# Multiuser Computation Offloading and Downloading for Edge Computing With Virtualization

Zezu Liang, *Student Member, IEEE*, Yuan Liu, *Senior Member, IEEE*, Tat-Ming Lok, *Senior Member, IEEE*, and Kaibin Huang, *Senior Member, IEEE*

***Abstract*— Mobile-edge computing (MEC) is an emerging technology for enhancing the computational capabilities of the mobile devices and reducing their energy consumption via offloading complex computation tasks to the nearby servers. Multiuser MEC at servers is widely realized via parallel computing based on virtualization. Due to finite shared I/O resources, interference between virtual machines (VMs), called I/O interference, degrades the computation performance. In this paper, we study the problem of joint radio-and-computation resource allocation (RCRA) in multiuser MEC systems in the presence of I/O interference. Specifically, offloading scheduling algorithms is designed targeting two system performance metrics: sum offloading rate maximization and sum mobile energy consumption minimization. Their designs are formulated as non-convex mixed-integer programming problems, which account for latency due to offloading, result downloading, and parallel computing. A set of low-complexity algorithms are designed based on a decomposition approach and leveraging classic techniques from combinatorial optimization. The resultant algorithms jointly schedule offloading users, control their offloading sizes, and divide time for communication (offloading and downloading) and computation. They are either optimal or can achieve close-to-optimality as shown by simulation. The comprehensive simulation results demonstrate that considering of I/O interference can endow on an offloading controller robustness against the performance-degradation factor.**

***Index Terms*— Mobile edge computing (MEC), parallel computing, I/O interference, virtual machine (VM), resource allocation.**

## I. Introduction

DRIVEN by the increasing popularity of mobile devices (e.g., smart phones, tablets, wearable devices), a wide range of new mobile applications (e.g., augmented reality, face recognition, interactive online-gaming) are emerging. They usually require intensive computation to enable real-time machine-to-machine and machine-to-human interactions. The limited energy and computation resources at the mobile devices may not be sufficient for meeting the requirement. To address these limitations, mobile-cloud computing (MCC) [1] offers one possible solution by migrating the computation-intensive tasks from mobiles to the cloud. However, data propagation through wide area networks (including the radio-access network, backhaul-network, and Internet) can cause excessive latency. Therefore, MCC may not be able to support latency-critical applications.

Recently, mobile-edge computing (MEC) [2], [3], which provides users computing services using servers at the network edge, is envisioned as a promising way to enable computation-intensive and latency-sensitive mobile applications. Compared with MCC, users in MEC systems offload tasks to the proximate edge servers [e.g., base stations (BSs) and access points (APs)] for execution, which avoids data delivery over the backhaul networks and thereby dramatically reduces latency. An essential technology for implementing MEC is virtualization, referring to sharing of a physical machine (server) by multiple computing processes via the execution of virtual machines (VMs). Specifically, each VM is a virtual computer configured with a certain amount of the server's hardware resource (such as CPU, memory and I/O bus). According to technical standards for the MEC server architecture [2], the virtualization functionality is supported by a virtualization layer and a virtualization manager. The virtualization layer virtualizes the MEC hosting infrastructure by abstracting the detailed hardware implementation, while the virtualization manager provides the virtualized computer infrastructure as a service (IaaS). Applications run on top of an IaaS and are deployed within the packaged-operating systems (i.e.,VMs) that are well-isolated with the others. To this end, the MEC server can isolate co-hosted applications and provide multi-service support. Nevertheless, it has been shown in the literature [4]–[6] that sharing the same physical platform can incur the so-called I/O interference among VMs, resulting in

Manuscript received February 1, 2019; revised April 20, 2019; accepted June 4, 2019. Date of publication June 19, 2019; date of current version September 10, 2019. The work of Y. Liu was supported in part by the Natural Science Foundation of China under Grant U1701265 and in part by the Pearl River Science and Technology Nova Program of Guangzhou under Grant 201710010111. The work of T.-M. Lok was supported in part by the General Research Fund from the Research Grants Council, Hong Kong, under Project CUHK 14201118. This paper was presented in part at the IEEE International Conference on Communications (ICC), Shanghai, China, May 2019. The associate editor coordinating the review of this paper and approving it for publication was A. Banchs. *(Corresponding author: Yuan Liu.)*

Z. Liang and T.-M. Lok are with the Department of Information Engineering, The Chinese University of Hong Kong, Hong Kong (e-mail: lz017@ie.cuhk.edu.hk; tmlok@ie.cuhk.edu.hk).

Y. Liu is with the School of Electronic and Information Engineering, South China University of Technology, Guangzhou 510641, China (e-mail: eeyliu@scut.edu.cn).

K. Huang is with the Department of Electrical and Electronic Engineering, The University of Hong Kong, Hong Kong (e-mail: huangkb@eee.hku.hk).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TWC.2019.2922613

a certain degree of computation-speed reduction for each VM. As far as we know, prior research of this issue focuses on the interference modeling [5]–[7] and computation resource provisioning [8]. No previous works related to the computation offloading coupled with joint radio-and-computational resource allocation (RCRA) have been studied before. In this paper, we investigate the multiuser offloading problem in a MEC system in the presence of I/O interference.

### A. Prior Work

In recent years, extensive research has been conducted on efficient computation offloading for MEC systems. For single-user MEC systems, a research focus is designing policies for task assignment or partitioning. A binary-offloading policy (decides on whether an entire task should be offloaded for edge execution or computed locally) has been widely investigated in different system scenarios, including stochastic wireless channels [9], MEC systems powered by energy harvesting [10] or wireless energy transfer [11]. Based on program partitioning, partial offloading is possible where a computation task at a user can be partitioned into multiple parts for local computing and offloading at the same time. The optimal partial-offloading strategies are studied in [12], [13].

For multiuser MEC systems, the efficient computation offloading designs requires joint optimization of RCRA, i.e., how to allocate the finite radio-and-computational resources to multiple users for achieving a system-level objective, e.g., the sum energy consumption minimization. The problem is challenging as multiplicity of parameters and constraints are involved such as multi-user channel states and task information, computation capacities of servers and users, and deadline constraints. In [14], the resource-allocation strategies were proposed based on time-division multiple access (TDMA) and orthogonal frequency-division multiple access (OFDMA). It is assumed that the task-execution durations at the edge cloud are negligible, overlooking the effect of finite computation resources at servers in offloading decisions. In [15], [16], game theory was applied to designing efficient distributed offloading schemes. [17], [18] studied the multi-cell MEC systems, where joint RCRA under given offloading decisions was optimized in [17] while [18] further incorporated offloading decisions into optimization. In [19], [20], dynamic offloading policies were proposed to investigate the energy-delay tradeoff for stochastic MEC systems. Energy-efficient offloading designs have also been studied in other scenarios like wireless power transfer [21], [22] and cooperative transmissions [23]–[25]. The work in [26] is closely related to this paper, as they both address parallel computation at a MEC server for joint RCRA. However, simultaneous computation processes at the same server are assumed in [26] to be independent and conditioned on partitioned computation resources. The effect of I/O interference is neglected despite its being an importance issue in virtualization.

Omitting I/O interference in multiuser MEC based on virtualization leads to the unrealistic assumption that the total computation resource at a server remains fixed regardless of the number of VMs. In reality, the resource reduces as the number grows due to I/O interference. Thus, the number of VMs per server is usually constrained in practice, so as to maintain the efficiency in resource utilization. Despite its importance, I/O interference has received little attention in the literature. It motivates the current work on accounting for the factor in resource allocation for MEC systems.

### B. Contributions and Organization

In this paper, we revisit the RCRA problem in multiuser offloading and address the following two practical issues that have received scant attention in the literature.

1) (I/O interference) The I/O interference in practical parallel computing has been largely neglected in the existing "cake-slicing" model of computing-resource allocation (see e.g., [19], [26]). Considering I/O interference introduces a *dilemma*: scheduling more offloading users increases the multiplexing gain in parallel computing but degrades the speeds of individual VMs due to their interference.
2) (Result downloading) The communication overhead for computation-result downloading is commonly assumed in the literature to be negligible compared with that for offloading. The assumption does not always hold in applications such as augmented reality and image processing. Considering downloading complicates scheduling as the policy needs account for not only users' uplink channel states but also downlink states as well as the output-input-size ratio for each task.

In this paper, we consider a multiuser MEC system where parallel computing at the server is based on virtualization. The I/O interference is modeled using a practical model developed based on measurement data [7]. While the literature focuses on offloading latency, we consider offloading, parallel computing and downloading as factors contributing to latency. Based on the assumptions, scheduling policies are designed by solving two RCRA problems based on two criteria, namely *maximizing the sum offloading rate* and *minimizing the sum mobile energy consumption*, both under a latency constraint. The main contributions are summarized as follows.

- (Sum Offloading Rate Maximization) Based on this criterion, the RCRA problem is formulated as a non-convex problem for joint optimization of offloading scheduling, offloaded-data sizes, and communication-and-computation time division. By analyzing its properties, we present a solution approach of decomposing the problem into master and slave sub-problems. The former optimizes the number of offloading users and given the number, the later optimizes offloading-user set, offloaded-data sizes, and time division (offloading, computing, downloading). By adopting Dinkelbach method, an efficient iteratively algorithm is designed to solve the slave problem that is a *combinatorial-optimization* problem. With the algorithm, the master problem can be then solved by a simple search over a finite integer set of possible numbers of offloading users. In addition, special cases are studied to yield useful design guidelines.

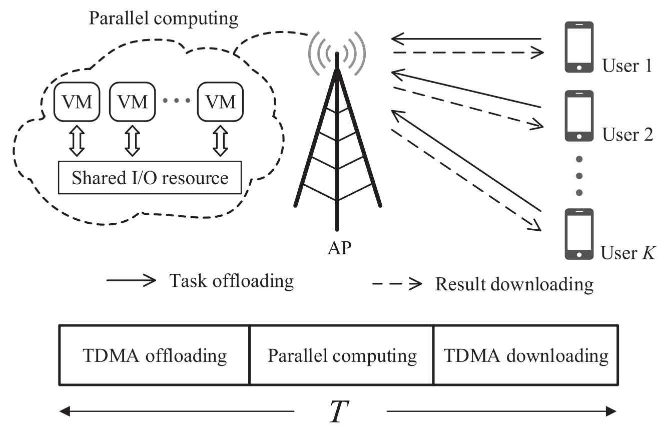


Fig. 1. A multiuser MEC system with a single AP and $K$ users.

- (Sum Mobile Energy Minimization) The problem of sum-energy minimization is also non-convex. To develop practical scheduling algorithms for efficiently solving the problem, we divide the whole user set into multiple subsets based on the corresponding levels of offloading gain in terms of energy reduction. Then some reasonable rules are introduced to prioritize the user subsets' offloading so as to enable tractable algorithmic design. Based on the rules, an efficient greedy algorithm is proposed to schedule different subsets of users which achieves close-to-optimal performance as demonstrated by simulation.

## II. System Model and Problem Formulation

### A. System Model

Consider an MEC system shown in Fig. 1, consisting of one AP integrated with an MEC server and $K$ users. Partial offloading is assumed in this paper so that each user can partition its computation task into two independent parts for local computing and offloading to the server. The two operations are simultaneous as the communication modulars and user CPUs are separated. All of the users have to complete their tasks within a fixed duration $T$ (in second) so as to meet a real-time requirement. The system operation is divided into three *sequential phases*: 1) TDMA-based task offloading by users, 2) parallel computing at the server, and 3) TDMA-based computation-result downloading from the server to users. Here we assume that the parallel-computing phase begins when all scheduled users' offloading are completed. Note that a more efficient way is the asynchronous offloading [27], i.e., the parallel computing can happen when a set of the scheduled users complete the offloading in the TDMA process and then for another set of the sequential scheduled users, until all the scheduled users' tasks are executed. That is, parallel computing is executed several times during the TDMA-based offloading process. This may be more efficient since the TDMA-based offloading process can also be used to parallel computing that runs a few VMs in each time. In this case, the scheduling order of users is important and needed for consideration. In this paper, we consider the above three sequential phases for ease of implementation and frame structure. The asynchronous offloading will be an interesting problem in future work. Corresponding models and assumptions are described as follows.

*1) Offloading and Downloading Phases:* Let $\ell_i$ denote the input data bits offloaded by user $i$ to the server. It is assumed that each input bit generates $\gamma_i$ bits of computation result. Then for an offloaded data $\ell_i$, the computed result contains $\gamma_i \ell_i$ bits. The transmission delay for user $i$ for offloading and downloading can be written separately as

$$t_i^u = a_i \ell_i, \tag{1}$$
$$t_i^d = b_i \gamma_i \ell_i, \tag{2}$$

where $a_i$ and $b_i$ are the required time for transmitting a single bit in uplink and downlink, respectively, which are the inverse of the corresponding uplink and downlink rates. In this paper, we assume that the users' transmit powers are determined by some power control algorithms (see e.g., [28]). Therefore, for a given channel realization, the data rates are fixed for one channel realization but can vary over different channel realizations.

*2) Parallel-Computing Phase With Virtualization:* After receiving all the offloaded tasks, the server executes them in parallel by creating multiple VMs. We consider the important factor of I/O interference in parallel computing [29] and adopt a model developed in the literature based on measurement data [3], [7], which is described as follows. Group the user indices into the set $\mathcal{K}$. The subset $\mathcal{S} \subseteq \mathcal{K}$ identifies the set of scheduled offloading users, $t_e$ the time allocated to the parallel-computing phase, and $r_i$ the expected computation-service rate (bits/sec) of a VM given task $i$ when running in isolation. Following [3], [7], a performance degradation factor $d > 0$ is defined to specify the percentage reduction in the computation-service rate of a VM when multiplexed with another VM[1]. Suppose that one VM is created and assigned to a task, the degraded computing rate for each task is modeled as $r_i(1+d)^{1-|\mathcal{S}|}$ [7], where $|\mathcal{S}|$ denotes the number of tasks (or offloading users) for parallel computing. Therefore, for given $t_e$, the numbers of offloadable bits are constrained by

$$0 \le \ell_i \le t_e r_i (1+d)^{1-|\mathcal{S}|}, \quad \forall i \in \mathcal{S}. \tag{3}$$

The constraints in (3) show that the maximum number of offloadable bits per user decreases with the number of offloaded tasks due to the I/O interference in parallel computing. Moreover, relaxing the duration for parallel computing ($t_e$) can accommodate more offloaded bits ($\{\ell_i\}$), however, at the cost of less time for the offloading and downloading phases. This introduces a tradeoff between the three phases under the following total-latency constraint:

$$\sum_{i\in\mathcal{S}} t_i^u + t_e + \sum_{i\in\mathcal{S}} t_i^d = \sum_{i\in\mathcal{S}} \ell_i \left(a_i + b_i \gamma_i\right) + t_e \le T. \tag{4}$$

### B. Problem Formulation

In this paper, we consider two popular system-performance metrics: sum offloading rate maximization and sum energy

[1] The parameter $d$ depends on the specific VM multiplexing and placement strategy [30], [31]. Its value can be estimated by theoretical studies or statistical observations.

consumption minimization by users. The metrics target two different scenarios where users are constrained in computing capacity and energy, respectively. Correspondingly, offloading aims at either enhancing user capacities or reducing their energy consumption. Using the metrics, two RCRA problems are formulated as follows.

*1) Sum Offloading Rate Maximization:* The objective is to maximize the weighted sum of the users' offloading rates by joint offloading-user scheduling, offloaded-bits control, and three-phase time allocation. Here, the sum offloading rate is defined as the sum offloadable bits over the time duration $T$. Let $\omega_i$ denote a positive weight assigned to user $i$ based on the users' priority. Mathematically, the optimization problem can be formulated as

$$\text{(P1)}: \max_{\mathcal{S}\subseteq\mathcal{K},\{\ell_i\},t_e} \quad R=\frac{1}{T}\sum_{i\in\mathcal{S}}\omega_i\ell_i \tag{5a}$$

$$\text{s.t.} \quad \sum_{i\in\mathcal{S}}\ell_i\left(a_i+b_i\gamma_i\right)+t_e\le T, \tag{5b}$$

$$0\le\ell_i\le t_e r_i\left(1+d\right)^{1-|\mathcal{S}|}, \quad \forall i\in\mathcal{S}. \tag{5c}$$

Problem (P1) is a mixed-integer programming problem comprising both a combinatorial variable $\mathcal{S}$ and continuous variables $(\{\ell_i\},t_e)$ and non-convex constraints (5c). Therefore, Problem (P1) is *non-convex*. Though such a problem is usually difficult to solve exactly, an algorithm is designed in sequel to find the optimal solution.

*2) Sum Energy Minimization:* We aim at minimizing the total energy consumed at all the users. Suppose that each user $i\in\mathcal{K}$ has a computation task of length $L_i$ bits, of which $\ell_i$ bits are offloaded to the AP and $(L_i-\ell_i)$ bits computed locally. In parallel with offloading, the allowed duration for local computing at any user is $T$. Let the duration for user $i$ be denoted as $t_i^{\text{loc}}$. Then we have the following time constraint on local computing:

$$t_i^{\text{loc}}=\frac{c_i(L_i-\ell_i)}{f_i}\le T, \quad \forall i\in\mathcal{K}, \tag{6}$$

where $c_i$ denotes the fixed number of CPU cycles required to compute a single bit and $f_i$ denotes the CPU frequency at user $i$ (CPU cycles/sec). The energy consumption for computing $(L_i-\ell_i)$ bits at user $i$ can be written as

$$E_i^{\text{loc}}=\kappa_i c_i(L_i-\ell_i)f_i^2, \tag{7}$$

where $\kappa_i$ is a coefficient depending on the specific hardware architecture. Combining constraints (3) and (6) and considering the fact that $0\le\ell_i\le L_i$ yield the constraint on the number of offloadable bits as

$$L_i^{\min}\le\ell_i\le\min\left\{L_i,t_e r_i\left(1+d\right)^{1-|\mathcal{S}|}\right\}, \quad \forall i\in\mathcal{S}, \tag{8}$$

where $L_i^{\min}\triangleq\left[L_i-\frac{Tf_i}{c_i}\right]^+$, with $[\cdot]^+\triangleq\max\{\cdot,0\}$, is derived from (6) by setting $t_i^{\text{loc}}=T$. On the other hand, the energy consumption for offloading $\ell_i$ bits is $E_i^{\text{off}}=t_i^u p_i=a_i p_i\ell_i$. Therefore, the total energy consumption of each user is $E_i=E_i^{\text{loc}}+E_i^{\text{off}}$. Then, the weighted sum energy consumption of all users can be expressed as

$$\begin{aligned} E&=\sum_{i\in\mathcal{K}}\omega_i\left[E_i^{\text{loc}}+E_i^{\text{off}}\right]\\ &=\sum_{i\in\mathcal{K}}\omega_i\left[\kappa_i c_i\left(L_i-\ell_i\right)f_i^2+a_i p_i\ell_i\right]\\ &=\sum_{i\in\mathcal{K}}\omega_i\left(a_i p_i-\kappa_i c_i f_i^2\right)\ell_i+\sum_{i\in\mathcal{K}}\omega_i\kappa_i c_i L_i f_i^2\\ &=\sum_{i\in\mathcal{K}}\theta_i\ell_i+\mathrm{e}_0, \end{aligned} \tag{9}$$

where $\theta_i\triangleq\omega_i\left(a_i p_i-\kappa_i c_i f_i^2\right)$ and $\mathrm{e}_0\triangleq\sum_{i\in\mathcal{K}}\omega_i\kappa_i c_i L_i f_i^2$ are both constants.

Given the objective of minimizing the sum-energy consumption in (9) subject to the time constraint in (4) and the offloadable bits constraints in (8), the corresponding RCRA problem can be formulated as

$$\text{(P2)}: \min_{\mathcal{S}\subseteq\mathcal{K},\{\ell_i\},t_e} \quad \sum_{i\in\mathcal{S}}\theta_i\ell_i \tag{10a}$$

$$\text{s.t.} \quad \sum_{i\in\mathcal{S}}\ell_i\left(a_i+b_i\gamma_i\right)+t_e\le T, \tag{10b}$$

$$L_i^{\min}\le\ell_i\le\min\left\{L_i,t_e r_i\left(1+d\right)^{1-|\mathcal{S}|}\right\}, \quad \forall i\in\mathcal{S}, \tag{10c}$$

where the objective function (10a) is derived from (9) by omitting the constant term $\mathrm{e}_0$ and combining the fact that $\ell_i=0$ for all non-scheduled users.

Like Problem (P1), Problem (P2) is also *non-convex*. An efficient algorithm is developed in the sequel to approximately solve this problem.

## III. Sum Offloading Rate Maximization

In this section, we develop an optimal algorithm for solving Problem (P1). First, an important property of the optimal offloading-user set $\mathcal{S}^*$ will be obtained, which allows tractable analysis of the optimal offloading scheme and thereby simplifies the problem. Subsequently, an iterative algorithm based on the Dinkelbach method is proposed to exactly solve the simplified problem. Last, we discuss several special cases to obtain useful insights.

### *A. Sum Offloading Rate Maximization for a Given Offloading-User Set*

We made a key observation that Problem (P1) becomes a linear programming (LP) problem if the offloading-user set is given. The conditional optimal offloading strategy, specified by the offloaded data sizes $\{\ell_i^*\}$, satisfies the following property.

*Lemma 1:* Given an arbitrary offloading-user set $\mathcal{S}$, the optimal offloading strategy $\{\ell_i^*\mid i\in\mathcal{S}\}$ must be the maximum or minimum value in the constraint in (5c).

*Proof:* See Appendix A. ■

Lemma 1 indicates that the optimal offloading strategy of each scheduled user follows a binary policy, i.e., offloading with the maximum data size or nothing. Accordingly, we can divide the elements in $\{\ell_i\mid i\in\mathcal{S}\}$ into two groups, with one group $\widetilde{\mathcal{S}}$ for the users offloading maximum bits and the other performing no offloading, i.e.,

$$\ell_i = \begin{cases} t_e r_i (1+d)^{1-|\mathcal{S}|}, & i \in \widetilde{\mathcal{S}}, \\ 0, & i \in \mathcal{S}\backslash\widetilde{\mathcal{S}}. \end{cases} \tag{11}$$

Note that $\widetilde{\mathcal{S}}$ is needed to be determined and we first use $\widetilde{\mathcal{S}}$ to express $\{\ell_i\}$ and $t_e$. It is intuitive that the equality must hold in constraint (5b) for the optimal solution of Problem (P1). Then, by substituting (11) into (5b), we obtain the conditional optimal parallel-computing time $t_e$ as

$$t_e = \frac{T(1+d)^{|\mathcal{S}|-1}}{(1+d)^{|\mathcal{S}|-1} + \sum_{i\in\widetilde{\mathcal{S}}}(a_i + b_i\gamma_i)r_i}. \tag{12}$$

Combining (11) and (12), Problem (P1) for a given $\mathcal{S}$ can be formulated as one for determining the subset $\widetilde{\mathcal{S}}$ in $\mathcal{S}$:

$$\max_{\widetilde{\mathcal{S}}\subseteq\mathcal{S}} \quad R = \frac{\sum_{i\in\widetilde{\mathcal{S}}}\omega_i r_i}{(1+d)^{|\mathcal{S}|-1} + \sum_{i\in\widetilde{\mathcal{S}}}(a_i + b_i\gamma_i)r_i}. \tag{13}$$

It is observed that, for any given $\mathcal{S}$, if $\widetilde{\mathcal{S}}^* \neq \mathcal{S}$, $R$ in (13) can be further improved via replacing $\mathcal{S}$ with the smaller subset $\widetilde{\mathcal{S}}^*$. In other words, there exists the users $i \in \mathcal{S}\backslash\widetilde{\mathcal{S}}^*$ who offload zero bits but are scheduled to unnecessarily create a VM at the server, resulting in waste of resources. Thereby, removing them from the offloading-user set and only allocating VMs to the full offloading users can further increase the sum offloading rate. By the above argument, the necessary condition for $\mathcal{S}$ being optimal of Problem (P1) is $\widetilde{\mathcal{S}}^* = \mathcal{S}$ in Problem (13). That is, all the scheduled users offload their maximum bits, or otherwise the given $\mathcal{S}$ is not optimal.

Thus, we re-define $R$ as the sum offloading rate achieved by $\widetilde{\mathcal{S}} = \mathcal{S}$ in Problem (13), i.e.,

$$R = \frac{\sum_{i\in\mathcal{S}}\omega_i r_i}{(1+d)^{|\mathcal{S}|-1} + \sum_{i\in\mathcal{S}}(a_i + b_i\gamma_i)r_i}. \tag{14}$$

We have the following proposition to identify whether $\mathcal{S}$ meets the necessarily optimal condition.

*Proposition 1:* $\widetilde{\mathcal{S}} = \mathcal{S}$ is the optimal solution of Problem (13) if and only if the given offloading-user set $\mathcal{S}$ satisfies

$$R \leq \min_{i\in\mathcal{S}}\left\{\frac{\omega_i}{a_i + b_i\gamma_i}\right\}. \tag{15}$$

*Proof:* See Appendix B. ■

To better understand (15), we multiply the term $t_e(1+d)^{1-|\mathcal{S}|}$ in both the numerator and denominator of $R$ and using the result that $\ell_i = t_e r_i(1+d)^{1-|\mathcal{S}|}$, $\forall i \in \mathcal{S}$, then $R$ in (14) can be rewritten as

$$R = \frac{\sum_{i\in\mathcal{S}}\omega_i\ell_i}{t_e + \sum_{i\in\mathcal{S}}(a_i+b_i\gamma_i)\ell_i}. \tag{16}$$

The numerator in (16) denotes the sum offloaded bits and the denominator denotes the total time and equals $T$. Then, $R$ in (16) can be physically interpreted as the sum offloading rate of the system with users of set $\mathcal{S}$. On the other hand, $\frac{\omega_i}{a_i+b_i\gamma_i}$ can be rewritten as $\frac{\omega_i\ell_i}{(a_i+b_i\gamma_i)\ell_i}$, where the numerator denotes the offloaded bits of user $i$ while the denominator denotes the transmission duration that includes both offloading and downloading time. Thus, $\frac{\omega_i}{a_i+b_i\gamma_i}$ can be regarded as the transmission rate of user $i$. Proposition 1 implies that the system offloading rate should be less than or equal to the minimum transmission rate among users in $\mathcal{S}$ if it solves Problem (P1).

*Remark 1:* If the given offloading-user set $\mathcal{S}$ violates condition (15), $R$ can be further improved by removing users with minimum transmission rate from the offloading-user set.

Let index $j$ denote the user with minimum transmission rate in set $\mathcal{S}$. Remark 1 can be illustrated using the following inequality:

$$\begin{aligned} \frac{\omega_j\ell_j}{(a_j+b_j\gamma_j)\ell_j} &< \frac{\sum_{i\in\mathcal{S}}\omega_i\ell_i}{t_e+\sum_{i\in\mathcal{S}}(a_i+b_i\gamma_i)\ell_i} \\ &< \frac{\sum_{i\in\mathcal{S}\backslash\{j\}}\omega_i\ell_i}{t_e+\sum_{i\in\mathcal{S}\backslash\{j\}}(a_i+b_i\gamma_i)\ell_i}, \end{aligned} \tag{17}$$

where the middle term in (17) is identical to $R$ and the right hand side of (17) is the sum offloading rate achieved after removing user $j$. (17) reveals that when the given offloading-user set $\mathcal{S}$ violates condition (15), there exists a slow user (i.e., user $j$) that is a bottleneck in the transmission process. Even without accounting for the parallel computing time, its transmission rate is already slower than the system offloading rate. Therefore, removing this bottleneck user can further improve the system offloading rate.

### B. Offloading-User Scheduling

Building on the results from the last subsection, we present in this subsection an efficient scheduling algorithm for computing the optimal offloading-user set. To this end, the variables $\{\ell_i\}$ and $t_e$ can be expressed in term of $\mathcal{S}$ when $\mathcal{S}$ meets the necessarily optimal condition (15). This simplifies Problem (P1) as a scheduling problem that finds the optimal offloading-user set under constraint (15):

$$\begin{aligned} \max_{\mathcal{S}\subseteq\mathcal{K}} \quad & R = \frac{\sum_{i\in\mathcal{S}}\omega_i r_i}{(1+d)^{|\mathcal{S}|-1} + \sum_{i\in\mathcal{S}}(a_i+b_i\gamma_i)r_i} \\ \text{s.t.} \quad & R \leq \min_{i\in\mathcal{S}}\left\{\frac{\omega_i}{a_i+b_i\gamma_i}\right\}. \end{aligned} \tag{18}$$

The problem can be further reduced to an unconstrained optimization problem using the following useful result.

*Proposition 2:* Constraint (15) can be removed from Problem (18) without loss of optimality.

*Proof:* See Appendix C. ■

Using Proposition 2, Problem (18) can be safely relaxed into the following non-constrained optimization problem:

$$\max_{\mathcal{S}\subseteq\mathcal{K}} \quad R = \frac{\sum_{i\in\mathcal{S}}\omega_i r_i}{(1+d)^{|\mathcal{S}|-1}+\sum_{i\in\mathcal{S}}(a_i+b_i\gamma_i)r_i}. \tag{19}$$

However, with the non-convex term $(1+d)^{|\mathcal{S}|-1}$ in the denominator of $R$, Problem (19) is still challenging to solve. To tackle this difficulty, we fix $|\mathcal{S}| = m$, with $m = 1, \cdots, K$. For a given $m$, since term $(1+d)^{|\mathcal{S}|-1}$ becomes a constant, Problem (19) is reduced to a *mixed-integer linear fractional programming* problem. We solve Problem (19) by decomposing it into master-and-slave problems without loss of the optimality. The slave problem is determining the optimal offloading-user set using the Dinkelbach method [32] for a given number of scheduled users $m$. Then the master problem is obtaining the optimal value of $m$, denoted as $m^*$, by a simple search. The detailed solutions of the decomposed problems are presented in the sequel, which yield Algorithm 1 for computing the optimal scheduled-user set $\mathcal{S}^*$.

*1) Optimal Scheduling for a Given Number of Scheduled Users:* In this section, we solve Problem (P1) conditioned on a given number of offloading users $m$, i.e., $|\mathcal{S}| = m$. To this end, we introduce a set of binary variables $\mathbf{x} = [x_1, \cdots, x_K]$, where $x_i = 1$ means that user $i$ is scheduled (i.e., $i \in \mathcal{S}$), and $x_i = 0$ otherwise. Then, using the binary variables and conditioned on $|\mathcal{S}| = m$, Problem (P1) can be transformed into a combinatorial optimization problem as

(Slave Problem)

$$\max_{\mathbf{x}} \quad R_m = \frac{\sum_{i=1}^K x_i \omega_i r_i}{(1+d)^{m-1} + \sum_{i=1}^K x_i(a_i + b_i\gamma_i) r_i} = \frac{N(\mathbf{x})}{D(\mathbf{x})}$$
$$\text{s.t.} \quad \sum_{i=1}^K x_i = m, \quad x_i \in \{0,1\}, \quad i = 1, \cdots, K, \tag{20}$$

where $N(\mathbf{x}) \triangleq \sum_{i=1}^K x_i \omega_i r_i$ and $D(\mathbf{x}) \triangleq (1+d)^{m-1} + \sum_{i=1}^K x_i(a_i + b_i\gamma_i) r_i$. Let $R_m^*$ denotes the maximum conditional sum offloading rate from solving the slave problem. For ease of notation, we define the feasible set for Problem (20) as $\mathcal{F}_m \triangleq \{\mathbf{x} | \sum_{i=1}^K x_i = m$ and $x_i \in \{0,1\}, i = 1, \cdots, K\}$. Since the objective function has a fractional form, the problem can be solved by *non-linear fractional programming*. To this end, define a function $g(\cdot)$ of the conditional rate $R_m$ by an optimization problem in a substrative form:

$$g(R_m) = \max_{\mathbf{x} \in \mathcal{F}_m} \left[ N(\mathbf{x}) - D(\mathbf{x}) R_m \right]. \tag{21}$$

Let $\mathbf{x}^*$ be an optimal solution of Problem (20). We have the following property.

*Lemma 2:* The maximum conditional sum offloading rate $R_m^*$ that solves Problem (20) can be achieved if and only if

$$\begin{aligned} g(R_m^*) &= \max_{\mathbf{x} \in \mathcal{F}_m} \left[ N(\mathbf{x}) - D(\mathbf{x}) R_m^* \right] \\ &= N(\mathbf{x}^*) - D(\mathbf{x}^*) R_m^* = 0. \end{aligned} \tag{22}$$

*Proof:* See Appendix D. ■

Lemma 2 reveals the fact that the targeted fractional-form problem in (20) shares the solution $\mathbf{x}^*$ as the subtractive-form problem in (21) when $R_m = R_m^*$. This provides an indirect method for solving the former using an iterative algorithm derived in the sequel, in which the derived condition $g(R_m) = 0$ is applied to checking the optimal convergence.

Based on Dinkelbach method [32], we propose an iterative algorithm to obtain $R_m^*$ in (22), thereby solving the slave problem in (20). Specifically, we concern the optimal solution to the subtractive-form Problem (21) for a given $R_m$:

$$g(R_m) = \max_{\mathbf{x} \in \mathcal{F}_m} \left\{ \sum_{i=1}^K x_i r_i \left[ \omega_i - R_m(a_i + b_i\gamma_i) \right] - R_m (1+d)^{m-1} \right\}. \tag{23}$$

To facilitate exposition, we can rewrite the expression of $g(R_m)$ as

$$g(R_m) = \max_{\mathbf{x} \in \mathcal{F}_m} \left\{ \frac{\sum_{i=1}^K x_i \psi_i(R_m)}{t_e (1+d)^{1-m}} - R_m (1+d)^{m-1} \right\}, \tag{24}$$

with

$$\begin{aligned} \psi_i(R_m) &= x_i r_i \left[ \omega_i - R_m(a_i + b_i\gamma_i) \right] t_e (1+d)^{1-m} \\ &= x_i \left[ \omega_i \ell_i - R_m(a_i + b_i\gamma_i)\ell_i \right] \\ &= \omega_i \ell_i - R_m (t_i^u + t_i^d), \end{aligned} \tag{25}$$

where the second equality is derived by the result that $\ell_i = t_e r_i (1+d)^{1-m}$ and the last equality is obtained by substituting (1) and (2).

*Remark 2 (Per-user Revenue):* The variable $\psi_i(R_m)$ can be interpreted as the net revenue of scheduling user $i$ as explained shortly. With the system offloading rate $R_m$, $R_m(t_i^u + t_i^d)$ represents the expected number of user bits that can be computed successfully by offloading and result downloading over the duration of $(t_i^u + t_i^d)$. By allocating the time to user $i$ for offloading and downloading, the weighted number of actual computed bits is $w_i \ell_i$. Therefore, the difference between expected and actual bits, $\psi_i(R_m)$, measures the net system revenue obtained from scheduling user $i$.

- Step 1: Based on Remark 2, the objective of the optimization in (23) can be interpreted as one for maximizing the total system revenue. It follows that the optimal solution, denoted as $\mathbf{x}^*$, is to select $m$ users having the largest per-user revenue:
$$x_i^* = \begin{cases} 1, & \text{if } \psi_i(R_m) \text{ is one of the } m \text{ largest}, \\ 0, & \text{otherwise}, \end{cases} \tag{26}$$
with $i = 1, \cdots, K$, where $\psi_i(R_m)$ is defined in (25).
- Step 2: Given $\mathbf{x}^*$ computed in Step 1, the sum offloading rate $R_m$ can be updated as
$$R_m = \frac{N(\mathbf{x}^*)}{D(\mathbf{x}^*)}, \tag{27}$$
where $N(\cdot)$ and $D(\cdot)$ are given in (20). Then the per-user revenues $\{\psi_i(R_m)\}$ are updated using the new value of $R_m$.

Based on the Dinkelbach method, the above two steps are iterated till $g(R_m) = 0$. Since this is the optimality condition according Lemma 2, the convergence of the iteration yields the maximum $R_m^*$ and the corresponding $m$ scheduled users $\mathcal{S}^*(m) = \{i \mid x_i^* = 1\}$. It can be proved that the convergence rate is *superlinear* (see e.g., [32]).

*2) Finding the Optimal Number of Scheduled Users:* With the slave problem in (20) solved in the preceding sub-section, the master problem is to optimize $m$:

(Master Problem)

$$\max_{1 \le m \le K} R_m^* = \frac{\sum_{i \in \mathcal{S}^*(m)} \omega_i r_i}{(1+d)^{m-1} + \sum_{i \in \mathcal{S}^*(m)} (a_i + b_i\gamma_i) r_i}. \tag{28}$$

To solve the problem, an intelligent search for $m^*$ over $\{1, 2, \cdots, K\}$ seems to be difficult for the reason that $\{R_m^*\}$

**Algorithm 1** Iterative User Scheduling Algorithm Based on Dinkelbach Method

```
for $m = 1, \cdots, K$ do
  initialize $R_m = 0$.
  repeat
    For a given $R_m$, compute $\mathbf{x}^*$ according to (26);
    Update $R_m = \frac{N(\mathbf{x}^*)}{D(\mathbf{x}^*)}$;
  until $g(R_m) = 0$.
  Return $R_m^* = R_m$, $\mathbf{x}_m^* = \mathbf{x}^*$.
end for
Return $m^* = \arg\max_{1 \le m \le K} \{R_m^*\}$, $R^* = R_{m^*}^*$ and
  $\mathcal{S}^* = \{i \mid x_i^* = 1, i \in \mathcal{K}\}$.
Output: $R^*$ and $\mathcal{S}^*$.
```

is not a monotone sequence, which arises from the fact that scheduling more users increases multiplexing gain in parallel computing but causes stronger I/O interference and vice versa. Due to the lack of monotonicity, we resort to enumerating all possible values of $m$ from 1 to $K$ to find $m^*$. The complexity of the exhaustive search is reasonable as it scales only linearly with the total number of users $K$.

*3) Overall Algorithm and Its Complexity:* The overall algorithm for solving the scheduling problem in (19) is shown in Algorithm 1 which combines the iterative algorithm for solving the slave problem and the exhaustive search for solving in the master problem which are designed in the preceding sub-sections.

The complexity of the overall algorithm is discussed as follows. The iterative algorithm for solving the slave problem using Dinkelbach method has complexity upper bounded by $O(\log K)$ [33]. Solving the master problem repeats at most $K$ runs of the iterative algorithms. Therefore, the worst-case complexity of the overall algorithm is $O(K \log K)$.

### *C. Special Cases*

Several special cases are considered to derive additional insights into the optimal multiuser offloading. For simplicity, the users' weights are assumed to be uniform, i.e., $\omega_i = 1$, $\forall i \in \mathcal{K}$.

*1) Homogenous Users and Channels:* Consider the special case where users are homogeneous in task types and channels such that their offloading parameter sets $\{r_i, a_i, b_i, \gamma_i\}$ are identical. Then Problem (19) reduces to the simple problem of determining the number of offloading users. The sum offloading rate $R$ can be simplified as $R = \frac{mr}{(1+d)^{m-1}+mr(a+b\gamma)}$. By letting $\frac{dR}{dm} = 0$, the optimal number of scheduled users is obtained as

$$m^* \approx \left[\frac{1}{\ln(1+d)}\right]_1^K, \tag{29}$$

where $[x]_1^K = \max\{\min\{x, K\}, 1\}$ restricts the $m^*$ in the range from 1 to $K$. The result shows that for this special case, the optimal number of offloading users (or equivalently the optimal number of VMs in parallel computing) only depends on the I/O-interference parameter $d$ in the parallel-computing model in (3).

*2) Homogeneous Transmission Rates:* Relaxing the assumption of homogeneous task types in the preceding case leads to the current case of homogeneous transmission rates due to channel homogeneity, corresponding to $\frac{1}{a_1+b_1\gamma_1} = \frac{1}{a_2+b_2\gamma_2} = \cdots = \frac{1}{a_K+b_K\gamma_K}$. Due to variation in task types, users have different computation-service rate specified by the parameter $\{r_i\}$ in the computation model in (3). We obtain for the current case the optimal offloading-user set as shown in the following proposition.

*Proposition 3 (Homogeneous Transmission Rates):* Consider the special case of homogeneous transmission rates $\frac{1}{a_1+b_1\gamma_1} = \frac{1}{a_2+b_2\gamma_2} = \cdots = \frac{1}{a_K+b_K\gamma_K}$. Without loss of generality, assume the computation-service rates $r_1 \ge r_2 \ge \ldots \ge r_K$. Let $n_0$ denote the largest user index $n$ that satisfies $r_n \ge d\sum_{i=0}^{n-1} r_i$ with $r_0 = 0$. Then, the optimal scheduled-user set $\mathcal{S}^*$ that solves Problem (18) is given by

$$\mathcal{S}^* = \{i \,|1 \le i \le n_0\}. \tag{30}$$

*Proof:* Please see Appendix E. ■

*Remark 3 (To schedule or not?):* The computation-service rate $r_n$ can be seen as the gain of scheduling user $n$ while $d\sum_{i=0}^{n-1} r_i$ represents the performance degradation imposed on the preceding scheduled users (i.e., user 1 to $(n-1)$). As long as $r_n \ge d\sum_{i=0}^{n-1} r_i$ is met, the gain of scheduling user $n$ outweighs its cost and thus it is worthwhile to schedule user $n$ for improving the sum offloading rate.

*Remark 4 (Optimal Scheduling):* Proposition 3 shows that the optimal scheduling policy is to select $n_0$ users with the best computing rates and the index $n_0$ can be obtained by adopting greedy approach that selects users in descending order of the computation-service rate (i.e., $r_i$) until the condition $r_n \ge d\sum_{i=0}^{n-1} r_i$ becomes invalid.

*3) No I/O Interference:* Consider the ideal case without I/O interference, namely $d = 0$ in (3). This case corresponds to sufficient I/O resources at the AP. We can show that for this case the optimal offloading-user set has a threshold based structure where the threshold is determined by transmission rates. The details are given in the following proposition.

*Proposition 4:* Without loss of generality, assume the transmission rates follow the descending order: $\frac{1}{a_1+b_1\gamma_1} > \cdots > \frac{1}{a_K+b_K\gamma_K}$. Let $m_0$, with $1 \le m_0 \le K$, denote the largest user index that meets

$$\frac{1}{a_{m_0}+b_{m_0}\gamma_{m_0}} \ge \frac{\sum_{i=1}^{m_0} r_i}{1+\sum_{i=1}^{m_0}(a_i+b_i\gamma_i)r_i}. \tag{31}$$

The optimal scheduled-user set $\mathcal{S}^*$ that solves Problem (18) is given by

$$\mathcal{S}^* = \{i \,|1 \le i \le m_0\}. \tag{32}$$

The proof is similar to Proposition 3 and thus omitted. It can observe that condition (31) is a simplified version of (15) by setting $d = 0$. However, it is important to note that the former provides a sufficient condition of the optimal offloading-user set in (32) for the current special case while the latter only provides a necessary condition for optimal scheduling in the general case. Last, similar to the preceding case, the index $m_0$ can be obtained via a greedy method.

## IV. Sum Mobile Energy Minimization

In this section, we attempt to solve Problem (P2) of minimizing sum-energy consumption over mobiles in the multiuser-offloading process. First, the feasibility region of the problem is analyzed. Then Problem (P2) is converted into an equivalent problem, which facilitates the design of an algorithm for finding a sub-optimal solution.

### *A. Feasibility Analysis*

The feasible region of Problem (P2) is non-empty if the latency constraint $T$ is larger than the minimum required time for computing all offloaded tasks, denoted as $T_{\min}$. To find $T_{\min}$ is equivalent to solving the following *latency minimization problem*:

$$\text{(P3)}: \quad \min_{\mathcal{S}\subseteq\mathcal{K},\{\ell_i\},t_e,T} \quad T \quad \text{s.t.} \quad (10\text{b})-(10\text{c})$$

The solution of Problem (P3) can be obtained as shown in the following proposition.

*Proposition 5:* The minimum computation time $T_{\min}$ that solves Problem (P3) is the root of the following equation with the variable $T$:

$$\sum_{i\in\mathcal{K}} (a_i + b_i\gamma_i)\, L_i^{\min} + \max_{i\in\mathcal{K}} \left\{ \frac{L_i^{\min}}{r_i(1+d)^{1-N(T)}} \right\} = T, \quad (33)$$

where $N(T) \triangleq \sum_{i\in\mathcal{K}} \chi\{L_i^{\min} > 0\}$ with $L_i^{\min}$ being the minimum offloaded data size under the latency constraint, and $\chi\{\cdot\}$ is an indicator function that outputs 1 when an event occurs and 0 otherwise.

*Proof:* Please see Appendix F. ■

The left hand side of (33) represents the time required for completing the minimum offloaded data with sizes of $\left\{L_i^{\min}\right\}$ with $L_i^{\min} = \left[L_i - \frac{Tf_i}{c_i}\right]^+$ [see (8)]. In this expression, the first term is the time used for offloading and downloading, and the second term is the time for parallel computing, which is dominated by the task with the maximum execution time. Note that minimum offloading with sizes $\left\{L_i^{\min}\right\}$ requires full utilization of local-computing capacities, i.e., the local computing time is at its maximum extended to $t_i^{\text{loc}} = T$, for all $i$. It follows that $T_{\min}$ occurs when the time for local computing and MEC are both equal to $T$. Since the left hand side of (33) is non-differentiable but monotonically decreasing with $T$, $T_{\min}$ can be easily found by simple bisection search. Then $T \geq T_{\min}$ yields the condition of nonempty feasibility region for Problem (P2).

### *B. Problem Transformation*

Problem (P2) can be transformed into an equivalent problem whose solution facilitates scheduling design. A close observation of Problem (P2) reveals that, when the $i$-th minimum offloaded data size $L_i^{\min} > 0$, it indicates that task $i$ cannot be computed locally within the duration $T$ and a fraction with at least $L_i^{\min}$ bits has to be offloaded. Therefore, $L_i^{\min} > 0$ means that user $i$ needs offloading. On the other hand, the condition $\theta_i > 0$ corresponds to the case where task offloading consumes more energy than local computing. For the purpose of energy-saving, users with $\theta_i > 0$ should offload the minimum of $L_i^{\min}$ bits. Based on if none, one, or both of the above two conditions holds, we can divide $K$ users into four disjoint subsets as follows:

$$\begin{aligned}
\mathcal{M}_0 &= \{i \mid L_i^{\min} > 0 \text{ and } \ \theta_i > 0\},\\
\mathcal{M}_1 &= \{i \mid L_i^{\min} > 0 \text{ and } \ \theta_i < 0\},\\
\mathcal{N}_0 &= \{i \mid L_i^{\min} = 0 \text{ and } \ \theta_i > 0\},\\
\mathcal{N}_1 &= \{i \mid L_i^{\min} = 0 \text{ and } \ \theta_i < 0\}.
\end{aligned}$$

As a result, $\mathcal{M}_0$ and $\mathcal{M}_1$ are the sets of users requiring offloading under the latency constraint. To save energy, users in $\mathcal{M}_0$ should offload minimum data $\ell_i^* = L_i^{\min}$ while users in $\mathcal{N}_0$ should perform local computing only (i.e., $\ell_i^* = 0$). The other sets $\mathcal{M}_1$ and $\mathcal{N}_1$ are the sets of users who favour offloading since it is more energy-efficient than local computing. Furthermore, users in $\mathcal{M}_1$ have to offload at least $L_i^{\min}$ bits under the latency constraint. In summary, for sum energy minimization, the optimal scheduling policy should schedule all users in $\mathcal{M}_0$ with minimum offloading, all in $\mathcal{M}_1$ to offload at least $L_i^{\min}$ bits, none from $\mathcal{N}_0$, a subset of users from $\mathcal{N}_1$ with nonzero offloading.

Based on the above discussion and by denoting an arbitrary subset of $\mathcal{N}_1$ as $\mathcal{S}_1$, Problem (P2) can be transformed into the following equivalent problem:

$$\text{(P4)}: \quad \min_{\mathcal{S}_1\subseteq\mathcal{N}_1,\{\ell_i\},t_e,} \quad \sum_{i\in\mathcal{S}_1\cup\mathcal{M}_1} \theta_i \ell_i \qquad (34)$$

$$\text{s.t.} \quad \sum_{i\in\mathcal{S}_1\cup\mathcal{M}_1} \ell_i\,(a_i + b_i\gamma_i) + t_e \leq \widetilde{T}, \qquad (35)$$

$$L_i^{\min} \leq \ell_i \leq \min\left\{L_i, t_e r_i\,(1+d)^{1-|\mathcal{M}|-|\mathcal{S}_1|}\right\}, \quad \forall i \in \mathcal{M}_1 \qquad (36)$$

$$0 \leq \ell_i \leq \min\left\{L_i, t_e r_i\,(1+d)^{1-|\mathcal{M}|-|\mathcal{S}_1|}\right\}, \quad \forall i \in \mathcal{S}_1 \qquad (37)$$

$$t_e \geq \max_{i\in\mathcal{M}_0} \left\{ \frac{L_i^{\min}}{r_i(1+d)^{1-|\mathcal{M}|-|\mathcal{S}_1|}} \right\}. \qquad (38)$$

where $\widetilde{T} \triangleq T - \sum_{i\in\mathcal{M}_0} L_i^{\min}(a_i + b_i\gamma_i)$ and $|\mathcal{M}| \triangleq |\mathcal{M}_0| + |\mathcal{M}_1|$. Note that $\{\theta_i < 0 | i \in \mathcal{S}_1 \cup \mathcal{M}_1\}$, corresponding to the fact that offloading saves mobile energy. It follows that the minimization in Problem (P4) attempts to maximize offloading for users from $\mathcal{N}_1$ and $\mathcal{M}_1$. Last, given $\mathcal{S}_1$, the total number of offloading users is obtained as $|\mathcal{M}_0| + |\mathcal{M}_1| + |\mathcal{S}_1|$. The constraint in (38) is derived from the constraint (8) for $i \in \mathcal{M}_0$, which ensures that $t_e$ is no less than the minimum required time for computing any task in $\mathcal{M}_0$.

Problem (P4) is a mixed integer programming problem. Its solution potentially requires an exhaustive search over all possible user subsets $\{\mathcal{S}_1\}$, resulting in complexity exponentially increasing with number of users. For this reason, we find a close-to-optimal solution by developing a low-complexity suboptimal algorithm in the next subsection.

### *C. Suboptimal Scheduling Algorithm*

The tractability of algorithmic design relies on applying a set of sub-optimal rules on offloading so as to ensure that the latency requirement can be met. To this end, we make the observation that, without considering the latency constraint in (35), the optimal offloading policy for solving Problem (P4) is one that all the users in $\mathcal{M}_1$ and $\mathcal{N}_1$ offload data with maximum sizes, i.e., $\{\ell_i = L_i | i \in \mathcal{M}_1 \cup \mathcal{N}_1\}$. To rein in the latency, the following rules are proposed to simplify the design problem:

1) The users in $\mathcal{N}_1$ are constrained to adopt binary offloading scheme, i.e., $\ell_i = L_i$ if $i \in \mathcal{S}_1$ and $\ell_i = 0$ if $i \in \mathcal{N}_1 \backslash \mathcal{S}_1$. The incentive of considering binary policy is to make the process of offloading-decision making as simple and efficient as possible.
2) If $\mathcal{S}_1 \neq \emptyset$, the users in $\mathcal{M}_1$ offload their data with the maximum sizes $L_i$. The rule is motivated by the observation that increasing the number of simultaneous VMs incurs higher server's operational cost. Applying this rule can maximize the utilization of each subscribed VM resource so as to reduce the number of VMs for minimizing the cost, while ensuring a high system performance.
3) By applying the rules in 1) and 2), if the latency requirement (35) is violated, we first remove the users in $\mathcal{S}_1$ for reducing the total latency. When $\mathcal{S}_1 = \emptyset$ and the reduced total latency still violates the requirement, we proceed to reduce the offloading bits from users in $\mathcal{M}_1$ for further latency reduction.

Based on the above rules, a sub-optimal algorithm for solving Problem (P4) is designed as follows. To this end, we define a function of total offloaded-computation latency for scheduled users as follows

$$\mathcal{D}(\mathcal{S}_1) = t_e(\mathcal{S}_1) + \sum_{i \in \mathcal{S}_1 \cup \mathcal{M}_1} L_i(a_i + b_i \gamma_i) + \sum_{i \in \mathcal{M}_0} L_i^{\min}(a_i + b_i \gamma_i), \quad (39)$$

with

$$t_e(\mathcal{S}_1) = \frac{\max\left\{\max_{i \in \mathcal{S}_1 \cup \mathcal{M}_1}\left\{\frac{L_i}{r_i}\right\}, \max_{i \in \mathcal{M}_0}\left\{\frac{L_i^{\min}}{r_i}\right\}\right\}}{(1+d)^{1-|\mathcal{M}|-|\mathcal{S}_1|}}, \quad (40)$$

Since all users in $\mathcal{M}_1 \cup \mathcal{M}_0$ should be all scheduled as discussed earlier, the function has only one variable $\mathcal{S}_1$. The variable $t_e(\mathcal{S}_1)$ in (40) represents the corresponding minimum parallel computing time, which is derived using (36) to (38).

**Algorithm 2** Suboptimal Algorithm for solving Problem (P4)

```
Input: $T \geq T_{\min}$.
1: Set $\{\ell_i = 0 | i \in \mathcal{N}_0\}$ and $\{\ell_i = L_i^{\min} | i \in \mathcal{M}_0\}$.
2: initialize $\{\ell_i = L_i | i \in \mathcal{M}_1 \cup \mathcal{N}_1\}$.
3: if $T \in [\mathcal{D}(\mathcal{N}_1), +\infty)$ then
4:     Return $\mathcal{S} = \mathcal{M}_0 \cup \mathcal{M}_1 \cup \mathcal{N}_1$, $\{\ell_i\}$, and $t_e = t_e(\mathcal{N}_1)$.
5: else if $T \in [\mathcal{D}(\emptyset), \mathcal{D}(\mathcal{N}_1))$ then
6:     initialize $\mathcal{S}_1 = \mathcal{N}_1$.
7:     repeat
8:         Let $j = \arg\min_{i \in \mathcal{S}_1}\left\{\frac{-\theta_i}{a_i + b_i\gamma_i}\right\}$. Update $\mathcal{S}_1 = \mathcal{S}_1 \backslash \{j\}$
           and $\ell_j = 0$.
9:     until $\mathcal{D}(\mathcal{S}_1) \leq T$.
10:    Return $\mathcal{S} = \mathcal{M}_0 \cup \mathcal{M}_1 \cup \mathcal{S}_1$, $\{\ell_i\}$, and $t_e = t_e(\mathcal{S}_1)$.
11: else
12:    Solve Problem (41) and obtain its optimal solution
       $\{\ell'_i\}_{i \in \mathcal{M}_1}$ and $t'_e$.
13:    Set $\{\ell_i = 0 | i \in \mathcal{N}_1\}$, $\{\ell_i = \ell'_i | i \in \mathcal{M}_1\}$, and $t_e = t'_e$.
14:    Return $\mathcal{S} = \mathcal{M}_0 \cup \mathcal{M}_1$, $\{\ell_i\}$, and $t_e$.
15: end if
Output: $\mathcal{S}$, $\{\ell_i\}$, $t_e$.
```

By applying the aforementioned rules and using the definition in (39), the suboptimal algorithm for solving Problem (P4) is presented in Algorithm 2 with the key steps described as follows. Specifically, we consider three scenarios of the latency constraint $T$. First, if $T \in [\mathcal{D}(\mathcal{N}_1), +\infty)$, the latency constraint in (35) is met by the offloading policy from Steps 1-2, which is thus optimal without the need of further modification. Second, if $T \in [\mathcal{D}(\emptyset), \mathcal{D}(\mathcal{N}_1))$, based on the said rules, we remove the members in $\mathcal{S}_1$ incrementally to reduce the total latency by the following iterative procedure. We initialize $\mathcal{S}_1$ as $\mathcal{N}_1$ and continue to remove users from $\mathcal{S}_1$ until $\mathcal{D}(\mathcal{S}_1) \leq T$ is met. In each removal, we take away one user using greedy strategy, i.e., selecting the user in $\mathcal{S}_1$ with the minimum energy efficiency (i.e., $\frac{-\theta_i}{a_i + b_i\gamma_i}$). Last, if the given $T$ is still smaller than $\mathcal{D}(\emptyset)$, we begin to reduce the offloaded bits of users in $\mathcal{M}_1$ for further latency reduction. Fortunately, when $\mathcal{S}_1 = \emptyset$, Problem (P4) is reduced to the problem of determining $\ell_i$'s in $\mathcal{M}_1$ as follow:

$$\begin{aligned} \min_{\{\ell_i\},\ t_e} \quad & \sum_{i \in \mathcal{M}_1} \theta_i \ell_i \\ \text{s.t.} \quad & \sum_{i \in \mathcal{M}_1} \ell_i(a_i + b_i\gamma_i) + t_e \leq \widetilde{T}, \\ & L_i^{\min} \leq \ell_i \leq \min\left\{L_i, t_e r_i (1+d)^{1-|\mathcal{M}|}\right\}, \ \forall i \in \mathcal{M}_1, \\ & t_e \geq \max_{i \in \mathcal{M}_0}\left\{\frac{L_i^{\min}}{r_i(1+d)^{1-|\mathcal{M}|}}\right\}. \end{aligned} \quad (41)$$

which is an LP problem and can be solved efficiently by the LP solver. The complexity of Algorithm 2 is dominated by solving the LP problem in Step 12, which is $\mathcal{O}((|\mathcal{M}_1|)^{3.5})$.

*Remark 5 (Power Control):* Our framework can be extended to include the issue of power control. Specifically, for the offloading rate maximization problem (P1), it is straightforward that the optimal power control of users is to transmit with their peak power budget as the goal is to maximize the offloading rate. For the sum-energy minimization problem (P2), power control of users will significantly complicate the optimization problem. However, it is easy to prove that problem (P2) becomes a convex problem if the offloading-user set $\mathcal{S}$ is given. Then we can extend the proposed Algorithm 2 to solve the power control included problem by two steps: First, for given offloading-user set $\mathcal{S}$, the standard convex optimization methods can be used to solve the problem. Then the proposed Algorithm 2 can be extended to find the offloading-user set $\mathcal{S}$.

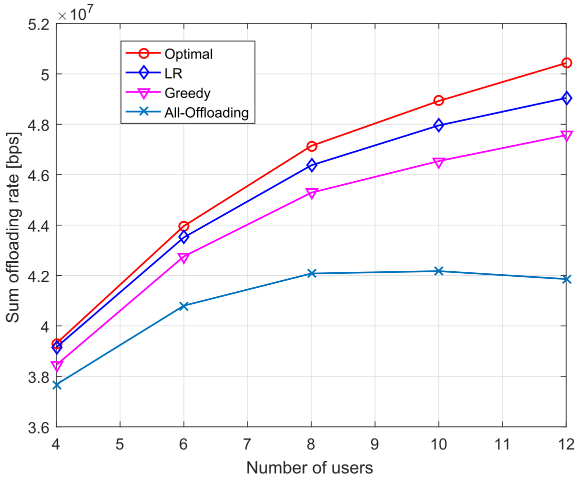


Fig. 2. Sum offloading rate versus $K$.

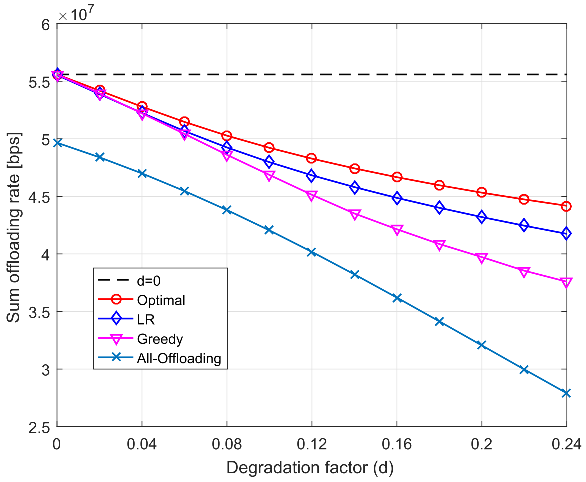


Fig. 3. Sum offloading rate versus $d$.

## V. Simulation Results

In this section, we provide simulation results to evaluate the proposed algorithms. The parameters are set as follows, unless otherwise stated. We set $T = 35$ ms and $\omega_i = 1, \forall i \in \mathcal{K}$. For each user $i$, we set the uplink and downlink transmission rates $a_i^{-1}$ and $b_i^{-1}$ uniformly distributed in $[100, 150]$ Mbps and $[150, 200]$ Mbps, respectively. The computation-service rate $r_i$ follows uniform distribution over $\left[1 \times 10^7, 2 \times 10^7\right]$ bits/sec. In addition, we set the ratio of output/input data $\gamma_i = 10^{-x}$, where $x$ is uniformly distributed over $[0.5, 1.5]$. All random variables are independent for each user and the simulation results are obtained by averaging over 500 realizations.

### *A. Offloading Rate Maximization*

For performance comparison, we introduce three benchmark algorithms in the following.

1) *All-Offloading:* All the users are scheduled to offload, i.e., $\mathcal{S} = \mathcal{K}$.
2) *Greedy:* $\mathcal{S}$ is obtained through a greedy method, i.e., selecting users in the descending order of the transmission rate (i.e., $\frac{\omega_i}{a_i + b_i \gamma_i}$) until condition (15) is invalid.
3) *Linear Programming Relaxation (LR):* $\mathcal{S}$ is obtained by solving $K$ slave problems in (20) using linear programming relaxation [34].

Note that the three benchmarks are used to find the offloading-user set, then the rest problem is reduced to an LP that can be solved efficiently.

In Fig. 2, we compare the sum offloading rate performance of different algorithms when the number of users $K$ varies from 4 to 12, where $d$ is set as 0.1. First, we can see that the sum offloading rate is increasing with $K$ for the optimal, LR and greedy algorithms, while for the scheme that all users offload, it grows slowly when $K \leq 10$ and begins to decrease afterwards. This is because the former three algorithms have more flexible user-scheduling schemes to balance the degradation impact caused by I/O interference and thus have superior system performance. In contrast, the last algorithm with no control on the number of offloading users, will suffer more severe performance degradation as $K$ increases. Besides, it can be observed that the optimal algorithm outperforms the benchmark algorithms especially when $K$ is large. For instance, when $K = 12$, the optimal algorithm obtains about $3\%$, $6\%$, and $20\%$ performance improvements over the three benchmarks respectively.

In Fig. 3, we illustrate the relationship between the degradation factor $d$ and the sum offloading rate performance, where $K = 10$. As expected, the sum offloading rate is decreasing with $d$ in all considered algorithms while the descending rate of the optimal algorithm is the slowest. This indicates that our proposed algorithm has the best performance resistance against the I/O-interference effect. One observes that, the performance of LR and greedy algorithms is close-to-optimal when $d$ is small. This coincides with the result of special case 4) in Section III-C that when the degradation factor $d$ is zero, the optimal solution can be obtained by the greedy approaches. On the other hand, the line of $d = 0$ can been seen as the sum offloading rate of the conventional case without considering the I/O interference issue. Its performance gap with the optimal algorithm can be interpreted as the overestimation of the system performance builded on the optimistic assumption of no I/O interference.

### *B. Energy Minimization*

For measuring the energy consumption, we set $\kappa_i = 10^{-28}$ [10] and $p_i = 0.1$ W. The data size of task, the required number of CPU cycles per bit, and the local-computing speed follow uniform distribution with $L_i \in [50, 100]$ KB, $c_i \in [500, 1000]$ cycles/bit, and $f_i \in [2 \times 10^8, 6 \times 10^8]$ cycles/sec, respectively.

To evaluate the proposed suboptimal algorithm, we present the performance of all-offloading scheme mentioned in the preceding subsection and the optimal performance that is obtained via decomposing Problem (P2) into $K$ mixed-integer linear programming (MILP) subproblems and each subproblem is optimally solved using the MILP solver. The MILP solver

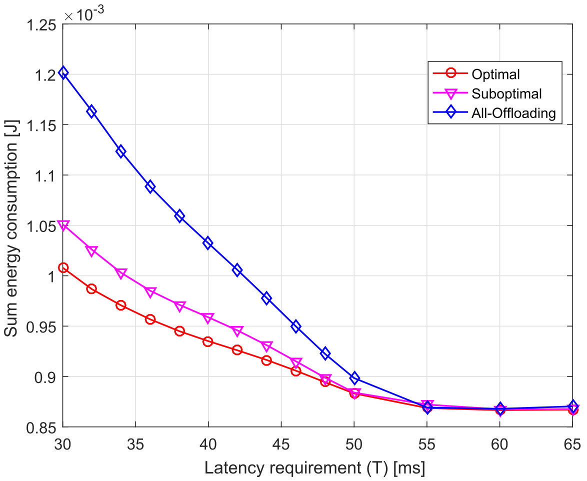


Fig. 4. Sum energy consumption versus $T$.

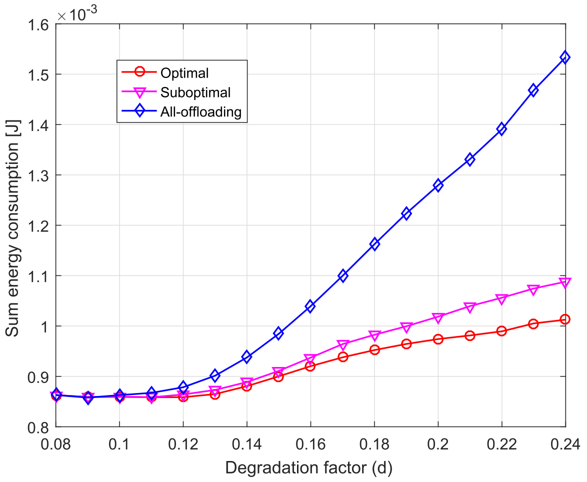


Fig. 5. Sum energy consumption versus $d$.

implements Branch-and-Bound based algorithms which have complexities exponentially increasing with the size of $K$.

In Fig. 4, we investigate the impact of latency constraint $T$ on the sum energy consumption, where $K = 10$ and $d = 0.2$. First, we see that as the maximum tolerance latency $T$ increases, the sum energy consumption decreases. This is because a more relaxed latency requirement facilitates larger task fraction to offload and consequently saves more energy. However, when $T$ is sufficiently large, the sum energy consumption achieves its minimum and becomes irrespective of $T$ since the optimal offloading scheme always meets the latency constraint in (35). Next, it can be observed that the suboptimal algorithm has superior performance compared to the all-offloading scheme especially when $T$ is small. In particular, when $T$ is 30 ms, the suboptimal algorithm obtains about 14% energy-saving compared to the all-offloading algorithm.

In Fig. 4, we present the performance of the proposed suboptimal algorithm versus the degradation factor $d$, where $K = 10$. We observe that compared with the dramatic increase of sum energy consumption with $d$ in the all-offloading algorithm, the sum energy consumption in the proposed suboptimal algorithm grows at a much slower rate, which is close to the line of the optimal performance.

## VI. Conclusions

In this paper, we investigate joint radio-and-computation resource allocation in a multiuser MEC system, where the computation interference issue has been considered. We formulate two optimization problems: sum offloading rate maximization and sum energy minimization. To address rate maximization, we first solve the optimal offloaded data size and computation time allocation for any given offloading-user set that meets the necessarily optimal condition. Then we develop an optimal algorithm based on Dinkelbach method to find the optimal offloading-user set. For solving energy minimization, we transform the original problem into an equivalent one that facilitates the scheduling design and propose an algorithm to find a sub-optimal solution. Simulation results demonstrate that our proposed algorithms achieve superior performance gain compared with the benchmark algorithms.

## Appendix

### A. Proof of Lemma 1

To prove this lemma, it is sufficient to show that for any given $\mathcal{S}$, one of the constraints $\ell_i \le t_e r_i (1+d)^{1-|\mathcal{S}|}$ or $\ell_i \ge 0$ must be active at the optimal $\ell_i^*$, $\forall i \in \mathcal{S}$.

For a given $\mathcal{S}$, since Problem (P1) is an LP problem with a bounded feasible region, there exists an optimal solution located at a vertex (i.e., extreme point) [35]. Denote $\overline{x} = [\ell_1, \ell_2, \dots, \ell_{|\mathcal{S}|}, t_e]^T \in \mathbb{R}^{|\mathcal{S}|+1}$ as the vertex that is optimal. By the vertex definition, there are $|\mathcal{S}|+1$ linearly independent active constraints at $\overline{x}$. First, it is easy to check that (5b) should be active at $\overline{x}$. Moreover, the pair of constraints $\ell_i \le t_e r_i (1+d)^{1-|\mathcal{S}|}$ and $\ell_i \ge 0$ for each $i \in \mathcal{S}$ should not be active or inactive simultaneously at $\overline{x}$. Both of them can be verified by contradiction as follows.

Suppose the former case is satisfied at $\overline{x}$, then it has $\ell_i = 0$ and $t_e = 0$, leading to a trivial solution $\overline{x} = \mathbf{0}$ that violates the active condition on (5b). Next, if the later case is satisfied, i.e., there exists a constraint pair (e.g., $i \in \mathcal{S}$) being inactive simultaneously at $\overline{x}$, we have to select $|\mathcal{S}|$ constraints to be active from the rest $2(|\mathcal{S}|-1)$ constraints in (5c). In this case, another constraint pair (say, $j \in \mathcal{S}$ with $j \neq i$) needs to be active concurrently for composing the active constraint set, which eventually returns back to the former case. Therefore, one and only one of the constraints between $\ell_i \le t_e r_i (1+d)^{1-|\mathcal{S}|}$ and $\ell_i \ge 0$ of each $i \in \mathcal{S}$ can be active at $\overline{x}$. This completes the proof.

### B. Proof of Proposition 1

To prove this, we need the following lemma which can be easily proved.

*Lemma 3:* Consider the fractions $x_1/y_1$ and $x_2/y_2$, with $x_i, y_i > 0, i = 1, 2$. Then,

$$\min\left\{\frac{x_1}{y_1}, \frac{x_2}{y_2}\right\} \le \frac{x_1+x_2}{y_1+y_2} \le \max\left\{\frac{x_1}{y_1}, \frac{x_2}{y_2}\right\}.$$

We first prove the proposition from sufficiency. If the given $\mathcal{S}$ satisfies condition (15), using Lemma 3, the following inequality holds for all $i \in \mathcal{S}$:

$$\begin{aligned}
&\frac{\omega_i}{(a_i + b_i\gamma_i)} \\
&\quad = \frac{\omega_i r_i}{(a_i + b_i\gamma_i) r_i} \\
&\quad \overset{(a)}{\geq} \frac{\sum_{j\in\mathcal{S}\backslash\{i\}} \omega_j r_j + \omega_i r_i}{(1+d)^{|\mathcal{S}|-1} + \sum_{j\in\mathcal{S}\backslash\{i\}} (a_j + b_j\gamma_j) r_j + (a_i + b_i\gamma_i) r_i} \\
&\quad \overset{(b)}{\geq} \frac{\sum_{j\in\mathcal{S}\backslash\{i\}} \omega_j r_j}{(1+d)^{|\mathcal{S}|-1} + \sum_{j\in\mathcal{S}\backslash\{i\}} (a_j + b_j\gamma_j) r_j}, \qquad (42)
\end{aligned}$$

where the right hand side of the first inequality is identical to $R$. The term in the second inequality is the sum offloading rate achieved by setting $\widetilde{\mathcal{S}} = \mathcal{S}\backslash\{i\}$ in (13). (a) holds for all $i \in \mathcal{S}$ since condition (15) is met. (b) is deduced by Lemma 3. The relation (b) holding for all $i \in \mathcal{S}$ indicates that $\widetilde{\mathcal{S}} = \mathcal{S}$ has a larger sum offloading rate than any neighbors $\widetilde{\mathcal{S}} = \mathcal{S}\backslash\{i\}$, $\forall i \in \mathcal{S}$. Thus, $\widetilde{\mathcal{S}} = \mathcal{S}$ is the local optimum of Problem (13). According to the results in [36], any point of local optimum of Problem (13) is also point of global optimum. Therefore, $\widetilde{\mathcal{S}} = \mathcal{S}$ is the optimal solution of Problem (13).

Next, from necessity, since $\widetilde{\mathcal{S}} = \mathcal{S}$ is the optimal solution of Problem (13), the local optimum condition is also met. Thus, we have the relation (b) in (42) satisfying for all $i \in \mathcal{S}$. Using Lemma 3, (a) is deduced for any $i \in \mathcal{S}$, i.e., condition (15) holds the given $\mathcal{S}$. This completes the proof.

### *C. Proof of Proposition 2*

Let $\mathcal{S}^*$ and $R^*$ denote the optimal solution and the optimal objective value of the Problem (18) that has relaxed constraint (15), respectively. For ease of expression, we sort the entities $\left\{\frac{\omega_i}{a_i+b_i\gamma_i}\right\}$ in $\mathcal{S}^*$ in the descending order $\frac{\omega_1}{a_1+b_1\gamma_1} > \cdots > \frac{\omega_m}{a_m+b_m\gamma_m}$, with $m = |\mathcal{S}^*|$.

The proposition can be proved by contradiction. Suppose that the optimal solution $\mathcal{S}^*$ violates constraint (15), we have $\frac{\omega_1}{a_1+b_1\gamma_1} > \cdots > \frac{\omega_{j-1}}{a_{j-1}+b_{j-1}\gamma_{j-1}} > R^* > \frac{\omega_j}{a_j+b_j\gamma_j} > \cdots > \frac{\omega_m}{a_m+b_m\gamma_m}$. Denote $\mathcal{S}' = \mathcal{S}^*\backslash\{j\}$. Since $R^* > \frac{\omega_j}{a_j+b_j\gamma_j}$, using Lemma 3, we have the following inequality

$$\begin{aligned}
R^* &= \frac{\sum_{i\in\mathcal{S}^*} \omega_i r_i}{(1+d)^{|\mathcal{S}^*|-1} + \sum_{i\in\mathcal{S}^*} (a_i + b_i\gamma_i) r_i} \\
&\overset{(a)}{<} \frac{\sum_{i\in\mathcal{S}^*\backslash\{j\}} \omega_i r_i}{(1+d)^{|\mathcal{S}^*|-1} + \sum_{i\in\mathcal{S}^*\backslash\{j\}} (a_i + b_i\gamma_i) r_i} \\
&\overset{(b)}{<} \frac{\sum_{i\in\mathcal{S}'} \omega_i r_i}{(1+d)^{|\mathcal{S}'|-1} + \sum_{i\in\mathcal{S}'} (a_i + b_i\gamma_i) r_i}, \qquad (43)
\end{aligned}$$

where (a) is deduced from Lemma 3 and (b) is due to $|\mathcal{S}^*| > |\mathcal{S}'|$. (43) shows that sum offloading rate of $\mathcal{S}'$ is larger than that of $\mathcal{S}^*$, which contradicts to the assumption that $\mathcal{S}^*$ is optimal. Notice that $\mathcal{S}'$ may not satisfy (15) at present. However, as long as $\mathcal{S}'$ does not meet condition (15), we can treat the current $\mathcal{S}'$ as another $\mathcal{S}^*$ and use the same manner in (43) to construct a new $\mathcal{S}'$. This guarantees to find $\mathcal{S}'$ that meets (15), since the extreme case is $\mathcal{S}'$ with $|\mathcal{S}'| = 1$ (i.e., single user) that always satisfies (15).

If $\mathcal{S}$ does not meet (15), we can always find an $\mathcal{S}'$ satisfying (15) and with larger objective value than $\mathcal{S}$. Thus, it can be concluded that $\mathcal{S}$ violating (15) cannot be the optimal solution of the relaxed problem. This completes the proof.

### *D. Proof of Lemma 2*

We provide the sufficiency proof of Lemma 2 since the necessity proof is just reversed. Let $R'_m$ denote the root of $g(R_m) = 0$ (note that the existence and uniqueness of the root of $g(R_m) = 0$ are proved in [32]) and $\mathbf{x}' \in \mathcal{F}_m$ an optimal solution of $g(R'_m)$.

Since $g(R'_m) = \max_{x\in\mathcal{F}_m} \{N(\mathbf{x}) - D(\mathbf{x})R'_m\} = 0$, we have

$$N(\mathbf{x}) - D(\mathbf{x})R'_m \leq N(\mathbf{x}') - D(\mathbf{x}')R'_m = 0, \quad \forall x \in \mathcal{F}_m. \qquad (44)$$

As $D(\mathbf{x}), D(\mathbf{x}') > 0$, $\forall \mathbf{x} \in \mathcal{F}_m$, (44) can be re-written as $R'_m = \frac{N(\mathbf{x}')}{D(\mathbf{x}')} \geq \frac{N(\mathbf{x})}{D(\mathbf{x})}, \forall \mathbf{x} \in \mathcal{F}_m$, i.e., $R'_m = R^*_m$ and $\mathbf{x}' = \mathbf{x}^*$ are the optimal objective value and an optimal solution of Problem (20), respectively. This completes the proof.

### *E. Proof of Proposition 3*

With $\frac{1}{a_1+b_1\gamma_1} = \ldots = \frac{1}{a_K+b_K\gamma_K} = \frac{1}{a+b\gamma}$, th sum offloading rate $R$ can be simplified as

$$\begin{aligned}
R &= \frac{\sum_{i\in\mathcal{S}} r_i}{(1+d)^{|\mathcal{S}|-1} + (a+b\gamma)\sum_{i\in\mathcal{S}} r_i} \\
&= \left(\frac{(1+d)^{|\mathcal{S}|-1}}{\sum_{i\in\mathcal{S}} r_i} + (a+b\gamma)\right)^{-1}.
\end{aligned}$$

Therefore, for maximizing $R$, it is sufficient to minimize $\frac{(1+d)^{|\mathcal{S}|-1}}{\sum_{i\in\mathcal{S}} r_i}$ instead. Observe that for a given $|\mathcal{S}|$, the minimum $\frac{(1+d)^{|\mathcal{S}|-1}}{\sum_{i\in\mathcal{S}} r_i}$ is achieved by selecting $|\mathcal{S}|$ largest $r_i$'s. For simplicity, we notate $|\mathcal{S}| = n$ and sort $r_i$'s in the descending order $r_1 \geq \cdots \geq r_K$. Then the problem becomes finding the minimum point in the sequence $\{f_n \triangleq \frac{(1+d)^{n-1}}{\sum_{i=i}^{n} r_i}\}$, with $n = 1, \cdots, K$. It can be checked that the sequence $\{f_n\}$ has a monotone property with $n$. Specifically, there exists an index $n_0$, in which $\{f_n\}$ is monotonically decreasing when $1 \leq n \leq n_0$ and monotonically increasing when $n_0 \leq n \leq K$. Therefore, $f_{n_0}$ is the minimum point of sequence $\{f_n\}$. By defining $r_0 = 0$ and letting $f_n \leq f_{n-1}$, we derive condition $r_n \geq d\sum_{i=0}^{n-1} r_i$ and $n_0$ is the largest index satisfying this condition. Thus, $\mathcal{S} = \{i \,|1 \leq i \leq n_0\}$ is the optimal offloading-user set. This completes the proof.

### *F. Proof of Proposition 4*

For solving the minimum $T$ of Problem (P3), it can be observed from (10b) that $t_e$ should be minimized. By (10c), the minimum $t_e$ can be expressed as $\max_{i\in\mathcal{S}} \left\{\frac{\ell_i}{r_i(1+d)^{1-|\mathcal{S}|}}\right\}$. Meanwhile, since $\mathcal{S} = \{i|\ell_i > 0, i \in \mathcal{K}\}$, Problem (P3) can be refined as

$$\min_{\{\ell_i\},\ T} \quad T \tag{45}$$

$$\text{s.t.} \quad \frac{c_i(L_i-\ell_i)}{f_i} \le T, \quad 0 \le \ell_i \le L_i, \quad \forall i \in \mathcal{K}, \tag{46}$$

$$\sum_{i\in\mathcal{K}} (a_i + b_i\gamma_i)\,\ell_i + \max_{i\in\mathcal{K}} \left\{ \frac{\ell_i}{r_i(1+d)^{1-\sum_{i\in\mathcal{K}}\chi\{\ell_i>0\}}} \right\} \le T, \tag{47}$$

where $|\mathcal{S}| \triangleq \sum_{i\in\mathcal{K}} \chi\{\ell_i > 0\}$ and $\chi\{\cdot\}$ is a binary indicator function.

By (46), the minimum offloaded bits is obtained as $L_i^{\min} = \left[L_i - \frac{Tf_i}{c_i}\right]^+$, $\forall i$. Notice that here $L_i^{\min}$ is a function of $T$. For notation simplicity, we notate $\sum_{i\in\mathcal{K}} \chi\{\ell_i > 0\} = \sum_{i\in\mathcal{K}} \chi\{L_i^{\min} > 0\} \triangleq N(T)$. Then Problem (45) can be simplified as

$$\min_{\{\ell_i\},\ T} \quad T \tag{48}$$

$$\text{s.t.} \quad \sum_{i\in\mathcal{K}} (a_i + b_i\gamma_i)\, L_i^{\min} + \max_{i\in\mathcal{K}} \left\{ \frac{L_i^{\min}}{r_i(1+d)^{1-N(T)}} \right\} \le T, \tag{49}$$

where the left hand side of (49) is the minimum time required for completing the minimum offloaded tasks for a given $T$. Since it is monotonically decreasing with $T$ in the interval of $T \in [0, \max_{i\in\mathcal{K}}\{c_iL_i/f_i\}]$, it is easily observed that the minimum of Problem (48) is achieved only when (49) holds with equality. This completes the proof.

## References

[1] A. U. R. Khan, M. Othman, S. A. Madani, and S. U. Khan, "A survey of mobile cloud computing application models," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 1, pp. 393–413, 1st Quart., 2013.

[2] *Mobile-Edge Computing-Introductory Technical White Paper*, ETSI, Sophia Antipolis, France, Sep. 2014.

[3] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2322–2358, 4th Quart., 2017.

[4] X. Pu, L. Liu, Y. Mei, S. Sivathanu, Y. Koh, and C. Pu, "Understanding performance interference of I/O workload in virtualized cloud environments," in *Proc. IEEE 3rd Int. Conf. Cloud Comput.*, Jul. 2010, pp. 51–58.

[5] S. Ibrahim, B. He, and H. Jin, "Towards pay-as-you-consume cloud computing," in *Proc. IEEE SCC*, Jul. 2011, pp. 370–377.

[6] S.-G. Kim, H. Eom, and H. Y. Yeom, "Virtual machine consolidation based on interference modeling," *J. Supercomput.*, vol. 66, no. 3, pp. 1489–1506, Dec. 2013. doi: 10.1007/s11227-013-0939-2.

[7] D. Bruneo, "A stochastic model to investigate data center performance and QoS in IaaS cloud computing systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 3, pp. 560–569, Mar. 2014.

[8] W. B. Slama, Z. Brahmi, and M. M. Gammoudi, "Interference-aware virtual machine placement in cloud computing system approach based on fuzzy formal concepts analysis," in *Proc. IEEE WETICE*, Jun. 2018, pp. 48–53.

[9] W. Zhang, Y. Wen, K. Guan, D. Kilper, H. Luo, and D. O. Wu, "Energy-optimal mobile cloud computing under stochastic wireless channel," *IEEE Trans. Wireless Commun.*, vol. 12, no. 9, pp. 4569–4581, Sep. 2013.

[10] Y. Mao, J. Zhang, Z. Chen, and K. B. Letaief, "Dynamic computation offloading for mobile-edge computing with energy harvesting devices," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 12, pp. 3590–3605, Dec. 2016.

[11] C. You, K. Huang, and H. Chae, "Energy efficient mobile cloud computing powered by wireless energy transfer," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 5, pp. 1757–1771, May 2016.

[12] Y. Wang, M. Sheng, X. Wang, L. Wang, and J. Li, "Mobile-edge computing: Partial computation offloading using dynamic voltage scaling," *IEEE Trans. Commun.*, vol. 64, no. 10, pp. 4268–4282, Oct. 2016.

[13] O. Munoz, A. Pascual-Iserte, and J. Vidal, "Optimization of radio and computational resources for energy efficiency in latency-constrained application offloading," *IEEE Trans. Veh. Technol.*, vol. 64, no. 10, pp. 4738–4755, Oct. 2015.

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

[15] M. Liu and Y. Liu, "Price-based distributed offloading for mobile-edge computing with computation capacity constraints," *IEEE Wireless Commun. Lett.*, vol. 7, no. 3, pp. 420–423, Jun. 2018.

[16] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Trans. Netw.*, vol. 24, no. 5, pp. 2795–2808, Oct. 2016.

[17] S. Sardellitti, G. Scutari, and S. Barbarossa, "Joint optimization of radio and computational resources for multicell mobile-edge computing," *IEEE Trans. Signal Inf. Process. Over Netw.*, vol. 1, no. 2, pp. 89–103, Jun. 2015.

[18] C. Wang, F. R. Yu, C. Liang, Q. Chen, and L. Tang, "Joint computation offloading and interference management in wireless cellular networks with mobile edge computing," *IEEE Trans. Veh. Technol.*, vol. 66, no. 8, pp. 7432–7445, Aug. 2017.

[19] Y. Mao, J. Zhang, S. H. Song, and K. B. Letaief, "Stochastic joint radio and computational resource management for multi-user mobile-edge computing systems," *IEEE Trans. Wireless Commun.*, vol. 16, no. 9, pp. 5994–6009, Sep. 2017.

[20] J. Kwak, Y. Kim, J. Lee, and S. Chong, "DREAM: Dynamic resource and task allocation for energy minimization in mobile cloud systems," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 12, pp. 2510–2523, Dec. 2015.

[21] S. Bi and Y. Zhang, "Computation rate maximization for wireless powered mobile-edge computing with binary computation offloading," *IEEE Trans. Wireless Commun.*, vol. 17, no. 6, pp. 4177–4190, Jun. 2018.

[22] F. Wang, J. Xu, X. Wang, and S. Cui, "Joint offloading and computing optimization in wireless powered mobile-edge computing systems," *IEEE Trans. Wireless Commun.*, vol. 17, no. 3, pp. 1784–1797, Mar. 2018.

[23] Y. Huang and Y. Liu, "User cooperation for NOMA-based mobile edge computing," in *Proc. IEEE ICCS*, Dec. 2018, pp. 395–400.

[24] K. Guo, M. Sheng, J. Tang, T. Q. S. Quek, and Z. Qiu, "Exploiting hybrid clustering and computation provisioning for green C-RAN," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 52, pp. 4063–4076, Dec. 2016.

[25] Y. Liu, "Exploiting NOMA for cooperative edge computing," *IEEE Wireless Commun.*, to be published.

[26] X. Lyu, H. Tian, C. Sengul, and P. Zhang, "Multiuser joint task offloading and resource optimization in proximate clouds," *IEEE Trans. Veh. Technol.*, vol. 66, no. 4, pp. 3435–3447, Apr. 2017.

[27] C. You, Y. Zeng, R. Zhang, and K. Huang, "Asynchronous mobile-edge computation offloading: Energy-efficient resource management," *IEEE Trans. Wireless Commun.*, vol. 17, no. 11, pp. 7590–7605, Nov. 2018.

[28] M. Xiao, N. B. Shroff, and E. K. P. Chong, "A utility-based power-control scheme in wireless cellular systems," *IEEE/ACM Trans. Netw.*, vol. 11, no. 2, pp. 210–221, Apr. 2003.

[29] X. Pu *et al.*, "Who is your neighbor: Net I/O performance interference in virtualized clouds," *IEEE Trans. Services Comput.*, vol. 6, no. 3, pp. 314–329, Sep. 2013.

[30] J. N. Matthews *et al.*, "Quantifying the performance isolation properties of virtualization systems," in *Proc. ExpCS*, 2007, Art. no. 6.

[31] M. Mishra and A. Sahoo, "On theory of VM placement: Anomalies in existing methodologies and their mitigation using a novel vector based approach," in *Proc. IEEE 4th Int. Conf. Cloud Comput.*, Jul. 2011, pp. 275–282.

[32] W. Dinkelbach, "On nonlinear fractional programming," *Manage. Sci.*, vol. 13, no. 7, pp. 492–498, 1967. doi: 10.1287/mnsc.13.7.492.

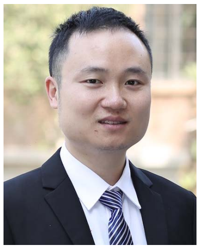
[33] T. Matsui, Y. Saruwatari, and M. Shigeno, "An analysis of Dinkelbach's algorithm for 0–1 fractional programming problems," Tech. Rep., 1992.

[34] T. Q. Dinh, J. Tang, Q. D. La, and T. Q. S. Quek, "Offloading in mobile edge computing: Task allocation and computational frequency scaling," *IEEE Trans. Commun.*, vol. 65, no. 8, pp. 3571–3584, Aug. 2017.

[35] D. Bertsimas and J. Tsitsiklis, *Introduction to Linear Optimization*. Belmont, MA, USA: Athena Scientific, 1997.

[36] P. L. Hammer and S. Rudeanu, *Boolean Methods in Operations Research and Related Areas*, vol. 7. Berlin, Germany: Springer-Verlag, 2012.

**Zezu Liang** (S'19) received the B.Eng. degree from the School of Electronic and Information Engineering, South China University of Technology, in 2017. He is currently pursuing the Ph.D. degree with the Department of Information Engineering, The Chinese University of Hong Kong (CUHK). His research interests include mobile edge computing and resource management.



**Yuan Liu** (S'11–M'13–SM'18) received the B.S. degree from the Hunan University of Science and Technology, Xiangtan, China, in 2006, the M.S. degree from the Guangdong University of Technology, Guangzhou, China, in 2009, and the Ph.D. degree from Shanghai Jiao Tong University, China, in 2013, all in electronic engineering.

Since 2013, he has been with the School of Electronic and Information Engineering, South China University of Technology, Guangzhou, where he is currently an Associate Professor. His research interests include 5G communications and beyond, mobile edge computation offloading, and machine learning in wireless networks. He serves as an Editor for the IEEE COMMUNICATIONS LETTERS and IEEE ACCESS.

**Tat-Ming Lok** (SM'03) received the B.Sc. degree in electronic engineering from The Chinese University of Hong Kong, Hong Kong, and the M.S.E.E. and Ph.D. degrees in electrical engineering from Purdue University, West Lafayette, IN, USA.

He is currently an Associate Professor with the Department of Information Engineering, The Chinese University of Hong Kong. His research interests include communication theory, communication networks, signal processing for communications, and wireless systems. He served on the technical program committees for many international conferences, including the IEEE International Conference on Communications, the IEEE Vehicular Technology Conference, the IEEE GLOBECOM, the IEEE Wireless Communications and Networking Conference, and the IEEE International Symposium on Information Theory. He also served as an Associate Editor for the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY from 2002 to 2008 and an Editor for the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS from 2015 to 2018.

**Kaibin Huang** received the B.Eng. (Hons.) and M.Eng. degrees in electrical engineering from the National University of Singapore and the Ph.D. degree from The University of Texas at Austin (UT Austin).

Since 2014, he has been an Assistant Professor with the Department of Electrical and Electronic Engineering (EEE), The University of Hong Kong. He was a Faculty Member with the Department of Applied Mathematics (AMA), The Hong Kong Polytechnic University (PolyU), and the Department of Electrical and Electronics Engineering, Yonsei University, South Korea. He has received a University Visiting Scholarship with Kansai University, Japan, in 2017. His research interests include edge machine learning, edge computing, and 5G-and-beyond communications. He was an Elected Member of the SPCOM Technical Committee of the IEEE Signal Processing Society from 2012 to 2015. He frequently serves on the technical program committees of major IEEE conferences in wireless communications. Most recently, he has served as the Lead Chair for the Communication Theory Symposium of the IEEE GLOBECOM 2014 and the Wireless Communication Symposium of the IEEE GLOBECOM 2017, and the TPC Co-Chair for the IEEE CTW 2013 and the IEEE PIMRC 2017. He has edited the JSAC 2015 special issue on communications powered by energy harvesting. He has received the IEEE Communication Society's 2019 Best Tutorial Paper Award, the 2015 Asia–Pacific Outstanding Paper Award, the Outstanding Teaching Award from Yonsei, Motorola Partnerships in Research Grant, the University Continuing Fellowship from UT Austin, and the Best Paper Award from the IEEE GLOBECOM 2006 and the IEEE/CIC ICCC in 2018. He is currently an Associate Editor for the IEEE TRANSACTIONS ON GREEN COMMUNICATIONS AND NETWORKING. He has also served as an Associate Editor for the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS (JSAC) series on green communications and networking, and the IEEE WIRELESS COMMUNICATIONS LETTERS.