energy cost and larger delay of UEs, as both the energy and delay consumption of computing increase. Secondly, the offloading cost will get higher with the increases of average data size. Thus UEs will experience a larger transmission delay and energy consumption in a fixed offloading ratio. To decrease the cost, UEs may choose an offloading mode with the lowest increasing rate of cost (e.g., D2D partial offloading). Compared with other schemes, the D2D partial offloading modes include cache-matched D2D partial offloading mode and cache-mismatched D2D offloading mode have greater advantages versus computing by cloud center and MEC servers in saving delay. According to Figure 6a, we can see that the cost of scheme MEC W. Local scheme has a slightly increasing from 15 to 16. The reason is that the mechanism of partial offloading leads to a higher computation proportion of tasks, which can relieve the consequences bring by the increasing of task size. From Figure 6a, the scheme DCOA has a larger increasing rate than scheme MEC w. Local and scheme MEC w. D2D. As the average size per task increases from 0.5 MB to 1.5 MB, the cost of DCOA is smaller than the two schemes. As the average size per task increases from 1.5 to 2.5, the cost of DCOA gradually exceeds that of the two schemes. The reason is that as the transmission cost and transmission consumption are proportional to the amount of task data, the transmission cost from MEC to cloud in DCOA is absented in other two schemes. Moreover, when the the average size of task is small (e.g., 0.5–1.5 MB), the extra cost can be supplied the gap in DCOA as there is no competition in cloud server compared with the other two schemes. But when the average task size increase to a certain amount (e.g., exceeds 1.5 MB), the advantage of sufficient computing resources is weakened by the cost of transmission, which leads to a lower cost compared with the other two schemes.

Figure 6b illustrates the workload of tasks to the total cost. In this part, the average data size of tasks is fixed to 1.5 MB, and the number of UEs is set to 40, which can better display the trends of schemes with D2D communication. The average workload of tasks in changed from $1 Gcycles$ to $3 Gcycles$. As shown in Figure 6b, the total costs increase with the increase of workload for all schemes. Moreover, by the comparison with other schemes, the proposed scheme has the best performance. The reason is that as the workload of tasks increases, more computation energy will consumed, which increases the total cost. Besides, the existence of more offloading modes in JORA enables UEs to choose an offloading strategy with the lowest cost according to the characteristics of the tasks requested.



(**a**)



(**b**)

Figure 6. Effect of task size and workload to the total cost. (**a**) The average total cost to the different average size per task. (**b**) The average total cost to the different average computations per task.

5.2.5. The Influence of Backhaul Delay to the Average cost of the System

The cloud server is seemed as the a supplement of edge servers and collaborative D2D computing, so the backhaul transmission delay is the key point that largely affects the total cost. In this part, we study the influence of idles UEs on the number of beneficial users that offload their computation tasks.

To analyze the impact of latency between SBSs and cloud center on the average cost in this system, we simulate the total cost with a different number of UEs and different backhaul latency settings in Figure 7. The ratio of number of active UEs to that of idle UEs is fixed to 1. As shown in Figure 7, the average system cost for the tasks' processing per UE tends to rise with the increasing number of UEs. It is because the competition for computing resources intensifies with the increases number of active UEs, as the limit resources in idle UEs and MEC server. Thus, the average cost of the system would increase in each mode.

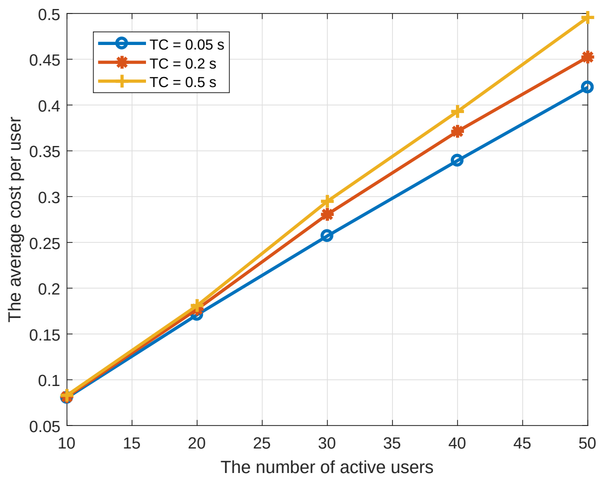


Figure 7. The influence of backhaul delay to the average cost of the system.

In addition, as the backhaul transmission delay increases from 0.05 s to 0.5 s, the average cost gets higher with fixed UEs' numbers. The reason is that the increasing number of UEs intensify the resource competition in MEC server, thus more UEs offload tasks to the cloud center which has additional transmission delays compared with computing in MEC server. As we know, it is universal of the phenomenon of resource competition in MEC. Thus the allocated computing resources in MEC server may not be sufficient. When backhaul transmission delay decreases, the backhaul transmission cost decreases, thus the advantage of computing in cloud is highlighted as it has adequate computing resources in cloud center.

5.2.6. The Beneficial UEs with Different Ratio of Active UEs to Idle UEs

It is easy for us to understand that the ratio of active UEs to idle UEs directly affects the cost in D2D offloading modes, which is decided by available computing resources and the degree of competition in the D2D networks. Thus, it is meaningful to evaluate the influence of the ratio to UEs' offloading decision. So in this part, we illustrate the beneficial UEs with different ratios of active UEs to idle UEs.

In Figure 8, we change the ratio of active UEs to the idles UEs and evaluate the number of active UEs in each offloading mode. In this setting, the total number of UEs are 40 and the cache size of UEs is 5.

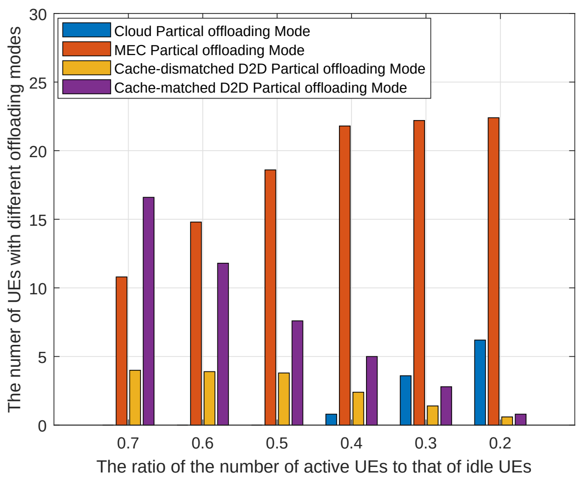


Figure 8. The beneficial UEs with different ratios of active UEs to idle UEs.

From Figure 8, we can observe that the number of UEs in the two D2D partial offloading modes decreases with the ratio increases. Moreover, the number of UEs in MEC partial offloading mode has a little increasing while the number of UEs in cloud offloading mode grows significantly. It is because with the increasing of ratio, the idle UEs in D2D networks are fewer which result in lack of computation resource in D2D computing networks. Thus, a large number of actives UEs could not find sufficient idle computing resources nearby, which motivate them to choose other offloading modes. Comparing with cloud partial offloading mode, MEC partial offloading has a huge advantage as only fronthaul transmission delay is considered in the case. But as the ratio increases, more active UEs who originally choose D2D partial offloading modes participate in the competition of MEC servers, which results in an increasing number of UEs in MEC offloading mode with an aggravating of resource competition. So, some of these UEs should be associated with cloud offloading mode, when the resources of MEC server are saturated.

6. Conclusions

In this paper, we investigated the offloading and resource allocation in a three-tier system which includes cloud center, MEC servers and D2D UEs. We considered that each UE has a task request from the task library and needs to make a decision on the task execution. Four processing modes were specifically considered, including cache-matched D2D partial mode, cache-mismatched D2D partial mode, MEC partial offloading mode and cloud partial offloading mode. A cost minimization problem was formulated with a trade of service delay and consumption. Particularly, the offloading decision, the computational resources and the offloading ratio in each mode were all optimized. Offloading decision, the computational resources and the offloading ratio were optimized for each mode. To solve the nonlinear and non-convex problems, we designed an iterative algorithm that converges the stationary optimal solution with polynomial computational complexity.

Author Contributions: Conceptualization, Y.L.; data curation, C.W.; formal analysis, Y.L. and C.W.; funding acquisition, X.W. and D.W.; investigation, Y.L.; methodology, Y.L.; project administration, X.W.; resources, Y.L.; software, Y.L.; supervision, Y.L., X.W. and D.W.; validation, Y.L., C.W. and Q.L.; visualization, D.W.; writing—original draft, Y.L.; writing—review and editing, C.W. and Q.L.

Funding: This research was funded by in part by "the National key R&D Program of China No.2018YFC1504502," and in part by "the National Natural Science Foundation of China under Grant 61701038."

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

Notations

Symbol	Definition
K	The number of UEs
N	The number of tasks in the task library
$\mathcal{K}$	The set of UEs
$\mathcal{N}$	The library of tasks
$\mathcal{X}^0$	The offloading strategy set of UEs
P_n	The requesting probability of task L_n
p_u	The transmit power of UEs
R_i^T	The average transmit rate of UE i in cellular networks
θ_i^C	The offloading ratio of UE i in cloud partial offloading mode
θ_i^M	The offloading ratio of UE i in MEC partial offloading mode
θ_i^M	The offloading ratio of UE i requesting task L_n in MEC partial offloading mode
$\theta_{i,j}^{NC}$	The offloading ratio of UE i to idle UE j in cache-mismatched D2D partial offloading mode
$\theta_{i,j}^{DC}$	The offloading ratio of UE i to idle UE j in cache-matched D2D partial offloading mode
Φ_i^{NC}	The set of offloading ratio of UE i in cache-mismatched D2D partial offloading mode
Φ_i^{DC}	The set of offloading ratio of UE i in cache-mismatched D2D partial offloading mode
$t_{i,n}^{MT}$	The average transmit delay of UE i requesting task L_n in MEC mode
$t_{i,n}^{CT}$	The uploading time of task L_n to cloud center for UE i
$t_{i,n}^{L}$	The local computing time by MUE i for task L_n
$t_{i,n}^{C}$	The computation execution time of UE i with cloud computing for task L_n
$t_{i,n}^{M}$	The computation execution time of task
T^C	The backhaul delay between MEC servers and cloudlets
ρ^t	The revenue coefficient per unit of saved delay
ρ^e	The revenue coefficient per unit of saved energy
$E_{i,n}^{LC}$	The energy consumption for processing task L_n locally
$E_{i,n}^{l}$	The energy consumption of MUE i for task L_n processing locally
$E_{i,n}^{MC}$	The energy consumption of UE i for task L_n processing in MEC
$E_{i,n}^{CC}$	The energy consumption of UE i for task L_n processing in cloud
$E_{i,m,n}^{u}$	The energy consumption of transmission in uplink from MUE i to FN m for task L_n
C^o	The total cost of the system in offloading mode o

Appendix A. Proof of Lemma 1

For each UEs in MEC partial offloading mode, we can discuss its cost into two cases by comparing the values of local computation delay and MEC processing delay. We first rewrite the problem (23) by merging the energy cost and energy cost of each UEs into the max function which is presented as follows.

$$\begin{aligned} &\min_{f_i^M} C^M = \sum_{i \in \mathcal{K}} x_i^M \min\left\{C_i^{M1}, C_i^{M2}\right\} \\ &\text{subject to} \quad (20d), (20e), \end{aligned} \tag{A1}$$

where C_i^{M1} which is presented in Formula (35) means the cost of UE i when the local computation delay for the left proportion of task is larger than that of MEC total processing delay include data transmission delay and computation delay, C_i^{M2} which is presented in Formula (37) means the cost in the case that the former delay is smaller than the latter delay.

Firstly, we discuss the convexity of the problem (A1). The function of C_i^{M1} and C_i^{M2} for any UE $i \in \mathcal{K}$ in Formula (A1) are both continuity and differentiable. Their convexity can be verified as follows

$$\begin{cases} \nabla^2_{f_i^M} C_i^{M1} = 2\rho_e \eta_M \theta_i^M D_n \succ 0 \\ \nabla^2_{f_i^M} C_i^{M2} = \dfrac{2\rho_t D_n \theta_i^M}{f_i^M} + 2\rho_e \eta_M \theta_i^M D_n \succ 0. \end{cases} \tag{A2}$$

As a max function of two convex functions is also a convex function according to the conclusion in [45], so the cost function for processing tasks for each UE $min\{C_i^{M1}(f_i^M), C_i^{M2}(f_i^M)\}_{\forall i \in \mathcal{K}}$ is convexity. From the observation of problem (A1), we find that it is a linearity summing of the cost function of UEs and its constraints (20d), (20e) are also linear, we can get the conclusion that the problem (A1) is convex optimization problem.

Secondly, we will use the counter-evidence method to prove the equivalence between the problem (A1) and the problem (24). Assume the solutions $\mathcal{F}^* = \{f_1^*, f_2^*, \cdots, f_K^*\}$ is optimal solution to problem (A1). If Lemma 1 is not held, there must be at least one idle UE j whose $f_j^*(j \in \mathcal{K})$ satisfies the conditions in the follows

$$C_j^{M2}(f_j^*) \le C_j^{M1}(f_j^*), 0 < f_i^* < min\{f_{max}^j, 1\}, \tag{A3}$$

where f_{max}^j is the value of f_j^* that satisfies $C_i(f_j^*)^{M1} = C_i(f_j^*)^{M2}$. As the first derivative of its cost is larger than zero. So the optimal solution of resource allocation for j must be 0, which is a conflict to the setting in Lemma 1. Thus Lemma 1 is proofed.

Appendix B. Proof of Theorem 1

For an active UE i in D2D cache-mismatched partial offloading mode, assume j is the idle UE with a maximum computing delay when the optimal task ratio allocation is reached.

Thus we have

$$j = \arg\max\left\{t_{i,j,n}^{DC}\right\}_{j \in V_i^{NC}} \tag{A4}$$

By substituting related variables into (A4), we can rewrite the Formula (A4) as

$$\theta_{i,j}^{NC} \geqslant \frac{\theta_{i,g}^{NC} f_j^L}{f_g^L}, \forall g \in V_i^{NC} \backslash j. \tag{A5}$$

Now, we break this optimal ratio allocation by changing the workload allocated to j. If we reduce the offloading ratio $\theta_{i,j}^{NC}$ by an infinitesimal value $\triangle\theta$ (e.g., $\triangle\theta \to 0^+$) to a new value of ratio $\theta_{i,j}^{NC'}$ and allocate the rest workload to any of other idle UEs. Thus, the total cost for processing the task of i is impossible to decrease.

Next we discuss the delay of local computing and offloading delays in idle UEs. After the reallocation of the ratio of j, the longest processing delay among idle UEs may change or not.

Case 1: there is no changing of the longest processing delay of idles UEs.

If this case occurs, as the total offloading data size of idle UEs is not changed, the multicast delay is as same as that before. Thus, there must be at least one idle UE which have the same processing delay as UE j in the optimal ratio allocation solution. Let U_i^{NC} denotes the set of idles UEs that with a smaller computing delay than j's when meeting to the optimal ratio solution. The computing delay of idle UE, which receives the reallocated workload is one of UEs in U_i^{NC}.

$$\frac{\theta_{i,n}^{NC} D_n}{f_j^L} > \frac{\theta_{i,j}^{NC} D_n}{f_g^L}, g \in U_i^{NC}. \tag{A6}$$

The increment of cost after changing ratio can be expressed as

$$\Delta C_i^{NC} = \rho_e \left(-\Delta\theta \rho_u D_n \left(f_j^L\right)^2 + \Delta\theta \rho_u D_n \left(f_g^L\right)^2 \right). \tag{A7}$$

As discussed before, the optimal solution is broken up which results in the current cost is not less than that of changing before, which leads to $\Delta C_i^{NC} \geq 0$. Thus, we can easily get that $f_j^L \leq f_g^L$. If we replace idle UE g by any other UEs in U_i^{NC}, the case would still occur, and the same conclusion can be also be obtained. In other words, if this case occurs, when the optimal ratio allocation solution meets, the processing delay of idles UEs can be directly presented by that of idle UEs with the worst computing power, as shown in Theorem 1.

Case 2: the longest processing delay of idles UEs are changed.

As $\Delta\theta$ is infinitesimal value, the idle UE j is also the UE with the worst computing delay among other idles UEs. In this case, there must be at least one idle UEs with a less computing delay and the idle UE received the reallocated workload of the task is one of such UEs. As same as that in case 1, we let U_i^{NC} denote the set of such idle UEs.

After reallocation, the cost increment can be presented as:

$$\Delta C_i^{NC} = -\frac{\rho_t \Delta\theta D_n}{f_j^L} - \rho_e \left(\Delta\theta \eta_u D_n \left(f_j^L\right)^2 - \Delta\theta \eta_u D_n \left(f_g^L\right)^2 \right). \tag{A8}$$

As in case 1, by letting $\Delta C_i^{NC} \geq 0$, we have

$$\rho_e \left(\triangle\, \theta \eta_u D_n \left(f_g^L\right)^2 - \triangle\, \theta \eta_u D_n \left(f_j^L\right)^2 \right) \geqslant \frac{\rho_t \triangle \theta D_n}{f_j^L} > 0. \tag{A9}$$

So we can easily know that $f_j^L \leq f_g^L$. If we replace idle UE g by any other UEs in U_i^{NC}, the case would still occur and the same conclusion can be also be got. If this case occurs, when the optimal ratio allocation solution meets, the processing delay of idles UEs can be directly presented by that of idle UEs with the worst computing power(e.g., the computing delay of UE j).

By concluding the two cases, the Theorem 1 can be proofed.

References

1. Peng, M.; Sun, Y.; Li, X.; Mao, Z.; Wang, C. Recent advances in Cloud radio access networks: System architectures, key techniques, and open issues. *IEEE Commun. Surv. Tutor.* **2016**, *18*, 2282–2308. [CrossRef]
2. Mao, Y.; You, C.; Zhang, J.; Huang, K.; Letaief, K.B. A Survey on Mobile Edge Computing: The Communication Perspective. *IEEE Commun. Surv. Tutor.* **2017**, *19*, 2322–2358. [CrossRef]
3. Li, Y.; Jin, D.; Hui, P.; Han, Z. Optimal base station scheduling for device-to-device communication underlaying cellular networks. *IEEE J. Sel. Areas Commun.* **2016**, *34*, 27–40. [CrossRef]
4. Wang, H.; Xu, F.; Li, Y.; Zhang, P.; Jin, D. Understanding mobile traffic patterns of large scale cellular towers in urban environment. In Proceedings of the 2015 ACM Internet Measurement Conference, Tokyo, Japan, 28–30 October 2015; pp. 225–238.
5. Xu, J.; Ren, S. Online learning for offloading and autoscaling in renewable-powered mobile edge computing. In Proceedings of the 2016 IEEE Global Communications Conference (GLOBECOM), Washington, DC, USA, 4–8 December 2016; pp. 1–6.

6. Hu, Y.C.; Patel, M.; Sabella, D.; Sprecher, N.; Young, V. *Mobile Edge Computing—A Key Technology towards 5G*; White Paper 11; ETSI: Sophia Antipolis, France, September 2015.
7. Liu, M.; Liu, Y. Price-Based Distributed Offloading for Mobile-Edge Computing With Computation Capacity Constraints. *IEEE Wirel. Commun. Lett.* **2018**, *7*, 420–423. [CrossRef]
8. Tong, L.; Li, Y.; Gao, W. A hierarchical edge cloud architecture for mobile computing. In Proceedings of the IEEE INFOCOM 2016—The 35th Annual IEEE International Conference on Computer Communications, San Francisco, CA, USA, 10–14 April 2016; pp. 1–9.
9. Ceselli, A.; Premoli, M.; Secci, S. Mobile edge cloud network design optimization. *IEEE/ACM Trans. Netw.* **2017**, *25*, 1818–1831. [CrossRef]
10. Zhao, T.; Zhou, S.; Guo, X.; Niu, Z. Tasks scheduling and resource allocation in heterogeneous Cloud for delay-bounded mobile edge computing. In Proceedings of the 2017 IEEE International Conference on Communications (ICC), Paris, France, 21–25 May 2017; pp. 1–7.
11. Taleb, T.; Samdanis, K.; Mada, B.; Flinck, H.; Dutta, S.; Sabella, D. On Multi-Access Edge Computing: A Survey of the Emerging 5G Network Edge Cloud Architecture and Orchestration. *IEEE Commun. Surv. Tutor.* **2017**, *19*, 1657–1681. [CrossRef]
12. Fodor, G.; Dahlman, E.; Mildh, G.; Parkvall, S.; Reider, N.; Mikl'os, G.; Tur'anyi, Z. Design aspects of network assisted device-to-device communications. *IEEE Commun. Mag.* **2012**, *50*, 170–177. [CrossRef]
13. Tan, Z.; Yu, F.R.; Li, X.; Ji, H.; Leung, V.C.M. Virtual Resource Allocation for Heterogeneous Services in Full Duplex-Enabled SCNs With Mobile Edge Computing and Caching. *IEEE Trans. Veh. Technol.* **2018**, *67*, 1794–1808. [CrossRef]
14. 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]
15. Chen, X.; Jiao, L.; Li, W.; Fu, X. Efficient multi-user computation offloading for mobile-edge Cloud computing. *IEEE/ACM Trans. Netw.* **2016**, *24*, 2795–2808. [CrossRef]
16. Yang, L.; Zhang, H.; Li, M.; Guo, J.; Ji, H. Mobile Edge Computing Empowered Energy Efficient Task Offloading in 5G. *IEEE Trans. Veh. Technol.* **2018**, *67*, 6398–6409. [CrossRef]
17. Zhang, H.; Guo, F.; Ji, H.; Zhu, C. Combinational auction-based service provider selection in mobile edge computing networks. *IEEE Access* **2017**, *5*, 13455–13464. [CrossRef]
18. Al-Shuwaili, A.; Simeone, O. Energy-efficient resource allocation for mobile edge computing-based augmented reality applications. *IEEE Wirel. Commun. Lett.* **2017**, *6*, 398–401. [CrossRef]
19. Qian, L.P.; Feng, A.; Huang, Y.; Wu, Y.; Ji, B.; Shi, Z. Optimal SIC Ordering and Computation Resource Allocation in MEC-Aware NOMA NB-IoT Networks. *IEEE Internet Things J.* **2019**, *6*, 2806–2816. [CrossRef]
20. Du, Y.; Wang, K.; Yang, K.; Zhang, G. Energy-Efficient Resource Allocation in UAV Based MEC System for IoT Devices. In Proceedings of the 2018 IEEE Global Communications Conference (GLOBECOM), Abu Dhabi, UAE, 9–13 December 2018; pp. 1–6.
21. You, C.; Huang, K.; Chae, H.; Kim, B.-H. Energy-efficient resource allocation for mobile-edge computation offloading. *IEEE Trans. Wirel. Commun.* **2017**, *16*, 1397–1411. [CrossRef]
22. Zhao, P.; Tian, H.; Qin, C.; Nie, G. Energy-Saving Offloading by Jointly Allocating Radio and Computational Resources for Mobile Edge Computing. *IEEE Access* **2017**, *5*, 11255–11268. [CrossRef]
23. Yang, L.; Zhang, H.; Li, X.; Ji, H.; Leung, V.C.M. A Distributed Computation Offloading Strategy in Small-Cell Networks Integrated with Mobile Edge Computing. *IEEE/ACM Trans. Netw.* **2018**, *26*, 2762–2773. [CrossRef]
24. El Haber, E.; Nguyen, T.M.; Assi, C. Joint Optimization of Computational Cost and Devices Energy for Task Offloading in Multi-Tier Edge-Clouds. *IEEE Trans. Commun.* **2019**, *67*, 3407–3421. [CrossRef]
25. Tran, T.X.; Pompili, D. Joint Task Offloading and Resource Allocation for Multi-Server Mobile-Edge Computing Networks. *IEEE Trans. Veh. Technol.* **2019**, *68*, 856–868. [CrossRef]
26. Zhang, J.; Xia, W.; Yan, F.; Shen, L. Joint Computation Offloading and Resource Allocation Optimization in Heterogeneous Networks with Mobile Edge Computing. *IEEE Access* **2018**, *6*, 19324–19337. [CrossRef]
27. Wang, D.; Lan, Y.; Zhao, T.; Yin, Z.; Wang, X. On the Design of Computation Offloading in Cache-Aided D2D Multicast Networks. *IEEE Access* **2018**, *6*, 63426–63441. [CrossRef]
28. Liu, P.; Xu, G.; Yang, K.; Wang, K.; Meng, X. Jointly Optimized Energy-Minimal Resource Allocation in Cache-Enhanced Mobile Edge Computing Systems. *IEEE Access* **2019**, *7*, 3336–3347. [CrossRef]
29. Hao, Y.; Chen, M.; Hu, L.; Hossain, M.S.; Ghoneim, A. Energy Efficient Task Caching and Offloading for Mobile Edge Computing. *IEEE Access* **2018**, *6*, 11365–11373. [CrossRef]

30. Liu, X.; Zhang, J.; Zhang, X.; Wang, W. Mobility-Aware Coded Probabilistic Caching Scheme for MEC-Enabled Small Cell Networks. *IEEE Access* **2017**, *5*, 17824–17833. [CrossRef]
31. Hou, T.; Feng, G.; Qin, S.; Jiang, W. Proactive Content Caching by Exploiting Transfer Learning for Mobile Edge Computing. In Proceedings of the GLOBECOM 2017—2017 IEEE Global Communications Conference, Singapore, 4–8 December 2017; pp. 1–6.
32. He, Y.; Yu, F.R.; Zhao, N.; Yin, H. Secure Social Networks in 5G Systems with Mobile Edge Computing, Caching, and Device-to-Device Communications. *IEEE Wirel. Commun.* **2018**, *25*, 103–109. [CrossRef]
33. Sun, Y.; Chen, Z.; Tao, M.; Liu, H. Communications, Caching and Computing for Mobile Virtual Reality: Modeling and Tradeoff. *IEEE Trans. Commun.* **2019**, *67*, 7573–7586. [CrossRef]
34. Xu, X.; Liu, J.; Tao, X. Mobile Edge Computing Enhanced Adaptive Bitrate Video Delivery with Joint Cache and Radio Resource Allocation. *IEEE Access* **2017**, *5*, 16406–16415. [CrossRef]
35. Giatsoglou, N.; Ntontin, K.; Kartsakli, E.; Antonopoulos, A.; Verikoukis, C. D2D-Aware Device Caching in mmWave-Cellular Networks. *IEEE J. Sel. Areas Commun.* **2017**, *35*, 2025–2037. [CrossRef]
36. Chen, Z.; Pappas, N.; Kountouris, M. Probabilistic Caching in Wireless D2D Networks: Cache Hit Optimal Versus Throughput Optimal. *IEEE Commun. Lett.* **2017**, *21*, 584–587. [CrossRef]
37. Noghani, K.A.; Ghazzai, H.; Kassler, A. A Generic Framework for Task Offloading in mmWave MEC Backhaul Networks. In Proceedings of the 2018 IEEE Global Communications Conference (GLOBECOM), Abu Dhabi, UAE, 9–13 December 2018; pp. 1–7.
38. Sarrigiannis, I.; Ramantas, K.; Kartsakli, E.; Mekikis, P.-V.; Antonopoulo, A.; Verikoukis, C. Online VNF Lifecycle Management in a MEC-enabled 5G IoT Architecture. *IEEE Internet Things J.* **2019**. [CrossRef]
39. Liu, M.; Yu, F.R.; Teng, Y.; Leung, V.C.M.; Song, M. Distributed Resource Allocation in Blockchain-Based Video Streaming Systems with Mobile Edge Computing. *IEEE Trans. Wirel. Commun.* **2019**, *18*, 695–708. [CrossRef]
40. Li, K.; Tao, M.; Chen, Z. Exploiting Computation Replication for Mobile Edge Computing: A Fundamental Computation-Communication Tradeoff Study. Available online: https://arxiv.org/abs/1903.10837 (accessed on 26 March 2019).
41. Shah-Mansouri, H.; Wong, V.W.S. Hierarchical Fog-Cloud Computing for IoT Systems: A Computation Offloading Game. *IEEE Internet Things J.* **2018**, *5*, 3246–3257 [CrossRef]
42. Lin, X.; Andrews, J.G.; Ghosh, A.; Ratasuk, R. An overview of 3GPP device-to-device proximity services. *IEEE Commun. Mag.* **2014**, *52*, 40–48. [CrossRef]
43. Wang, X.; Ning, Z.; Wang, L. Offloading in Internet of Vehicles: A Fog-Enabled Real-Time Traffic Management System. *IEEE Trans. Ind. Inform.* **2018**, *14*, 4568–4578. [CrossRef]
44. Shoham, Y.; Leyton-Brown, K. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*; Cambridge Univ. Press: Cambridge, UK, 2008.
45. Boyd, S.; Vandenberghe, L. *Convex Optimization*; Cambridge Univ. Press: Cambridge, UK, 2009.