Energy-Optimal and Delay-Bounded Computation Offloading in Mobile Edge Computing with Heterogeneous Clouds

Tianchu Zhao[1], Sheng Zhou[1,*], Linqi Song[2], Zhiyuan Jiang[3], Xueying Guo[4], Zhisheng Niu[1]

[1] Beijing National Research Center for Information Science and Technology, Department of Electronic Engineering, Tsinghua University, Beijing 100080, China
[2] Computer Science Department, City University of Hong Kong, Hongkong 999077, China
[3] School of Communication and Information Engineering, Shanghai University, Shanghai 200444, China
[4] Department of Computer Science, University of California, Davis, Davis 001313, USA
* The corresponding author: email: sheng.zhou@tsinghua.edu.cn

Abstract: By Mobile Edge Computing (MEC), computation-intensive tasks are offloaded from mobile devices to cloud servers, and thus the energy consumption of mobile devices can be notably reduced. In this paper, we study task offloading in multi-user MEC systems with heterogeneous clouds, including edge clouds and remote clouds. Tasks are forwarded from mobile devices to edge clouds via wireless channels, and they can be further forwarded to remote clouds via the Internet. Our objective is to minimize the total energy consumption of multiple mobile devices, subject to bounded-delay requirements of tasks. Based on dynamic programming, we propose an algorithm that minimizes the energy consumption, by jointly allocating bandwidth and computational resources to mobile devices. The algorithm is of pseudo-polynomial complexity. To further reduce the complexity, we propose an approximation algorithm with energy discretization, and its total energy consumption is proved to be within a bounded gap from the optimum. Simulation results show that, nearly 82.7% energy of mobile devices can be saved by task offloading compared with mobile device execution.

Keywords: mobile edge computing; heterogeneous clouds; energy saving; delay bounds; dynamic programming

I. INTRODUCTION

With the development of information and computing technologies, the proliferation of mobile devices is witnessed in recent years. However, mobile terminals are of constrained computational resources and battery life due to their limited physical sizes. Many computation-intensive applications, such as 3D gaming and virtual reality, pose great challenges to resource-constrained mobile devices [1]. Computation offloading provides a promising approach for the challenges, by which computation-intensive tasks are migrated from mobile devices to cloud servers [2].

Mobile Cloud Computing (MCC) is proposed as an architecture for handling computation offloading [3]. In MCC, mobile devices transmit data to remote clouds (e.g., Amazon Web Services) via wireless networks and the Internet, where the computation is then executed. But the data transmission over the Internet causes long access delay and large delay

Received: Aug. 23, 2019
Revised: Jan. 13, 2020
Editor: Dongliang Xie

Based on dynamic programming, we propose an algorithm that minimizes the energy consumption, by jointly allocating bandwidth and computational resources to mobile devices.

jitter, which can hardly meet the requirements of delay-sensitive applications [4]. Mobile Edge Computing (MEC) is an emerging technology to offload applications with stringent delay requirements [5]. In MEC systems (e.g., MEC proposed by ETSI [6]), cloud servers are deployed at the edge of networks (e.g., base stations). Thus, the data transmission delay between mobile devices and cloud servers is relatively small compared with MCC. Edge clouds are deployed in a distributed manner, and each edge cloud only serves a small number of users. A survey from Microsoft [7] indicates that the unit cost of a cloud increases when the total amount of computational resources decreases. Therefore, the resource multiplexing gain of edge clouds is not as large as remote clouds, and computational resources of edge clouds are relatively limited in comparison with remote clouds [8]. Edge clouds and remote clouds have different features in terms of offloading delay and computational resources. To better serve users with diverse requirements, heterogeneous clouds consisting of edge clouds and remote clouds should be jointly exploited [9].

In multi-user system, computation tasks generated from multiple mobile devices are offloaded to edge clouds and remote clouds simultaneously. Massive data is transmitted via wireless networks, leading to the usage of enormous bandwidth resources, which are precious. Also, edge clouds are of limited computational resources. To optimize the Quality of Service (QoS), the system should jointly allocate bandwidth and computational resources in an efficient way. But the joint resource allocation problem is in general NP-hard, and the optimal decision cannot be derived easily [14]. Despite the complexity of the problem, the system should make immediate decisions, because some tasks are of stringent delay requirements. These tasks cannot be offloaded, if the time to make the decision is longer than their delay requirements.

In this paper, to address the above issues, we focus on computation offloading in multi-user MEC system with heterogeneous clouds. Our Object is to minimize the energy consumption of mobile devices while satisfying different delay requirements of tasks, which is considered as a key metric to evaluate the QoS of MEC [13]. Compared with previous works [12]-[16], our work has several key differences: 1) We adopt different models, where stochastic wireless links and heterogeneous clouds are considered. 2) Our solutions are both of the optimality and low complexities (pseudo-polynomial time). The major contributions are summarized as follows:

Based on stochastic data transmission, we model task offloading to edge and remote clouds, respectively. Thereafter, we formulate the problem to minimize the total energy consumption of multiple mobile devices, as well as satisfying different delay bounds of tasks. By reduction from the knapsack problem, the energy-minimization problem is proved to be NP-hard. But the problem cannot be directly solved by existing algorithms for the knapsack problem, because the allocation of bandwidth and computational resources are coupled.

To solve the problem, we study the offloading of a single task, given the assigned bandwidth and computational resources. Based on the analytical results, we further propose an energy-optimal offloading algorithm for multiple tasks. The algorithm compares different task schedules and resource allocations by dynamic programming, and it is proved to minimize the total energy consumption of multiple mobile devices. The algorithm is of pseudo-polynomial complexity.

To further reduce the complexity, we propose an approximation offloading algorithm. By introducing a predefined parameter, the energy consumption of tasks is approximated as discrete values, and the algorithm minimizes the approximated energy consumption accordingly. The approximation ratio, defined as the ratio between the saved energy of the approximation offloading algorithm and that of the energy-optimal offloading algorithm, is proved to be within a bounded gap from the optimum.

The rest of the paper is organized as follows. In Section II, we review the related work. Section III introduces the system model. In Section IV, we formulate the problem to minimize the total energy consumption of mobile devices. In Section V and VI, we design energy-optimal offloading algorithm and approximation offloading algorithm respectively, so as to minimize the energy consumption. In Section VII, simulation results are shown to validate our analysis and the performance of the proposed algorithms. The paper is concluded in Section VIII.

II. Related Work

Computation offloading in MEC systems has lately been widely studied [10], and the energy saving of mobile devices attracts significant attentions [11].

To realize energy-efficient computation offloading, the joint allocation of computational and bandwidth resources is a key problem in multi-user MEC systems. In [12]-[14], the overall energy consumption of multiple mobile devices is minimized, by the joint resource allocation. The energy-minimization problem is NP-hard. Due to the complexity of the problem, these works derive sub-optimal solutions to the problem. The optimal solutions are proposed in [15] and [16] by Branch-and-Bound algorithms. However, the running time of their algorithms increases exponentially with the number of users, which can hardly be applied in real systems. In [17] and [18], the authors study the tradeoff between the mean energy consumption and the mean offloading delay of tasks. [17] optimizes the tradeoff by Lyapunov-based algorithm, and [18] relies on queuing analysis. Compared with the minimization of energy consumption, the tradeoff represents the weighted requirements on energy and delay. To provide massive connectivity (e.g., Internet of Things) in MEC, non-orthogonal multiple access (NOMA) is applied to MEC systems [19]. [20]-[23] focus on the energy minimization under delay constraints. [24] optimizes on a weighted parameter which linearly combines the offloading delay and energy consumption.

The architecture of MEC can be further extended, so as to serve users in different scenarios. In rural areas, the edge clouds are of low-density. To increase the coverage of each edge cloud, UAV (Unmanned Aerial Vehicle) assisted MEC is proposed, where communication relays are deployed on UAVs [25] [41]. In [26] and [27], the papers study MEC with UAV assisted networks. [26] minimizes the total energy consumption of mobile devices, while satisfying the QoS requirements of users. [27] minimizes the sum power consumption of both mobile devices and UAVs, under the constraints of delay and coverage.

For users with high-mobility, the activity range of a single user can be much larger than the coverage of a single edge cloud. To address the challenge, computation is migrated between multiple edge clouds, according to the movement of users [28]. In [29] and [30], the offloading delay is minimized under the constraint of energy consumption, by proposed Lyapunov algorithms. [29] focuses on single-user scenarios, and [30] focuses on multi-user scenarios.

To summarize, a great number of papers study the energy-efficient computation offloading in MEC systems, which focus on different objectives in different scenarios. These related works [12]-[30] are summarized in Table I.

III. System Model

We adopt the MEC architecture proposed by ETSI [7]. As shown in figure 1, multiple mobile devices offload tasks to edge and remote clouds. The mobile devices firstly send their tasks to the base station via the wireless network. Thereafter, the base station can forward tasks to the edge cloud directly, and it can also forward tasks to remote clouds via the Internet. In either the edge cloud or remote clouds, virtual machines are assigned to compute the offloaded tasks, and each virtual machine executes one task only.

3.1 Task model

We consider that the MEC system serves mobile users by scheduling intervals. In one scheduling interval, the system collects tasks from multiple mobile users, and makes decisions on task offloading for these current tasks. Thereafter, in the next interval, the system will repeat the same process for new mobile users. Without loss of generality, we focus on the energy-optimal offloading decisions for tasks in a specific interval.

By task partitioning, a task can be partitioned into multiple sub-tasks, and each sub-task is offloaded or executed in mobile devices [31]. For example, the face recognition can be partitioned into 4 sub-tasks [32], which are user interface, finding match, recognizer initialization and face detection respectively. In our model, we refer "task" to the smallest computational unit to be executed, which cannot be further partitioned.

Table I. *Comparison of related works.*

Work	System	Objective	Solutions
[12] [13] [14]	MEC with FDMA network	Energy minimization with delay constraint	Sub-optimal resource allocation algorithms of low-complexity
[15] [16]	MEC with FDMA network	Energy minimization with delay constraint	Optimal resource allocation algorithms of high-complexity
[17] [18]	MEC with cellular and Wifi	Energy-delay tradeoff	Lyapunov algorithm [17] and queuing analysis [18]
[20]-[23]	MEC with NOMA network	Energy minimization with delay constraint	Heuristic algorithms[20][21] Optimal solution[22][23]
[24]	MEC with NOMA network	Weighted energy and delay	Optimal solution
[26] [27]	MEC with UAV assisted network	Energy minimization with QoS constraint	Theoretical analysis
[29] [30]	Multiple edge clouds for task migration	Offloading delay with energy constraints	Lyapunov algorithms

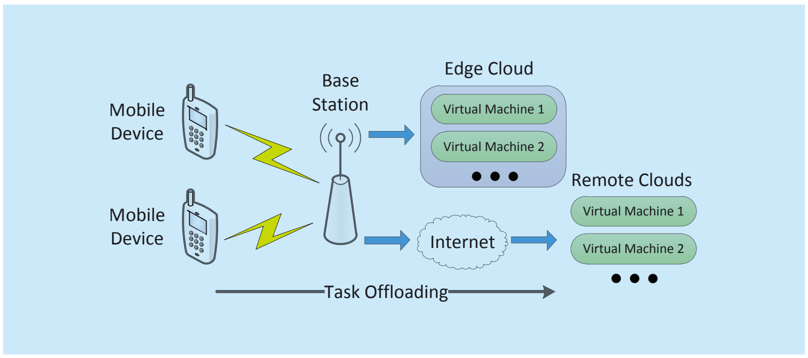


Fig. 1. *Task offloading with heterogeneous clouds.*

In one scheduling interval, assume that totally N tasks are generated from mobile devices, which are denoted by the set $\mathcal{N} = \{1,...,N\}$. Considering the delay requirements of tasks, T_n is referred to as the delay bound of the nth task, where $n \in \mathcal{N}$. To offload one task, the total processing delay, including the data transmission delay and the cloud execution delay, should not be larger than the delay bound. Otherwise, this task fails.

1) Task Offloading to Cloud Servers

If one task is offloaded to the edge cloud, it is firstly transmitted to the base station via the wireless network. Thereafter, it is executed in the edge cloud. We assume that the input data size of a task is much larger than the output data size (e.g., face recognition and vehicle plate detection), and mobile devices consume much more energy to upload data than download data [12][33][34]. Accordingly, we focus on the uploading of input data. For the nth task, let $D_{\mathrm{w},n}$ denote the wireless transmission delay, and let $D_{\mathrm{E},n}$ denote the task execution delay in the edge cloud. Thus, the constraint of total processing delay is $D_{\mathrm{w},n}+ D_{\mathrm{E},n} \le T_n$.

If one task is offloaded to remote clouds, its data is transmitted to the cloud via the wireless network and the Internet, and it is then executed. For the nth task, let $D_{\mathrm{R},n}$ denote the sum of the round-trip delay over the Internet and the task execution delay in remote clouds. The constraint of total processing delay $D_{\mathrm{W},n}+ D_{\mathrm{R},n} \le T_n$.

2) Task Execution in Mobile Devices

If the nth task is executed in the mobile device, the total processing delay is the execution delay. According to the study on mobile computing [1][14], the CPU cycles needed to execute a specific can be acquired. Thus, we assume that the total CPU cycles needed to execute the nth task is X_n. The mobile device can set its CPU frequency U_n, so that the delay bound is satisfied by $X_n/U_n \le T_n$.

3.2 Wireless transmission model

We consider that the wireless network adopts OFDMA as the multiple access scheme (e.g., LTE systems). In OFDMA networks, bandwidth resources are divided into orthogonal bandwidth resources, and they are allocated to mobile devices in bandwidth resource blocks. Assume that the total number of bandwidth resource blocks is W, and the bandwidth of each bandwidth resource block is ΔB. Thus, the total bandwidth resources are $B_{\text{Max}}=W\Delta B$. For the nth task, let $B_n=w_n\Delta B$ denote the allocated bandwidth resources, where w_n indicates the allocated bandwidth resource blocks. The constraint of bandwidth resources is given by

$$\sum\nolimits_{n=1}^{N} B_n \le B_{\text{Max}}. \tag{1}$$

For the nth task, let L_n denote the input data size. The input data is transmitted from the mobile device to the base station via the wireless channel. Let γ_n denote the SNR (signal-to-noise ratio) of the received wireless signal in the base station, which is given by

$$\gamma_n = \frac{|h_n|^2 p_n}{\sigma^2}, \tag{2}$$

where $|h_n|^2$ indicates the channel gain, p_n indicates the transmit power per unit bandwidth, and σ^2 indicates the density of Gaussian noise. Because the wireless channel is time-varying, h_n^2 is a random variable, and so does γ_n. The system can measure the mean value of the channel gain and SNR, which are denoted by H_n=E(h_n^2) and Υ_n= E(γ_n).

We assume that the distribution of h_n^2 is known to the system. Accordingly, for the nth task, the distribution of the wireless transmission delay is derived as follow.

Lemma 1 *The distribution of the wireless transmission delay is*

$$\mathrm{P}\left\{t \le D_{\mathrm{W},n}\right\} = 1 - \mathrm{e}^{-B_n \omega_n D_{\mathrm{W},n}}, \tag{3}$$

where

$$\omega_n = \max_{\{x\}} \frac{\mathrm{P}\left\{|h_n|^2 \ge \frac{H_n}{\gamma_n}\left(2^x - 1\right)\right\}}{L_n / x}. \tag{4}$$

Proof: See Appendix A.

In Lemma 1, it is shown that the distribution of wireless transmission delay follows exponential distribution, and $B_n\omega_n$ is the exponential factor. ω_n can be acquired in advance, because L_n, γ_n, and the distribution of h_n^2 are known to the system. Accordingly, the distribution of wireless transmission delay is determined by the allocated bandwidth resources B_n.

3.3 Cloud execution model

1) Edge Cloud Execution

Assume that the amount of computational resources in the edge cloud is F_{Max}, which indicates totally F_{Max} CPU cycles/s can be allocated to tasks. Let one unit of computational resources be ΔF, which denotes the smallest amount of computational resources for allocation. The number of total units is $C= F_{\text{Max}}/\Delta F$, and it is considered to be the scale of the edge cloud.

If the nth task is offloaded to the edge cloud, a virtual machine is assigned to execute the task. Let $F_n = c_n\Delta F$ denote the allocated computational resources, where c_n indicates the units of computational resources. We assume that F_n does not change during the offloading process. For all tasks offloaded to the edge cloud, the constraint of computational resources is given by

$$\sum\nolimits_{n=1}^{N} F_n \le F_{\text{Max}}. \tag{5}$$

As the CPU cycles needed to execute the nth task is X_n the edge cloud execution delay is $D_{\mathrm{E},n}= X_n/F_n$.

Table II. *Important notations.*

Symbol	Meaning
N	The total number of generated tasks.
T_n	The delay bound of the nth task.
X_n	The CPU cycles to execute the nth task.
γ_n	SNR of the received wireless signal of the nth task.
H_n	The mean value of the channel gain for the nth task.
σ^2	The density of Gaussian noise.
B_{Max}	The total bandwidth resources of the wireless network.
ΔB	The bandwidth resources of one resource block.
W	The total number of resource blocks.
F_{Max}	The total computational resources in the edge cloud.
ΔF	One unit of computational resources in the edge cloud.
C	The scale of the edge cloud.

2) Remote Cloud Execution

If the nth task is offloaded to remote clouds, the input data is transmitted to the cloud over the Internet, and the task is then executed. Because the dynamic Internet result in the randomness of data transmission delay, $D_{R,n}$ is a random variable.

According to [35], we assume that the round-trip delay over the Internet follows a given distribution, which is referred to as $P_R\{t \le T\}$. In remote clouds, a virtual machine is assigned to the nth task, and the allocated computational resources are assumed to be $F_{R,n}$. Because remote clouds are of abundant computational resources, $F_{R,n}$ is considered as a given parameter. Thus, the distribution of the remote cloud execution delay is $P_R\{t \le D_{R,n} - X_n/F_{R,n}\}$.

3.4 Success probability of tasks

Because $D_{W,n}$ and $D_{R,n}$ are random variables, the delay requirements of tasks are guaranteed by probability. We refer the probability that a task is successfully processed within its delay bound as the success probability $P_{S,n}$.

If the nth task is executed in the edge cloud, the success probability is $P_{S,n}=P(D_{W,n}+ D_{E,n} \le T_n)$. If the nth task is executed in remote clouds, $P_{S,n}=P(D_{W,n}+ D_{R,n} \le T_n)$.

Each mobile user has the requirement on the success probability, and the system should meet their requirements [8]. For the nth task, let $1-\varepsilon_n$ denote the minimum threshold of success probability, and the inequation $P_{S,n} \ge 1-\varepsilon_n$ holds for all offloaded tasks.

Thus, if a task is offloaded to the edge cloud, it requires sufficient bandwidth resources B_n and computational resources of the edge cloud F_n, so that the success probability is larger than the threshold. If a task is offloaded to remote clouds, it only requires sufficient bandwidth resources B_n, but it does not need the computational resources in the edge cloud.

3.5 Energy consumption of tasks

If the nth task is executed locally in the mobile device, the energy is consumed to run X_n CPU cycles. According to [34], the energy consumption to run one CPU cycle is ξU_n^2, in which U_n indicates the CPU frequency and ξ is a parameter of effective capacitance. To minimize the energy consumption of the nth task, the CPU frequency U_n is set as X_n/T_n, and the energy consumption of local execution is

$$E_{\mathrm{Loc},n} = X_n \xi \left(\frac{x_n}{T_n}\right)^2. \tag{6}$$

If the nth task is offloaded to the MEC system, the energy is consumed to transmit L_n bits of data. The total transmit power of the mobile device is $P_n=p_nB_n$, where p_n indicates the transmit power per unit bandwidth and B_n is the total bandwidth. According to the distribution of wireless transmission delay in Lemma 1, the mean energy consumption to transmit the input data is

$$E_{\mathrm{Off},n} = \mathrm{E}\left(D_{W,n}P_n\right) = \frac{\gamma_n \sigma^2}{H_n \omega_n}. \tag{7}$$

IV. Problem Formulation

If In this section, we formulate an optimization problem to minimize the energy consumption of multiple mobile devices. Each offloaded task requires sufficient bandwidth and computational resources so as to meet its delay requirement. However, as the bandwidth resources of the network and computational resources of the edge cloud are limited, only some of the tasks can be offloaded, whereas the rest of them have to be executed locally in the mobile devices. The system should make offloading decisions for these tasks: 1) How to schedule tasks to edge and remote clouds? 2) How to allocate the bandwidth and computational resources to the offloaded tasks? The two decisions are referred to as the task schedule and resource allocation, respectively.

Let y_n and z_n denote the schedule decision of the nth task. There are totally 3 different decisions for the task: 1) If the task is executed in the edge cloud, y_n=1 and z_n=0. 2) If the task is executed in remote clouds, y_n=0 and z_n=1. 3) Otherwise, the task is executed in the mobile device and thus y_n=0, z_n=0.

If the nth task is executed in the mobile device, the energy is consumed by the local execution. If the *n*th task is offloaded, the energy is consumed to transmit the data, instead of executing the task. Thus, the energy consumption of the *n*th task is

$$\left(1-y_n-z_n\right)E_{\mathrm{Loc},n}+\left(y_n+z_n\right)E_{\mathrm{Off},n}. \quad (8)$$

To minimize the total energy consumption of all tasks, the optimization problem is formulated as Problem P1.

$$\min_{\{y_n,z_n,B_n,F_n\}} \sum_{n=1}^{N}\left(1-y_n-z_n\right)E_{\mathrm{Loc},n} + \left(y_n+z_n\right)E_{\mathrm{Off},n} \quad (\mathrm{P1})$$

$$s.t. \quad \sum_{n=1}^{N} B_n \le B_{\mathrm{Max}} \quad (\mathrm{C1.1})$$

$$\sum_{n=1}^{N} F_n \le F_{\mathrm{Max}} \quad (\mathrm{C1.2})$$

$$\mathrm{P}\left\{D_{W,n}+D_{E,n}\le T_n\right\}\ge 1-\varepsilon_n,\ if\ \ y_n=1 \quad (\mathrm{C1.3})$$

$$\mathrm{P}\left\{D_{W,n}+D_{R,n}\le T_n\right\}\ge 1-\varepsilon_n,\ if\ \ z_n=1 \quad (\mathrm{C1.4})$$

$$y_n, z_n \in \{0,1\}, \forall n \quad (\mathrm{C1.5})$$

$$y_n+z_n \le 1, \forall n \quad (\mathrm{C1.6})$$

where (C1.1) indicates the limitation of bandwidth resources. (C1.2) indicates the limitation of computational resources of the edge cloud. Because F_n denotes the allocated computation resources in the edge cloud, $F_n=0$ if the *n*th task is offloaded to remote clouds. (C1.3) and (C1.4) are the constraints of delay requirements. (C1.5) and (C1.6) indicate that each task should be executed in the mobile device, or in the edge cloud, or in remote clouds.

We have the following proposition for the complexity of solving P1.

Proposition 1 *Problem P1 is NP-hard.*

Proof: See Appendix B.

By reduction from the knapsack problem, Problem P1 is proved to be NP-hard. However, Problem P1 cannot be solved by existing algorithms for the knapsack problem. In the knapsack problem, the required resources for each item are fixed; in Problem P1, the resource allocation of B_n and F_n can be changed and they are coupled. Thus, it is not easy to solve Problem P1. Nevertheless, inspired by the knapsack problem, we rely on dynamic programming to solve Problem P1 in the following sections. The proposed algorithms compare different task schedules as well as different resource allocations by dynamic programming. Accordingly, the energy-optimal task schedule and resource allocation are derived if they minimize the energy consumption of mobile devices. These algorithms are of relatively low complexities (pseudo-polynomial time).

Remark 1 The knapsack problem can be solved by a dynamic programming based algorithm [36], and the algorithm is of pseudo-polynomial complexity. In a knapsack problem, let N_k denote the number of items, and let S_k denote the size of the packet. The algorithm complexity is $O(N_kS_k)$. It does not contradict the fact that the knapsack problem is NP-hard. Because the length of an input is $log(S_k)$, the complexity is actually exponential to the length of the input. But the algorithm complexity is polynomial to the number of items N_k, and the running time is still acceptable even if N_k is large. The NP-hard problems of pseudo-polynomial complexit***y***, such as the knapsack problem, are defined as weakly NP-hard problems.

V. Energy-Optimal Task Schedule and Resource Allocation

In this section, the optimal solution to Problem P1 is derived. Firstly, we study the schedule decision of a single task when the allocated bandwidth and computational resources are given. Based on the analysis, the energy-optimal task schedule and resource allocation for multiple tasks are derived base on a dynamic programming based algorithm. Finally, the complexity of the algorithm is discussed, which is of pseudo-polynomial time.

5.1 Offloading of a single task

Taking the *n*th task as an example, we study the schedule decision of a single task.

If the *n*th task is offloaded to the edge cloud, the constraint of its delay requirement is shown in Section II. The allocated bandwidth resources B_n should be larger than

$$\alpha_n = \frac{1}{\omega_n T_n}\ln\frac{1}{\varepsilon_n}. \quad (9)$$

Otherwise, the delay requirement is not met even if the allocated computational resources are infinite. As B_n is integral multiples of ΔB, the minimum amount of required bandwidth resources is

$$B_{\mathrm{E},n} = \left\lceil \frac{\alpha_n}{\Delta B} \right\rceil \Delta B. \tag{10}$$

where $\lceil x \rceil$ indicates the smallest integer that is not smaller than x. The computational resources required by the nth task is a function of B_n, which are denoted by $\mathrm{F}_{\mathrm{E},n}(B_n)$. Because the allocated computational resources in the edge cloud should be integral multiples of ΔF,

$$F_{\mathrm{E},n}\left(B_n\right) = \left\lceil \frac{x_n}{\Delta F T_n \left(1-\alpha_n / B_n\right)} \right\rceil \Delta F. \tag{11}$$

If the nth task is offloaded to remote clouds, the constraint of its delay requirement is

$$\mathrm{P}\left\{D_{\mathrm{W},n} + D_{\mathrm{R},n} \le T_n\right\} \ge 1-\varepsilon_n. \tag{12}$$

As shown in Section II, $\mathrm{P}\{t \le D_{\mathrm{R},n}\}$ follows an distribution. $\mathrm{P}\{t \le D_{\mathrm{W},n}\}$ is derived in Lemma 1, which is determined by the allocated bandwidth resources B_n. Thus, $\mathrm{P}\{D_{\mathrm{W},n}+D_{\mathrm{R},n} \le T_n\}$ is determined by B_n. Let $B_{\mathrm{R},n}$ denotes the minimum amount of bandwidth resources so that (12) holds. For tasks with stringent delay bounds, the constraint on success probability never holds, even if the allocated bandwidth resources are infinite. In the condition, let $B_{\mathrm{R},n} \to \infty$, which indicates that the task cannot be offloaded to remote clouds. Because $D_{\mathrm{R},n} > 0$, $B_{\mathrm{R},n} > B_{\mathrm{E},n}$ always holds.

Based on the analysis, there are totally three different kinds of schedule decisions for the nth task.

- If $B_n \ge B_{\mathrm{R},n}$, the task should be offloaded to remote clouds, and no computational resource in the edge cloud is needed.
- If $B_{\mathrm{E},n} \le B_n < B_{\mathrm{R},n}$ and $F_n \ge \mathrm{F}_{\mathrm{E},n}(B_n)$, the task are allocated with sufficient bandwidth and computational resources, and it should be offloaded to the edge cloud.
- Otherwise, the task should be executed in the mobile device.

5.2 Energy-optimal offloading algorithm

A task schedule decides how tasks are offloaded to the edge and remote clouds. We define a task schedule as $\mathcal{P}$. Let S_E denote the set of tasks offloaded to the edge cloud, and let S_R denote the set of tasks offloaded to remote clouds. The tasks in the set $\mathcal{N} \setminus \left\{S_\mathrm{E} \bigcup S_\mathrm{R}\right\}$ are executed in mobile devices. Accordingly, the task schedule $\mathcal{P}$ is determined by S_E and S_R. The total energy consumption of $\mathcal{P}$ is

$$\sum\nolimits_{n \in S_\mathrm{E} \cup S_\mathrm{R}} E_{\mathrm{Off},n} + \sum\nolimits_{n \in \mathcal{N} \setminus \{S_\mathrm{E} \cup S_\mathrm{R}\}} E_{\mathrm{Loc},n}. \tag{13}$$

To minimize the total energy consumption of all tasks, the system should make a decision on the task schedule. Meanwhile, the system should also allocate bandwidth and computational resources to the offloaded tasks. We propose an algorithm to jointly get the energy-optimal task schedule and resource allocation, which is referred to as the energy-optimal offloading algorithm. The algorithm compares different tasks schedules and resource allocations by dynamic programming, and the optimal task schedule and resource allocation are derived if they minimize the total energy consumption.

Firstly, we define the value function of task schedules. In the algorithm, the problem is decomposed into N stages, and k denotes different stages. In the kth stage, the algorithm considers the tasks in the set $\mathcal{K} = \{1,...,k\}$. After N stages, all of the N tasks are jointly considered. The value function is defined as $V(k,B,F)$, which is the objective to be optimized. In the k stage, if the bandwidth resources of the network and the computational resources of the edge cloud are B and F respectively, the minimum energy consumption of the tasks is $V(k,B,F)$,. The corresponding task schedule is defined as $\mathcal{P}(k,B,F)$ in which tasks in the set $S_\mathrm{E}(k,B,F)$ are offloaded to the edge cloud and tasks in the set $S_\mathrm{R}(k,B,F)$ are offloaded to remote clouds. In other words, the energy consumption of the task schedule $\mathcal{P}(k,B,F)$ is $V(k,B,F)$.

Secondly, we define the representation of the resource allocation for a specific task schedule. Based on the task schedule $\mathcal{P}(k,B,F)$, the corresponding resource allocation is $R(k,B,F)$. Each element in the set $R(k,B,F)$ is indexed as (k,B_k,F_k) or (k,B_k). If the kth task is offloaded to the edge cloud, the corresponding element is indexed as (k,B_k,F_k), which denotes the bandwidth resources B_k and computational resources F_k allocated to the kth task. If the kth task is offloaded to remote clouds, the corresponding element is indexed as (k,B_k), which indicates that bandwidth resources B_k are allocated to the kth task.

The algorithm compares different task schedules and resource allocations stage by stage. As the algorithm considers a new task in each stage, it decides if to offload the new task by making comparison between new task schedules and the task schedules in the previous stage. If it consumes less energy by offloading the new task, the algorithm will update the task schedule accordingly. In each stage, $\forall B$ that $0 \le B \le B_{\text{Max}}$ and $\forall F$ that $0 \le F \le F_{\text{Max}}$, the value function $V(k,B,F)$ is updated. Given the total bandwidth resources B_{Max} and computational resources F_{Max}, the algorithm minimizes the energy consumption of all tasks by $V(N,B_{\text{Max}},F_{\text{Max}})$. The corresponding task schedule is $\mathcal{P}(N,B_{\text{Max}},F_{\text{Max}})$, and resource allocation is $R(N,B_{\text{Max}},F_{\text{Max}})$.

$\forall k$, in the kth stage, the algorithm begins to consider the kth task. Let B_k denote the bandwidth resources allocated to the task. Let $F_k = \mathrm{F}_{\mathrm{E},k}(B_k)$ denote the required computational resources in the edge cloud, so as to meet the constraint of success probability. Based on the allocated bandwidth and computational resources, the offloading decision of the kth task is:

- If $B_k \ge B_{\mathrm{R},k}$, the task can be offloaded to remote clouds. In the case, if
$$\begin{aligned} &V(k-1,B,F)+E_{\text{Loc},k} \\ &> V\left(k-1,B-B_k,F\right)+E_{\text{Off},k} \end{aligned}, \quad (14)$$
the task schedule of $S_{\mathrm{E}}(k-1,B-B_k,F)$ and $S_{\mathrm{R}}\left(k-1,B-B_k,F\right)\cup\{k\}$ consumes less energy than the task schedule of $S_{\mathrm{E}}(k-1,B,F)$ and $S_{\mathrm{R}}(k-1,B,F)$.
- If $B_{\mathrm{E},k} \le B_k < B_{\mathrm{R},n}$ and $F_k \ge \mathrm{F}_{\mathrm{E},k}(B_k)$, the task can be offloaded to the edge cloud only. In the case, if
$$\begin{aligned} &V(k-1,B,F)+E_{\text{Loc},k} \\ &> V\left(k-1,B-B_k,F-F_k\right)+E_{\text{Off},k} \end{aligned}, \quad (15)$$
the task schedule of $S_{\mathrm{E}}(k-1,B-B_k,F-F_k)\cup\{k\}$ and $S_{\mathrm{R}}\left(k-1,B-B_k,F-F_k\right)$ consumes less energy than the task schedule of $S_{\mathrm{E}}(k-1,B,F)$ and $S_{\mathrm{R}}(k-1,B,F)$.
- Otherwise, the task cannot be offloaded to either the edge cloud or remote clouds.

By comparing all the possible B_k that $B_{\mathrm{E},k} \le B_k \le B$, the function $V(k,B,F)$ is updated by the minimum one, which indicates the least amount of energy consumption with bandwidth resources B and computational resources F. Furthermore, $\forall B$ that $0 \le B \le B_{\text{Max}}$ and $\forall F$ that $0 \le F \le F_{\text{Max}}$, the value function $V(k,B,F)$ is updated in each stage. The transition function of the dynamic programming based algorithm is:
$$\begin{aligned} V(k,B,F) = \min\Big(&\left\{V(k-1,B,F)+E_{\text{Loc,k}}\right\} \\ &\cup\left\{V\left(k-1,B-B_k,F-F_k\right)+E_{\text{off},k} \mid \forall B_k \in \left[B_{\mathrm{E},k},B_{\mathrm{R},k}\right)\right\} \\ &\cup\left\{V\left(k-1,B-B_k,F\right)+E_{\text{Off},k} \mid \forall B_k \in \left[B_{\mathrm{R},k},B\right]\right\}\Big) \end{aligned} . \quad (16)$$

According to the update of $V\left(k,B,F\right)$, the task schedule $\mathcal{P}(k,B,F)$ and resource allocation $R(k,B,F)$ are updated. If the kth task is offloaded to the edge cloud with bandwidth resources B_k and computational resources F_k,
$$\begin{cases} S_{\mathrm{E}}(k,B,F) = S_{\mathrm{E}}\left(k-1,B-B_k,F-F_k\right)\cup\{k\} \\ S_{\mathrm{R}}(k,B,F) = S_{\mathrm{R}}\left(k-1,B-B_k,F-F_k\right) \\ R(k,B,F) = R\left(k-1,B-B_k,F-F_k\right) \\ \qquad\qquad \cup\left\{\left(k,B_k,F_k\right)\right\} \end{cases} . \quad (17)$$

If the kth task is offloaded to remote clouds with bandwidth resources B_k,
$$\begin{cases} S_{\mathrm{E}}(k,B,F) = S_{\mathrm{E}}\left(k-1,B-B_k,F\right) \\ S_{\mathrm{R}}(k,B,F) = S_{\mathrm{R}}\left(k-1,B-B_k,F\right)\cup\{k\} \\ R(k,B,F) = R\left(k-1,B-B_k,F\right)\cup\left\{\left(k,B_k\right)\right\} \end{cases} . \quad (18)$$

In Algorithm 1, the energy-optimal off-

Algorithm 1. Energy-optimal offloading algorithm.

```
Input:  $\mathcal{N}$, $B_{\text{Max}}$, $F_{\text{Max}}$, $\Delta B$, $\Delta F$, $[X_1,\ldots,X_N]$, $[B_{\text{E},1},\ldots,B_{\text{E},N}]$, $[B_{\text{R},1},\ldots,B_{\text{R},N}]$, $[E_{\text{Off},1},\ldots,$
        $E_{\text{Off},N}]$, $[E_{\text{Loc},1},\ldots,E_{\text{Loc},N}]$
Output: $V(N,B_{\text{Max}},F_{\text{Max}})$, $S_{\text{E}}(N,B_{\text{Max}},F_{\text{Max}})$, $S_{\text{R}}(N,B_{\text{Max}},F_{\text{Max}})$,
        $R(N,B_{\text{Max}},F_{\text{Max}})$

for $k=1$; $k\le N$; $k=k+1$
for $B=0$; $B\le B_{\text{Max}}$; $B=B+\Delta B$
    for $F=0$; $F\le F_{\text{Max}}$; $F=F+\Delta F$
        $V(k,B,F)\leftarrow V(k-1,B,F)+E_{\text{Off},k}$
        for each $B_k\in[B_{\text{E},k}, B]$
            $F_k\leftarrow F_{\text{E},k}(B_k)$
            if $B_k\ge B_{\text{R},k}$
                $V_t\leftarrow V(k-1,B-B_k,F-F_k)+E_{\text{Off},k}$
            else if $B_k\ge B_{\text{E},k}$
                $V_t\leftarrow V(k-1,B-B_k,F-F_k)+E_{\text{Off},k}$
            end if
            if $V(k,B,F)> V_t$
                $V(k,B,F)\leftarrow V_t$
                Update $S_{\text{E}}(k,B,F)$, $S_{\text{R}}(k,B,F)$, $R(k,B,F)$
            end if
        end for
    end for
    end for
end for
```

loading algorithm is stated in details. The value function $V(N,B_{\text{Max}},F_{\text{Max}})$ denotes the minimum energy consumption. Furthermore, the energy-optimal task schedule is $\mathcal{P}(N,B_{\text{Max}},F_{\text{Max}})$, in which tasks in the set $S_E(N,B_{\text{Max}},F_{\text{Max}})$ are offloaded to the edge cloud and tasks in the set $S_R(N,B_{\text{Max}},F_{\text{Max}})$ are offloaded to remote clouds. The corresponding resource allocation is $R(N,B_{\text{Max}},F_{\text{Max}})$.

A task schedule $\mathcal{P}$ of S_{E} and S_{R} is defined to be feasible if there exists a resource allocation that all tasks in S_{E} and S_{R} can be offloaded, as stated by the following theorem.

Theorem 1 *For any feasible task schedule* $\mathcal{P}$ of S_{E} and S_{R},

$$\begin{aligned}&\sum\nolimits_{n\in S_{\text{Opt}}} E_{\text{Off},n}+\sum\nolimits_{n\in\mathcal{N}\setminus S_{\text{Opt}}} E_{\text{Loc},n}\\ &\le \Sigma_{n\in S_{\text{E}}\cup S_{\text{R}}} E_{\text{Off},n}+\sum\nolimits_{n\in\mathcal{N}\setminus\{S_{\text{E}}\cup S_{\text{R}}\}} E_{\text{Loc},n}\end{aligned}, \quad (19)$$

where

$S_{\text{Opt}}=S_{\text{E}}(N,B_{\text{Max}},F_{\text{Max}})\cup S_{\text{R}}(N,B_{\text{Max}},F_{\text{Max}})$.

Proof: See Appendix C.

Theorem 1 shows that the energy consumption of any feasible task schedule is not smaller than the task schedule $\mathcal{P}(N,B_{\text{Max}},F_{\text{Max}})$. Thus, the output of Algorithm 1 is the energy-optimal task schedule and resource allocation, which minimize the total energy consumption of mobile devices.

Remark 2 The energy-optimal offloading algorithm and the algorithm for the knapsack problem [36] are both based on dynamic programming. The algorithm for the knapsack problem allocates fixed resources to different items, which cannot be utilized to solve Problem P1. The energy-optimal offloading algorithm studies how to schedule tasks to edge and remote clouds with different resource allocations. Furthermore, the algorithm jointly compares different tasks schedules and different resource allocations by dynamic programming, and the optimal task schedule and resource allocation are derived if they minimize the total energy consumption.

C. Algorithm Complexity

In the energy-optimal offloading algorithm, there are totally 4 loops. The running time of the loops are $O(N)$, $O(W)$, $O(C)$ and $O(W)$ respectively, and the complexity of the algorithm is $O(NCW^2)$. Here, W and C indicate the number of bandwidth resource blocks and the scale of the edge cloud, respectively. In the algorithm, the lengths of the inputs are $log(W)$ and $log(C)$, and the complexity is exponential to the lengths of the inputs. Also, the complexity is polynomial to the total number of tasks N. Thus, the algorithm is of pseudo-polynomial complexity. Because Problem P1 can be solved within pseudo-polynomial time, it is actually a weakly NP-hard problem.

Generally speaking, it takes long time to find the optimal solutions of NP-hard problems, due to their high complexities. But algorithms of pseudo-polynomial complexities provide ways to solve NP-hard problems within relatively short time. With the energy-optimal offloading algorithm, the running time to solve Problem P1 is polynomial to the total number of mobile devices. Accordingly, even if the number of mobile devices is large, the running time of the algorithm can still be acceptable.

However, the algorithm of pseudo-polynomial complexity is more complex compared with algorithms of polynomial time. Its complexity is not only related to the number of mobile devices, but also related to the number of bandwidth resource blocks and the scale of the edge cloud. In OFDMA systems, the number of bandwidth resource block is generally fixed, which influences the complexity weakly. But the scales of edge clouds are quite diverse, and they can be small [37] or large [14]. The MEC system can evaluate the running time according to its scale of the edge cloud. If the running time is too long to be acceptable, the energy-optimal offloading algorithm cannot be applied.

VI. Approximate Task Schedule and Resource Allocation

In this section, we design an approximation offloading algorithm to reduce the complexity compared with the energy-optimal offloading algorithm. The approximation offloading algorithm is proved to perform within a bounded gap compared to the optimum. Meanwhile, the algorithm complexity is of pseudo-polynomial time, and it is not related to the scale of the edge cloud.

6.1 Approximation offloading algorithm

For the nth task, the saved energy is $E_n = E_{\mathrm{Loc},n} - E_{\mathrm{Off},n}$, if it is offloaded to either the edge cloud or remote clouds. Firstly, we approximate E_n as discrete values. $\forall n$, let E_{Max} be the maximum value of E_n. We define the approximate parameter as ρ with $0<\rho\leq 1$, and the unit of saved energy is $\Delta E = \rho E_{\mathrm{Max}}/N$. Based of ΔE, the amount of saved energy E_n is approximated as $\tilde{E}_n$. If ρ gets smaller, this approximation can be more precise.

$$\tilde{E}_n = \left\lfloor \frac{E_n}{\Delta E} \right\rfloor \Delta E. \tag{20}$$

Here, $\lfloor x \rfloor$ indicates the largest integer that is not larger than x. Based on the energy discretization, we transform Problem P1 into Problem P2.In Problem P1, the goal is to minimize the total energy consumption of N tasks. In Problem P2, the objective is to maximize the approximated amount of saved energy. In fact, Problem P2 is the approximation of Problem P1 with the same constraints.

$$\min_{\{y_n, z_n, B_n, F_n\}} \sum_{n=1}^{N} \left(y_n + z_n \right) \widetilde{E}_n \tag{P2}$$

$$s.t. \ \text{(C1.1)}, \text{(C1.2)}, \text{(C1.3)}, \text{(C1.4)}, \text{(C1.5)}, \text{(C1.6)}$$

We propose an algorithm to solve Problem P2, which is referred to as the approximation offloading algorithm. The algorithm does not consider the limitation of the computational resources in the edge cloud. Instead, the algorithm calculates the required computational resources, so as to save different amounts of energy. Accordingly, given the total computational resources F_{Max} the maximum amount of energy that can be saved is obtained, and the corresponding task schedule and resource allocation are derived.

Task schedules and resource allocations are defined as follows. The problem is decomposed into N stages, and the tasks in the set $\mathcal{K} = \{1,...,k\}$ are considered in the kth stage. The value function is defined as $A(k,B,E)$. In the kth stage, if the bandwidth resources are B, $A(k,B,E)$ indicates the minimum amount of computational resources to save the energy E. The corresponding task schedule is defined as $\mathcal{P}_{\mathrm{A}}(k,B,E)$ in which tasks in the set $S_{\mathrm{AE}}(k,B,E)$ are offloaded to the edge cloud and tasks in the set $S_{\mathrm{AR}}(k,B,E)$ are offloaded to remote clouds. Accordingly, the resource allocation is $R_{\mathrm{A}}(k,B,E)$.

The algorithm compares different task schedules and resource allocations stage by stage. In each stage, the algorithm considers a new task, and it decides if to offload the new task by making comparison between new task schedules and the task schedules in the previous stage. If it utilizes less computational resources to save the same amount of energy, the algorithm will update the task schedule accordingly. There are totally N tasks, and the maximum saved energy of any task is E_{Max}. Thus, the total saved energy is not larger than NE_{Max}. In each stage, $\forall$B that $0 \le B \le B_{\text{Max}}$ and $\forall$E that $0 \le E \le NE_{\text{Max}}$, the value function $A(k,B,E)$ is updated if the energy E can be saved with less computational resources. The algorithm finds the maximum value E^* satisfying $A(N,B_{\text{Max}},E^*) \le F_{\text{Max}}$, which represents the maximum saved energy E^* with computational resources F_{Max}. The approximate task schedule $\mathcal{P}_{\text{A}}(N,B_{\text{Max}},E^*)$ is derived accordingly, and the corresponding resource allocation is given by $R_{\text{A}}(N,B_{\text{Max}},E^*)$.

$\forall k$, in the kth stage, the algorithm takes the kth task into consideration. Let B_k denote the bandwidth resources allocated to the task. Let $F_k = \text{F}_{\text{E},k}(B_k)$ denote the required computational resources in the edge cloud. Based on the allocated bandwidth resources, the offloading decision of the kth task is:

- If $B_k \ge B_{\text{R},k}$, the task can be offloaded to remote clouds. In the case, if
$$A(k-1,B,E) > A\left(k,B-B_k,E-\tilde{E}_k\right), \quad (21)$$
the task schedule of $S_{\text{AE}}(k,B-B_k,E-\tilde{E}_k)$ and $S_{\text{AR}}(k,B-B_k,E-\tilde{E}_k) \cup \{k\}$ requires less computational resources than the task schedule of $S_{\text{AE}}(k-1,B,E)$ and $S_{\text{AR}}(k-1,B,E)$, while the two schedules save the same amount of energy.
- If $B_{\text{E},k} \le B_k < B_{\text{R},k}$ and $F_k \ge \text{F}_{\text{E},k}(B_k)$, the task can be offloaded to the edge cloud only. In the case, if
$$A(k-1,B,E) > A\left(k,B-B_k,E-\tilde{E}_k\right) + F_k, \quad (22)$$
the task schedule of $S_{\text{AE}}(k,B-B_k,E-\tilde{E}_k) \cup \{k\}$ and $S_{\text{AR}}(k,B-B_k,E-\tilde{E}_k)$ requires less computational resources than the task schedule of $S_{\text{AE}}(k-1,B,E)$ and $S_{\text{AR}}(k-1,B,E)$, while the two schedules save the same amount of energy.
- Otherwise, the task cannot be offloaded to either the edge cloud or remote clouds.

The transition function of the dynamic programming based algorithm is:
$$\begin{aligned} A(k,B,E) = \min(&\{A(k-1,B,E)\} \\ &\cup \left\{A\left(k-1,B-B_k,E-\tilde{E}_k\right)+F_k \mid \forall B_k \in \left[B_{\text{E},k},B_{\text{R},k}\right)\right\} \\ &\cup \left\{A\left(k-1,B-B_k,E-\tilde{E}_k\right) \mid \forall B_k \in \left[B_{\text{R},k},B\right]\right\}. \end{aligned} \quad (23)$$

According to the update of $A(k,B,E)$, the task schedule $\mathcal{P}_{\text{A}}(k,B,E)$ and resource allocation $R_{\text{A}}(k,B,E)$ are updated. If the kth task is offloaded to the edge cloud with bandwidth resources B_k and computational resources F_k,
$$\begin{cases} S_{\text{AE}}(k,B,F) = S_{\text{AE}}\left(k-1,B-B_k,E-\tilde{E}_k\right) \cup \{k\} \\ S_{\text{AR}}(k,B,F) = S_{\text{AR}}\left(k-1,B-B_k,E-\tilde{E}_k\right) \\ R_{\text{A}}(k,B,F) = R_{\text{A}}\left(k-1,B-B_k,E-\tilde{E}_k\right) \\ \qquad \cup \left\{\left(k,B_k,F_k\right)\right\} \end{cases} \quad (24)$$

If the kth task is offloaded to remote clouds with bandwidth resources B_k,
$$\begin{cases} S_{\text{AE}}(k,B,F) = S_{\text{AE}}\left(k-1,B-B_k,E-\tilde{E}_k\right) \\ S_{\text{AR}}(k,B,F) = S_{\text{AR}}\left(k-1,B-B_k,E-\tilde{E}_k\right) \cup \{k\} \\ R_{\text{A}}(k,B,F) = R_{\text{A}}\left(k-1,B-B_k,E-\tilde{E}_k\right) \\ \qquad \cup \left\{\left(k,B_k\right)\right\} \end{cases} \quad (25)$$

In Algorithm 2, the approximation offloading algorithm is stated in details. The output E^* denotes the maximum amount of saved energy by task offloading, and the approximate task schedule is of $S_{\text{AE}}(N,B_{\text{Max}},E^*)$ and $S_{\text{AR}}(N,B_{\text{Max}},E^*)$. $R_{\text{A}}(N,B_{\text{Max}},E^*)$ indicates the resource allocation for the offloaded tasks.

In the approximation offloading algorithm, task schedules are compared if they save the same amount of energy, and the optimal one is derived if it uses the least amount of computational resources. Because the required computational resources of task schedules are compared and optimized, the complexity of the algorithm is not related to the scale of the edge cloud C. However, the algorithm approximates the amount of saved energy as discrete values, which results in the inaccuracy for comparing different task schedules. For example, two task schedules save different amounts of energy indeed, but they are compared because of the approximation.

Theorem 2 *Given the approximate parameter ρ, compared with the energy-optimal offloading algorithm, the approximation ratio of the approximation offloading algorithm is bounded by*

$$\frac{\sum_{n\in S_{\text{App}}} E_n}{\sum_{n\in S_{\text{Opt}}} E_n} \geq 1-\rho, \tag{26}$$

where

$S_{\text{Opt}} = S_{\text{E}}\left(N, B_{\text{Max}}, F_{\text{Max}}\right) \cup S_{\text{R}}\left(N, B_{\text{Max}}, F_{\text{Max}}\right)$,

and

$S_{\text{App}} = S_{\text{AE}}\left(N, B_{\text{Max}}, E^*\right) \cup S_{\text{AR}}\left(N, B_{\text{Max}}, E^*\right)$.

Proof: See Appendix D.

In Theorem 2, the amount of energy that the approximation offloading algorithm saves is proved to be within a bounded gap to the optimum. The task schedule of the energy-optimal offloading algorithm is of $S_{\text{E}}\left(N, B_{\text{Max}}, F_{\text{Max}}\right)$ and $S_{\text{R}}\left(N, B_{\text{Max}}, F_{\text{Max}}\right)$, and the approximate task schedule is of $S_{\text{AE}}\left(N, B_{\text{Max}}, E^*\right)$ and $S_{\text{AR}}\left(N, B_{\text{Max}}, E^*\right)$. The approximation ratio of the approximation offloading algorithm is not smaller than 1-ρ. In other words, if the amount of saved energy by the energy-optimal task schedule is $\sum_{n\in S_{\text{Opt}}} E_n$, the energy that the approximate task schedule saves is at least $(1-\rho)\sum_{n\in S_{\text{Opt}}} E_n$.

6.2 Algorithm complexity

In the approximation offloading algorithm, there are totally 4 loops. The running time of the loops are $O(N)$, $O(W)$, $O(\lfloor N/\rho \rfloor N)$ and $O(W)$ respectively, where W indicates the number of bandwidth resource block. In the algorithm, the length of the input is $log(W)$, and the complexity is exponential to the length of the input. Also, the complexity is polynomial to the total number of tasks N. Thus, the al-

Algorithm 2. Approximation offloading algorithm.

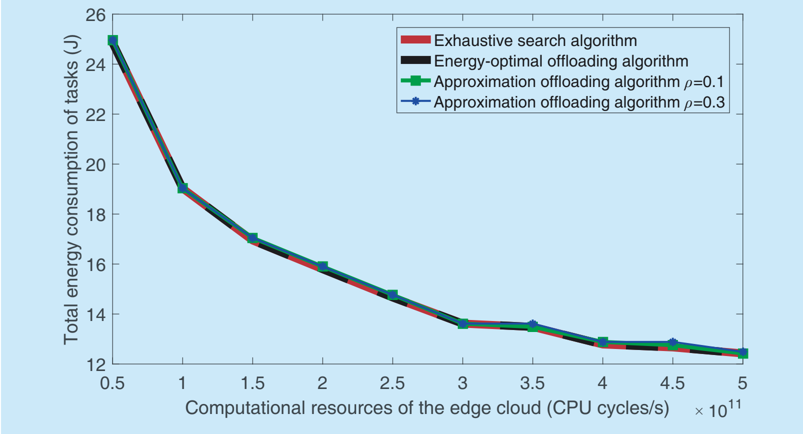
```
Input: $N$, $B_{\text{Max}}$, $F_{\text{Max}}$, $\Delta B$, $\Delta F$, $[X_1,\ldots,X_N]$, $[B_{\text{E},1},\ldots,B_{\text{E},N}]$, $[B_{\text{R},1},\ldots,B_{\text{R},N}]$, $[\tilde{E}_1,\ldots,\tilde{E}_N]$
Output: $E^*$, $S_{\text{AE}}\left(N,B_{\text{Max}},E^*\right)$, $S_{\text{AR}}\left(N,B_{\text{Max}},E^*\right)$, $R_{\text{A}}\left(N,B_{\text{Max}},E^*\right)$
        $E_{\text{Sum}} = \sum_{n=1}^{N}\tilde{E}_n$
for $k$=1; $k \le N$; $k$=$k$+1
for $B$=0; $B \le B_{\text{Max}}$; $B$=$B$+$\Delta B$
    for $E$=0; $E \le E_{\text{Sum}}$; $E$=$E$+$\Delta E$
      if $E-\tilde{E}_k$ can be saved by any task schedule
        for each $B_k \in [B_{\text{E},k}, B]$S
          $F_k \leftarrow F_{\text{E},k}(B_k)$
          if $B_k \ge B_{\text{R},k}$
            $A_t \leftarrow A\left(k-1, B-B_k, E-\tilde{E}_k\right)$
          else if $B_k \ge B_{\text{E},k}$
            $At \leftarrow A\left(k-1, B-B_k, E-\tilde{E}_k\right)+F_k$
          end if
          if $A(k,B,F) > A_t \vee E=\tilde{E}_k$
            $A(k,B,F) \leftarrow A_t$
            Update $S_{\text{AE}}(k,B,F)$, $S_{\text{AR}}(k,B,F)$, $R_{\text{A}}(k,B,F)$
          end if
        end for
      end if
  end for
  end for
end for
$E^* \leftarrow 0$
$E \leftarrow E_{\text{Sum}}$
while $E^*$=0 ^ $E \ge 0$
  if $A\left(N, B_{\text{Max}}, E\right) \le E$
    $E^* = E$
  end if
$E$=$E$-$\Delta E$
end while
```

Table III. *Simulation parameters.*

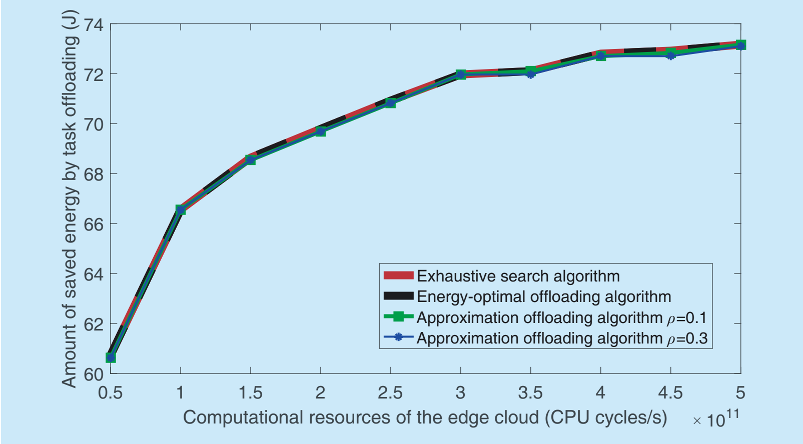
Parameters	Values
Up-link bandwidth B_{Max}	20MHz
Total number of bandwidth resource blocks W	100
Density of the Gaussian noise σ^2	10^{-12}
SNR of mobile devices γ_k	[5,30]
Mean round-trip delay over the Internet	150ms
Total computational resources in the edge cloud F_{Max}	$[10^{10}, 10^{12}]$ CPU cycle/s
One unit of computational resources in the edge cloud ΔF	$3*10^7$ CPU cycle/s
Input data size L_n	$[10^3, 10^6]$ bits
Delay requirements on RTT T_n	[100,500] ms
CPU cycles to execute a task X_k	$[10^6, 10^9]$
Effective capacitance ξ	10^{-27}

gorithm is of pseudo-polynomial complexity.

In section 4, the complexity of the energy-optimal offloading algorithm is $O(NCW^2)$. In comparison, the complexity of the approximation offloading algorithm is $O(\lfloor N/\rho \rfloor N^2 W^2)$, which is not related to the scale of the edge cloud C. Thus, if $\lfloor N/\rho \rfloor N \leq C$, the running time of the approximation offloading algorithm is smaller than that of the energy-optimal offloading algorithm. The approximation offloading algorithm should be applied to the system where the scale of the edge cloud is large, and the running time can be greatly reduced compared with the energy-optimal offloading algorithm.



(a) Total energy consumption of tasks.



(b) Amount of saved energy by offloading.

Fig. 2. *Energy saving performances of algorithms. (a) Total energy consumption vs. computational resources of the edge cloud. (b) Amount of saved energy vs. computational resources of the edge cloud.*

The approximation offloading algorithm can also be adaptively adjusted. The algorithm complexity is related to the approximate parameter ρ, and so does the approximation ratio of the algorithm. If ρ gets larger, the approximation offloading algorithm saves more energy while its complexity is larger.

VII. Simulation Results

In this section, the performance of the proposed algorithms is evaluated by Matlab simulations. The parameters of tasks are based on typical real-time applications (e.g., cloud gaming [38] and video transcoding [39]). The parameters of the system are based on general OFDMA networks [14] [16] and cloud servers [15] [34] [37] [40] [42]. The values of parameters are summarized in Table III.

We evaluate the performance of the energy-optimal offloading algorithm and the approximation offloading algorithm, respectively. The exhaustive search algorithm is compared as a benchmark algorithm. To solve the NP-hard Problem P1, the exhaustive search algorithm compares all different task schedules. For each specific task schedule, a feasible resource allocation for all tasks can be derived by theoretical analysis. Because a task can be executed in the mobile device, or offloaded to the edge cloud, or offloaded to remote clouds, the complexity of the exhaustive search algorithm is $O(3^N)$. We also compare our algorithms with the Branch-and-Bound algorithms in [15] and [16], because their algorithms can achieve the minimum energy consumption. The complexity of Branch-and-Bound algorithms is $O(4^N)$.

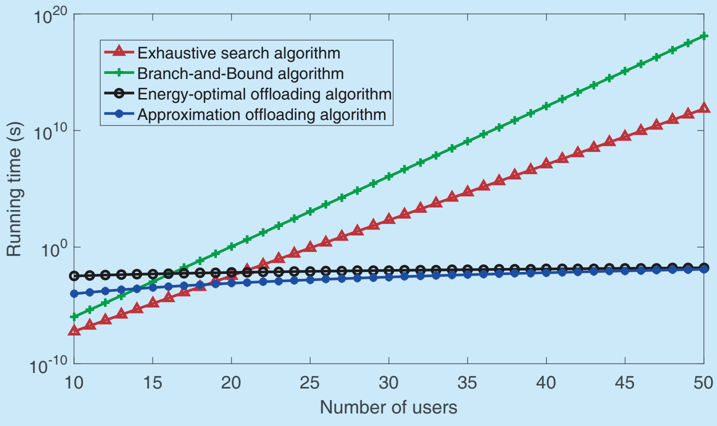
In figure 2(a), the energy consumptions of different algorithms are compared. As proved in Theorem 1, the energy-optimal offloading algorithm minimizes the total energy consumption, which is the same as the exhaustive search algorithm. The approximation offloading algorithm is near-optimal compared with the energy-optimal offloading algorithm.

To evaluate the approximation ratio of the approximation algorithm, we compare the amounts of saved energy between different algorithms, which is shown in figure 2(b). As proved in Theorem 2, the approximation ratio is bounded. If ρ=0.1, the ratio is at least 99.8% in the simulation results, which is proved to be bounded by 90% theoretically. If ρ=0.3, the ratio is at least 99.7% in the simulation results, which is proved to be bounded by 70% theoretically. For the approximation offloading algorithm, the amount of saved energy is bounded, and the bound is touched only in very special cases. In the simulation results, the system parameters are random variables, and the amount of saved energy is much larger than its bound.

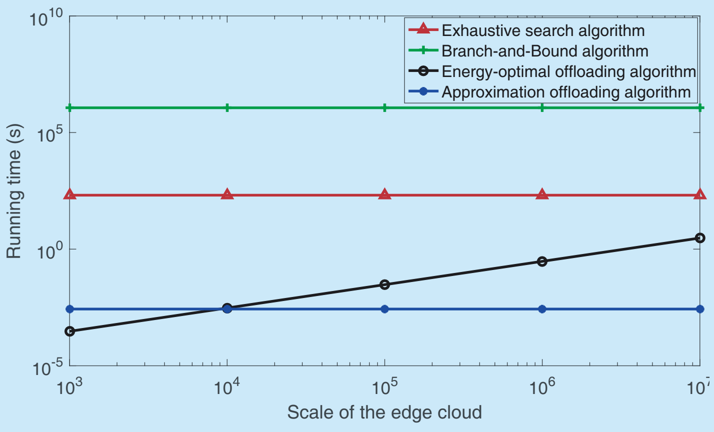
In figure 3(a), the complexities of the algorithms are compared with different numbers of users. Although the exhaustive search algorithm and the Branch-and-Bound algorithm achieve the minimum energy consumption, their running time increase exponentially with the increase of the number of users. In comparison, the energy-optimal offloading algorithm and the approximation offloading algorithm are both of pseudo-polynomial complexities. If the number of users is 30, the running time of the exhaustive search algorithm is 206s, the running time of the Branch-and-Bound is 10^6s, the running time of the energy-optimal offloading algorithm is 10ms, and the running time of the approximation offloading algorithm is 2.7ms. Considering the delay requirements of tasks, which vary from 100ms to 500ms, the running time to handle the energy-optimal offloading algorithm and the approximation offloading algorithm are of slight influence.

In figure 3(b), the complexities of the algorithms are compared with different scales of the edge cloud. The running time of the energy-optimal offloading algorithm is linearly influenced by the scale of the edge cloud. If $\Delta F = 3*10^7$, the scale of the edge cloud is $3*10^4$, and the running time of the energy-optimal offloading algorithm is 10ms. If $\Delta F = 3*10^5$, the scale of the edge cloud is 10^7, and the running time of the energy-optimal offloading algorithm is 3s. In comparison, the running time of the approximation offloading algorithm is not influenced by the scale of the edge cloud, which is 2.7ms. The MEC system can evaluate the running time of different algorithms, and decides which algorithm to be applied.

In figure 4, the benefit of exploiting heterogeneous clouds is demonstrated. With the MEC system, 82.7% energy consumption can be saved by task offloading, compared with mobile device execution. Compared with the MEC system with the edge cloud only, the MEC system with heterogeneous clouds decreases the energy consumption by 21.1%, when the computational resources of the edge



(a) Running time with different numbers of users.



(b) Running time with different scales of the edge cloud.

Fig. 3. *The running time of algorithms. (a) Running time vs. number of users. (b) Running time vs. scale of the edge cloud.*

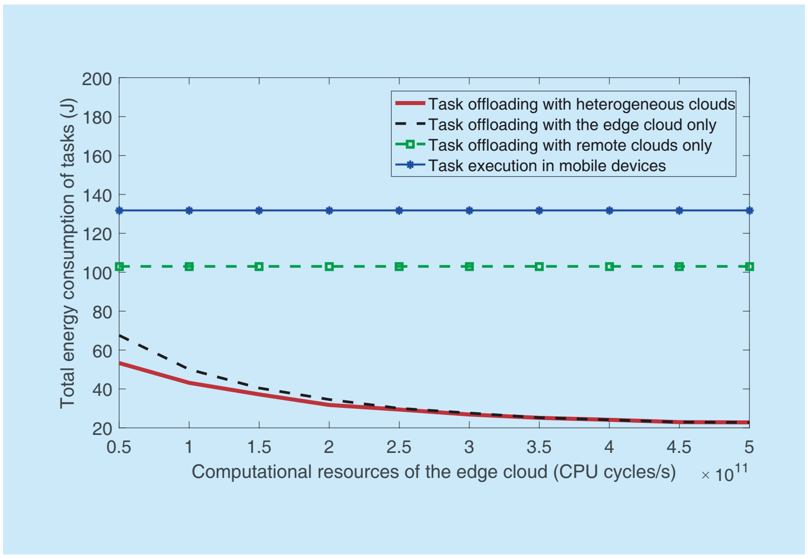


Fig. 4. *Energy saving performances in different systems. Total energy consumption vs. computational resources of the edge cloud.*

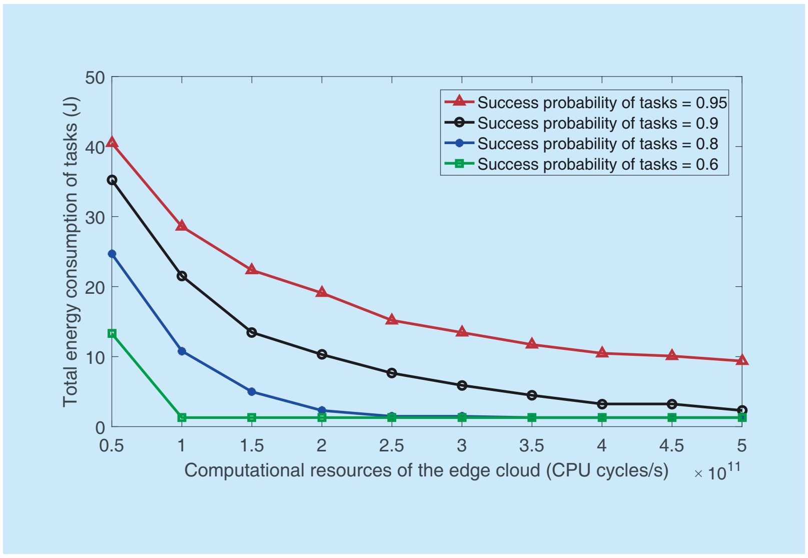


Fig. 5. *Energy saving performances with different requirements on success probabilities. Total energy consumption vs. computational resources of the edge cloud.*

cloud are limited. When the computational resources are sufficient, all of the tasks can be offloaded and the total energy consumption of mobile devices are similar in both of the systems. Compared with the system with remote clouds only, the MEC system with heterogeneous clouds decreases the energy consumption by 77.9%. Actually, in the system with remote clouds only, many tasks cannot be offloaded because of their stringent delay requirements, and the total energy consumption of tasks is quite large because tasks have to be executed in mobile devices.

In figure 5, the trade-off between 1) the energy consumption of tasks and 2) the success probabilities to meet the delay bounds is shown. If mobile users can lower their requirements on success probability, much more energy can be saved. If the requirement on success probability decreases from 95% to 90%, 75.4% energy consumption can be further saved. If the requirement on success probability decreases from 95% to 60%, 95.5% energy consumption can be further saved.

In figure 6, the relationship between delay bounds and energy consumption is shown, where the delay bounds of all tasks are considered as a unique value. If the delay bounds of tasks become larger, each task is allocated with less resources, and much more tasks can be offloaded to cloud servers. By increasing the delay bound from 200ms to 300ms, 73% energy consumption can be further saved. By increasing the delay bound from 100ms to 300ms, 93% energy consumption can be further saved.

VIII. Conclusion

In this work, we study multi-user task offloading in the MEC system with heterogeneous clouds. Our objective is to minimize the total energy consumption of multiple mobile devices, while satisfying different delay requirements of tasks. The system should make immediate decisions on the energy-optimal task schedule and resource allocation, which is a challenging problem. We propose an energy-optimal offloading algorithm to minimize the energy consumption, and it is of pseudo-polynomial complexity. To further reduce the complexity of the algorithm, we propose an approximation offloading algorithm which performs within a bounded gap to the optimum. In simulation results, compared with the scenario that all of the tasks are executed locally in mobile devices, the energy consumption of tasks can be saved by 82.7% with offloading. Compared with the MEC system with the edge cloud only, the MEC system with heterogeneous clouds decreases over 21.1% energy consumption of mobile devices, when the computational resources in the edge cloud are limited.

Appendix

A. Proof of Lemma 1

The mobile device sends the input data in packets, and the data transmission rate is referred to as r_n. According to Shannon capacity, the condition that the packet can be decoded by the base station is $B_n \log(1+\gamma_n) \geq r_n$. The transmission period of each packet is $D_{\mathrm{P},n}=L_n/r_n$. If the data is received within the time $D_{\mathrm{W},n}$, at least one packet is successfully decoded among $\lfloor D_{W,n} / D_{P,n} \rfloor$ packets. Thus, the distribution function of the transmission delay is

$$\mathrm{P}\left\{\mathrm{t} \leq D_{\mathrm{W},n}\right\} = 1-\left(1-\mathrm{P}\left\{B_n \log\left(1+\gamma_n\right) \geq r_n\right\}\right)^{\lfloor D_{\mathrm{W},n}/D_{\mathrm{P},n} \rfloor}. \tag{27}$$

Let $\omega_n = \dfrac{\mathrm{P}\left\{B_n \log\left(1+\gamma_n\right) \geq r_n\right\}}{B_n D_{\mathrm{P},n}}$, and

$$\mathrm{P}\left\{t \leq D_{\mathrm{W},n}\right\} \approx 1-\mathrm{e}^{-B_n \omega_n D_{\mathrm{W},n}}. \tag{28}$$

The maximum ω_n leads to the optimal packet transmission delay. Let $x=r_n/B_n$, and

$$\omega_n = \max_{\{x\}} \frac{\mathrm{P}\{|h_n|^2 \geq \frac{H_n}{\gamma_n}(2^x-1)\}}{L_n / x}. \tag{29}$$

Take (29) into (28), and the distribution of wireless transmission delay is derived.

B. Proof of Proposition 1

In Problem P1, let $F_{\max}\to\infty$, and there is a polynomial reduction from knapsack problem to Problem P1.

For the nth task, if the allocated computational resources are infinite, the task requires bandwidth resources $B_{\mathrm{I,n}}$ to meet the constraints on success probability. Let $E_n=E_{\mathrm{Loc},n}-E_{\mathrm{Off},n}$, which denotes the amount of saved energy by task offloading. To minimize the total energy consumption in Problem P1, the

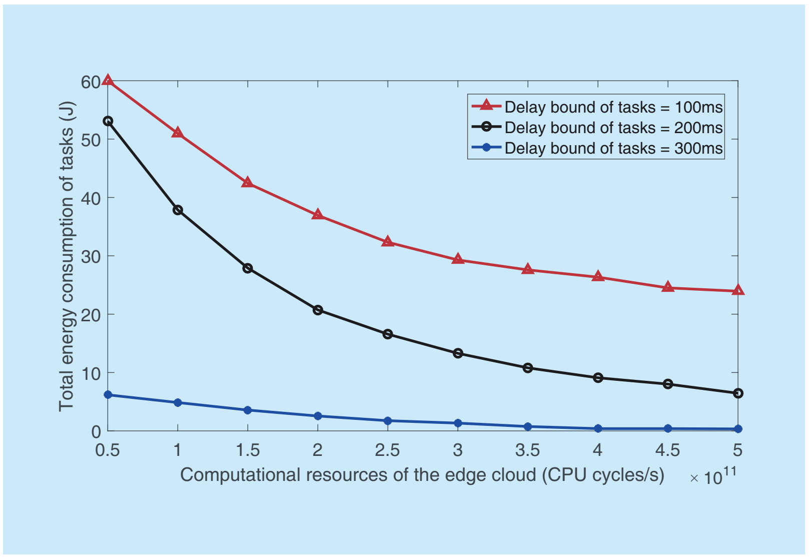


Fig. 6. *Energy saving performances with different delay bounds of tasks. Total energy consumption vs. computational resources of the edge cloud.*

system maximizes the sum of E_n by allocating the total bandwidth resources B_{Max}.

With these parameters, we can also formulate a knapsack problem. Assume that there are totally N items. For the nth item, the size is $B_{\mathrm{I,n}}$, and the price is E_n. The total size of the knapsack is bounded by B_{Max}. Obviously, the reduction process from the knapsack problem to Problem P1 is polynomial to the input size.

When the amount of computational resources in the edge cloud is not limited, knapsack problem is reduced to Problem P1, and solutions to both of them are the same. If Problem P1 is not NP-hard, knapsack problem is also not NP-hard, which is not true. Thus, Problem P1 is NP-hard. e $\mathcal{P}$ is determined by S_{E} and S_{R}.

C. Proof of Theorem1

According to the energy-optimal resource allocation $R(N, B_{\mathrm{Max}}, F_{\mathrm{Max}})$, we define that the bandwidth and computational resources allocated to the kth task are B_k and F_k, respectively. We consider a feasible task schedule $\mathcal{P}$ of S_{E} and S_{R}. To offload S_{E} and S_{R}, we define that the bandwidth and computational resources allocated to the kth task are B_k' and F_k'.

We assume that the task schedule $\mathcal{P}$ consumes less energy than $\mathcal{P}(N, B_{\mathrm{Max}}, F_{\mathrm{Max}})$. Let j be the largest number of task which is only offloaded by either $\mathcal{P}$ or $\mathcal{P}(N, B_{\text{Max}}, F_{\text{Max}})$. Accordingly, $S_{\text{E}}(j, B_{\text{Max}}, F_{\text{Max}})$ and $S_{\text{R}}(j, B_{\text{Max}}, F_{\text{Max}})$ are not updated by S_{E} and S_{R} in the jth stage, which is contradict to the update of value function in Algorithm 1. Thus, Algorithm 1 minimizes the total energy consumption.

D. Proof of Theorem2

The approximate task schedule is of $S_{AE}(N, B_{\text{Max}}, E^*)$ and $S_{AR}(N, B_{\text{Max}}, E^*)$, and the energy-optimal task schedule is of $S_E(N, B_{\text{Max}}, F_{\text{Max}})$ and $S_R(N, B_{\text{Max}}, F_{\text{Max}})$.

Let $S_{App}= S_{AE}(N, B_{\text{Max}}, E^*) \cup S_{AR}(N, B_{\text{Max}}, E^*)$, and $S_{Opt}=S_E(N, B_{\text{Max}}, F_{\text{Max}}) \cup S_R(N, B_{\text{Max}}, F_{\text{Max}})$. The inequation holds as:

$$\begin{aligned}&\sum_{k \in S_{\text{APP}}} E_k \le \sum_{k \in S_{\text{APP}}} \widetilde{E_k} \le \sum_{k \in S_{\text{Opt}}} \widetilde{E_k} \\ &\le \sum_{k \in S_{\text{Opt}}} E_k - \rho E_{\text{Max}} \le (1-\rho) \sum_{k \in S_{\text{Opt}}} E_k\end{aligned}. \tag{30}$$

Accordingly, the approximation ratio of the approximation offloading algorithm is bounded by 1-ρ.

Acknowledgement

This work is sponsored in part by the National Key R&D Program of China 2018YFB1800804, the Nature Science Foundation of China (No. 61871254, No. 61861136003, No. 91638204), and Hitachi Ltd.

References

[1] A. u. R. Khan, M. Othman, S. A. Madani and S. U. Khan, "A survey of mobile cloud computing application models," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 1, First Quarter 2014, pp. 393-413.

[2] K. Kumar and Y. Lu, "Cloud computing for mobile users: can offloading computation save energy?" *Computer*, vol. 43, no. 4, April 2010, pp. 51-56.

[3] J. Liu, E. Ahmed, M. Shiraz, A. Gani, R. Buyya, and A. Qureshi, "Application partitioning algorithms in mobile cloud computing: taxonomy, review and future directions," *Journal of Network and Computer Applications*, vol. 48, Feb 2015, pp. 99-117.

[4] N. Abbas, Y. Zhang, A. Taherkordi and T. Skeie, "Mobile edge computing: a survey," *IEEE Internet of Things Journal*, vol. 5, no. 1, Feb. 2018, pp. 450-465.

[5] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta and D. Sabella, "On multi-access edge computing: a survey of the Emerging 5G Network Edge Cloud Architecture and Orchestration," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, third quarter 2017, pp. 1657-1681.

[6] E. T. S. I. (ETSI), Mobile-edge computing-introductory technical white paper, 2014.

[7] R. Harms and M. Yamartino, The economics of the cloud, Microsoft whitepaper, Microsoft Corporation, 2010.

[8] T. Zhao, S. Zhou, X. Guo and Z. Niu, "Tasks scheduling and resource allocation in heterogeneous cloud for delay-bounded mobile edge computing," *2017 IEEE International Conference on Communications (ICC)*, 2017, pp. 1-7.

[9] Z. Sanaei, S. Abolfazli, A. Gani and R. Buyya, "Heterogeneity in mobile cloud computing: taxonomy and open challenges," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 1, First Quarter 2014, pp. 369-392.

[10] P. Mach and Z. Becvar, "Mobile edge computing: a survey on architecture and computation offloading," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, third quarter 2017, pp. 1628-1656.

[11] H. Wu, "Multi-objective decision-making for mobile cloud offloading: a survey," *IEEE Access*, vol. 6, 2018, pp. 3962-3976.

[12] S. Sardellitti, G. Scutari and S. Barbarossa, "Joint optimization of radio and computational resources for multicell mobile-edge computing," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 1, no. 2, June 2015, pp. 89-103.

[13] J. Cheng, Y. Shi, B. Bai and W. Chen, "Computation offloading in cloud-RAN based mobile cloud computing system," *2016 IEEE International Conference on Communications (ICC)*, 2016, pp. 1-6.

[14] C. You, K. Huang, H. Chae and B. Kim, "Energy-efficient resource allocation for mobile-edge computation offloading," *IEEE Transactions on Wireless Communications*, vol. 16, no. 3, March 2017, pp. 1397-1411.

[15] P. Zhao, H. Tian, C. Qin and G. Nie, "Energy-saving offloading by jointly allocating radio and computational resources for mobile edge computing," *IEEE Access*, vol. 5, 2017, pp. 11255-11268.

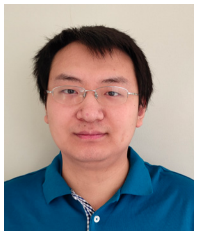
[16] X. Yang, X. Yu, H. Huang and H. Zhu, "Energy efficiency based joint computation offloading and resource allocation in multi-access MEC systems," *IEEE Access*, vol. 7, 2019, pp. 117054-117062.

[17] H. Wu , Y. Sun, K. Wolter. "Energy-efficient decision making for mobile cloud offloading." *IEEE Transactions on Cloud Computing*, pp. 1-1, 2018.

[18] H. Wu and K. Wolter, "Stochastic analysis of delayed mobile offloading in heterogeneous networks," *IEEE Transactions on Mobile Computing*,

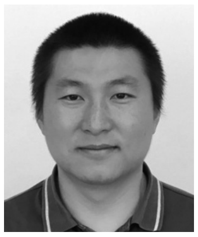
vol. 17, no. 2, Feb. 2018, pp. 461-474.
[19] Y. Liu, "Exploiting NOMA for cooperative edge computing," *IEEE Wireless Communications*, vol. 26, no. 5, October 2019, pp. 99-103.
[20] L. Liu, B. Sun, X. Tan, Y. S. Xiao and D. H. K. Tsang, "Energy-efficient resource allocation and channel assignment for NOMA-based mobile edge computing," *2019 IEEE Wireless Communications and Networking Conference (WCNC)*, 2019, pp. 1-6.
[21] M. Zeng and V. Fodor, "Energy-efficient resource allocation for NOMA-assisted mobile edge computing," *2018 IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, 2018, pp. 1794-1799.
[22] Z. Yang, J. Hou and M. Shikh-Bahaei, "Energy efficient resource allocation for mobile-edge computation networks with NOMA," *2018 IEEE Globecom Workshops (GC Wkshps)*, 2018, pp. 1-7.
[23] Y. Pan, M. Chen, Z. Yang, N. Huang and M. Shikh-Bahaei, "Energy-efficient NOMA-based mobile edge computing offloading," *IEEE Communications Letters*, vol. 23, no. 2, Feb. 2019, pp. 310-313.
[24] Z. Yang, C. Pan, J. Hou and M. Shikh-Bahaei, "Efficient resource allocation for mobile-edge computing networks with NOMA: completion time and energy minimization," *IEEE Transactions on Communications*, vol. 67, no. 11, Nov. 2019, pp. 7771-7784.
[25] W. Chen, B. Liu, H. Huang, S. Guo and Z. Zheng, "When UAV swarm meets edge-cloud computing: the QoS perspective," *IEEE Network*, vol. 33, no. 2, March/April 2019, pp. 36-43.
[26] S. Jeong, O. Simeone and J. Kang, "Mobile edge computing via a UAV-mounted cloudlet: optimization of bit allocation and path planning," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 3, March 2018, pp. 2049-2063.
[27] Z. Yang, C. Pan, K. Wang and M. Shikh-Bahaei, "Energy efficient resource allocation in UAV-enabled mobile edge computing networks," *IEEE Transactions on Wireless Communications*, vol. 18, no. 9, Sept. 2019, pp. 4576-4589.
[28] S. Wang, J. Xu, N. Zhang and Y. Liu, "A survey on service migration in mobile edge computing," *IEEE Access*, vol. 6, 2018, pp. 23511-23528.
[29] Y. Sun, S. Zhou and J. Xu, "EMM: energy-aware mobility management for mobile edge computing in ultra dense networks," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, Nov. 2017, pp. 2637-2646.
[30] T. Ouyang, Z. Zhou and X. Chen, "Follow me at the edge: mobility-aware dynamic service placement for mobile edge computing," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 10, Oct. 2018, pp. 2333-2345.
[31] H. Wu, W. J. Knottenbelt and K. Wolter, "An efficient application partitioning algorithm in mobile environments," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 7, July 2019, pp. 1464-1480.
[32] E. Cuervo, A. Balasubramanian, D. K. Cho, et al. "MAUI: making smartphones last longer with code offload." *Proceedings of the 8th International Conference on Mobile Systems*, Applications, and Services (MobiSys 2010), June 2010.
[33] W. Zhang, Y. Wen, K. Guan, D. Kilper, H. Luo and D. O. Wu, "Energy-optimal mobile cloud computing under stochastic wireless channel," *IEEE Transactions on Wireless Communications*, vol. 12, no. 9, September 2013, pp. 4569-4581.
[34] Y. Yu, J. Zhang and K. B. Letaief, "Joint subcarrier and CPU time allocation for mobile edge computing," *2016 IEEE Global Communications Conference (GLOBECOM)*, 2016, pp. 1-6.
[35] G. Hooghiemstra and P. Van Mieghem. Delay distributions on fixed internet paths. Delft University of Technology, Tech. Rep., 2001.
[36] S. I. Gass and C. M. Harris, Encyclopedia of operations research and management science. Kluwer Academic Publishers, 1996.
[37] D. Minarolli and B. Freisleben, "Utility-based resource allocation for virtual machines in cloud computing," *2011 IEEE Symposium on Computers and Communications (ISCC)*, 2011, pp. 410-417.
[38] M. Claypool, K. Claypool. "Latency and player actions in online games," *Communications of the Acm*, 2006..
[39] G. Gao, W. Zhang, Y. Wen, Z. Wang and W. Zhu, "Towards cost-efficient video transcoding in media cloud: insights learned from user viewing patterns," *IEEE Transactions on Multimedia*, vol. 17, no. 8, Aug. 2015, pp. 1286-1296.
[40] M. DeVirgilio, W. David Pan, L. L. Joiner and D. Wu, "Internet delay statistics: measuring internet feel using a dichotomous hurst parameter," *2013 Proceedings of IEEE Southeastcon*, 2013, pp. 1-6.
[41] L. Fan, W. Yan, X. Chen, Z. Chen and Q. Shi, "An energy efficient design for UAV communication with mobile edge computing," *China Communications*, vol. 16, no. 1, Jan. 2019, pp. 26-36.
[42] F. Wei, S. Chen and W. Zou, " A greedy algorithm for task offloading in mobile edge computing system," *China Communications*, vol. 15, no. 11, Nov. 2018, pp. 149-157.

Biographies



Tianchu Zhao, received the B.S. degree in electronic engineering from Tsinghua University, Beijing, China, in 2013, where he is currently pursuing the Ph.D. degree with the Department of Electronic Engineering. His research interests include mobile edge computing and cloud computing.

Sheng Zhou, received the B.E. and Ph.D. degrees in electronic engineering from Tsinghua University, Beijing, China, in 2005 and 2011, respectively. In 2010, he was a Visiting Student with the Wireless System Lab, Department of Electrical Engineering, Stanford University, Stanford, CA, USA. From 2014 to 2015, he was a Visiting Researcher with the Central Research Lab, Hitachi Ltd., Japan. He is currently an Associate Professor with the Department of Electronic Engineering, Tsinghua University. His research interests include cross-layer design for multiple antenna systems, mobile edge computing, vehicular networks, and green wireless communications. He received the IEEE ComSoc Asia–Pacific Board Outstanding Young Researcher Award in 2017.



Linqi Song, is an Assistant Professor in the Computer Science Department at the City University of Hong Kong, China. Prior to that, he was a Postdoctoral Scholar at the University of California, Los Angeles in the Electrical and Computer Engineering Department. He received the B.S. and M.S. degrees in Electronic Engineering, Tsinghua University, China and the Ph.D. degree in Electrical Engineering from UCLA. He received a UCLA Fellowship for his graduate studies. His research interests are in content-type coding, index coding, network coding, algorithms, big data, and machine learning.

Zhiyuan Jiang, received his B.E., Ph.D. degrees from the electronic engineering department of Tsinghua University in 2010, 2015, respectively. He is currently a Professor with the School of Communication and Information Engineering, Shanghai University. He visited University of Southern California during 2013-2014 and 2017-2018. He worked as an experienced researcher and a wireless signal processing scientist for Ericsson and Intel Labs in 2015-2016 and 2018, respectively. He was selected into the high-level talents overseas program of Shanghai and the program for professor of special appointment of Shanghai in 2019. He received the best in-session presentation award of IEEE INFOCOM 2019, and exemplary reviewer award of IEEE Wireless Communications Letters 2019. He served as an associate editor of IEEE/KICS Journal of Communications and Networks. His main research interests include low-latency wireless networks and the design, implementation of multi-antenna communication systems.

Xueying Guo, received her B.E. and Ph.D. degrees from the Department of Electronic Engineering, Tsinghua University, Beijing, China, in 2011 and 2017, respectively. From Oct. 2013 to Oct. 2014, she was a Visiting Scholar in the Department of Electrical and Computer Engineering, Texas A\&M University, College Station, TX, USA. From June 2016 to Sep. 2016, she worked as a research intern at Microsoft Research Asia and received the reward of excellent in Microsoft Research Asia internship program. From Feb. 2017 to Feb. 2019, She was a postdoctoral researcher in the Department of Computer Science, University of California, Davis, CA, USA. From Feb. 2019 till now, she is a machine learning engineer in Google. Her research interests include machine learning, reinforcement learning, and data-driven networking. She was the recipient of the Best Student Paper Award in the 25th International Teletraffic Congress (ITC) in 2013.

Zhisheng Niu, received the B.E. degree from Beijing Jiaotong University, China, in 1985, and the M.E. and D.E. degrees from the Toyohashi University of Technology, Japan, in 1989 and 1992, respectively. From 1992 to 1994, he was with Fujitsu Laboratories Ltd., Japan. In 1994, he joined Tsinghua University, Beijing, China, where he is currently a Professor with the Department of Electronic Engineering. His major research interests include queuing theory, traffic engineering, mobile Internet, radio resource management of wireless networks, and green communication and networks. He received the Outstanding Young Researcher Award from the Natural Science Foundation of China in 2009 and the Best Paper Award from the IEEE Communication Society Asia–Pacific Board in 2013. He has served as a Chair for the Emerging Technologies Committee from 2014 to 2015, the Director for Conference Publications from 2010 to 2011, and the Director for Asia–Pacific Board in the IEEE Communication Society, from 2008 to 2009. He is currently serving as the Director for Online Contents from 2018 to 2019 and as an Area Editor for the IEEE Transactions on Green Communications and Networking. He was also selected as a Distinguished Lecturer of the IEEE Communication Society from 2012 to 2015 and the IEEE Vehicular Technologies Society from 2014 to 2018. He is a fellow of IEICE.