IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received September 28, 2019, accepted October 7, 2019, date of publication October 10, 2019, date of current version October 24, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2946843

Delay-Driven Computation Task Scheduling in Multi-Cell Cellular Edge Computing Systems

YUAN ZHANG[1], (Member, IEEE), AND PENG DU[2]
[1]National Mobile Communications Research Laboratory, Southeast University, Nanjing 210096, China
[2]College of Automation, College of Artificial Intelligence, Nanjing University of Posts and Telecommunications, Nanjing 210023, China

Corresponding author: Yuan Zhang (y.zhang@seu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61571111 and in part by the National Key Research and Development Program of China under Grant 2018YFB1800800. The work of P. Du was supported by the Incubation Project of the National Natural Science Foundation of China at Nanjing University of Posts and Telecommunications under Grant NY219106.

ABSTRACT This paper studies the joint scheduling of subcarrier, base station, transmit power, and virtual machine in multi-cell cellular edge computing systems to minimize the total delay experienced by tasks of users. Traditional work considers the queue length based Lyapunov function and designs the corresponding scheduling algorithms. This work considers the delay based Lyapunov function. Firstly, the delay of the communication and computing queues in multi-cell cellular edge computing systems is modeled as the virtual delay queue based on which a delay based Lyapunov function is defined. Then, the joint subcarrier allocation, base station selection, power control, and virtual machine scheduling algorithms are proposed to minimize the conditional drift of the delay based Lyapunov function. Simulation results show the proposed scheduling algorithm performs better than the traditional queue length based one in the performance of total delay.

INDEX TERMS Cellular edge computing, delay, scheduling.

I. INTRODUCTION

In cellular edge computing systems [1]–[4], there are two types of resources: the first is the communication resource, the second is the computing resource in the base stations (BSs). This paper studies the resource scheduling algorithm. In the literature, the goals of resource scheduling algorithms are diverse. For example, in [5], the goal of resource scheduling is to maximize the profit of service providers; in [6], the goal of resource scheduling is to maximize the utility function of the user's long-term average transmission rate, etc. In this work, since low latency has been identified as one of the most important performance indicators for next-generation cellular systems [7], the scheduling algorithm will aim to minimize delay. In cellular edge computing systems, since there are two types of resources, there are correspondingly two types of delays: the first is the communication delay including the transmission delay and communication queue waiting delay, the second is the computing delay including the execution delay and computing queue waiting delay. This paper studies the resource scheduling algorithm which minimizes the total delay in cellular edge computing systems.

Scheduling in cellular edge computing systems has been studied recently. According to the assumption of the delay in cellular edge computing systems, existing studies can be classified into the following three categories. For the first category, only the transmission delay and execution delay are considered. But the queue waiting delay is not considered. For example, in [8]–[17], only transmission delay and execution delay are considered. The second category is queue model based. For this category, in addition to the transmission delay and execution delay, the queue waiting delay is considered, too. However, this category assumes the queue can be modelled as the traditional queue (e.g., the M/M/1 queue) so that the delay formulas of the queueing theory can be readily applied. For example, in [18]–[25], the communication and computing queues are assumed to be the M/M/1 or M/G/1 queues. The third category is Lyapunov optimization based. For this category, the queue waiting delay is also considered. Further, this category does not make any assumption of the model of the queues, but uses the Lyapunov optimization method to design scheduling algorithms. For example, the scheduling algorithms in [26]–[33]

The associate editor coordinating the review of this manuscript and approving it for publication was Eyuphan Bulut.

were designed according to the following procedure: firstly, consider the average delay and queue length are equivalent by Little's Law; secondly, define the queue length based Lyapunov function; thirdly, design the scheduling algorithm which minimizes the conditional drift of the Lyapunov function according to the framework established in [34]–[35]. In this work, we focus on the third category of scheduling (i.e., the Lyapunov optimization based scheduling).

Most existing Lyapunov optimization based scheduling algorithms were queue length based. These scheduling algorithms did not need any assumption of the model of the queues. This is the pros of the third category of scheduling. However, this also means the delay formulas of the queueing theory cannot be used. Then, due to the lack of delay formulas, the third category of scheduling cannot solve the delay control problem directly, but has to solve the queue length control problem instead. Additionally, most existing Lyapunov optimization based scheduling algorithms only considered the single-cell scenario. Different from those studies, this work will focus on the multi-cell scenario in which in addition to deciding how to allocate communication resource between users, the scheduler still needs to decide how to assign BSs to users.

Hence, this work will extend the traditional Lyapunov optimization based scheduling algorithms by studying the scheduling algorithm which solves the delay control problem directly and does not need any assumption of the model of the queues for multi-cell cellular edge computing systems. The work in [36] is our first step toward this direction which focused on the single-cell scenario. Compared with our previous work in [36], this work focuses on the multi-cell scenario and makes the following extensions: for the communication subsystem, instead of the linear communication model used in [36], this work uses the Shannon capacity formula to model the transmission of data bits over the air and studies the subcarrier allocation, BS selection, and power control subproblems; for the computing subsystem, instead of the linear computing model used in [36], this work uses the utility function of the number of virtual machine (VM) to model the execution time of tasks and studies the VM scheduling subproblem. Therefore, this work extends our previous work in [36] by using more practical system models and studying the scheduling of four types of resources (i.e., subcarriers, BSs, power, and VMs) in multi-cell cellular edge computing systems.

The contributions of this work are summarized as follows. Firstly, the model of the delay experienced by tasks in the communication and computing queues in multi-cell cellular edge computing systems is established and expressed as the virtual delay queues. Secondly, based on the virtual delay queue model, a delay based Lyapunov function is defined, then a novel Lyapunov optimization based joint subcarrier allocation, BS selection, power control, and VM scheduling is proposed to stabilize the virtual delay queues. Thirdly, simulation experiments are carried out to verify that the delay performance of the proposed delay-based algorithm is better than the traditional queue length-based algorithm. The main notations used in this paper are summarized in Table 1.

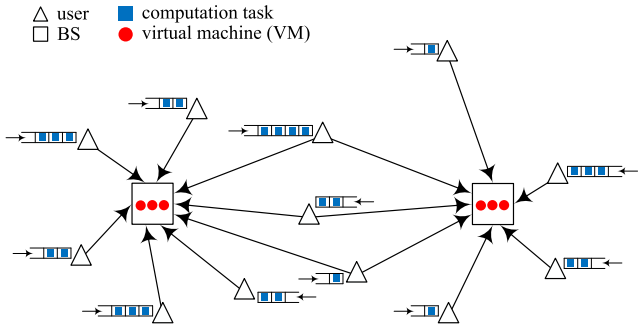


FIGURE 1. The multi-cell cellular edge computing system.

II. SYSTEM MODEL

Consider a multi-cell cellular edge computing system, as shown in Fig. 1, in which there are I users and J BSs. For each user i, he will generate the need to execute a computation task at random instants. Each time there is a computation task requirement, user i will immediately transfer the request for the computation task to the BS via uplink communications. There are a total of K subcarriers in the uplink. The execution of a computation task is considered as a whole and let D_i denote the number of communication bits required by user i to transfer the request of one computation task to a BS. So there is a need for a mechanism for efficient scheduling of all of these communication process related resources over the air interface, including subcarrier allocation, BS selection, and power control. After these computation task requests arrive at the BS, the VM computing resources of the BS will be used to execute these computation tasks. For each BS j, there are a total of F_j VMs. So there is a need for a mechanism for efficient scheduling of all of these computing resources at each BS, that is, VM scheduling. Therefore, this paper studies the task scheduling algorithm, including: (i) subcarrier allocation, (ii) BS selection, (iii) power control, and (iv) VM scheduling to minimize the delay in the multi-cell cellular edge computing system.

The overall queue model of the system is shown in Fig. 2. Next, the queue model of the communication subsystem and the computing subsystem will be presented separately.

A. COMMUNICATION SUBSYSTEM

Let $h_{i,j,k}[n]$ denote the channel gain of the link from user i to BS j on subcarrier k in slot n, $p_{i,j,k}[n]$ denote the power allocated to the link from user i to BS j on subcarrier k in slot n, and $R_{i,j,k}[n]$ denote the number of bits transmitted from user i to BS j on subcarrier k in slot n. We can use the following Shannon capacity formula to calculate the value of $R_{i,j,k}[n]$, that is:

$$R_{i,j,k}[n] = TB \log_2 \left(1 + \frac{h_{i,j,k}[n] p_{i,j,k}[n]}{P_I + P_{in}}\right), \qquad (1)$$

TABLE 1. Summary of notations.

notation	description
I	number of users
J	number of BSs
K	number of subcarriers
D_i	number of communication bits required by user i to transfer the request of one computation task to a BS
$h_{i,j,k}[n]$	channel gain of the link from user i to BS j on subcarrier k in slot n
$p_{i,j,k}[n]$	power allocated to the link from user i to BS j on subcarrier k in slot n
$R_{i,j,k}[n]$	number of bits transmitted from user i to BS j on subcarrier k in slot n
T	duration of each time slot
B	bandwidth of each subcarrier
P_{in}	power of background interference and noise
$d_{i,e}$	index of BS to which the eth task of user i is offloaded
$n_{i,e}$	index of slot in which the value of $d_{i,e}$ is determined
$x_{i,j,k}[n]$	index of task of user i to which the subcarrier k from user i to BS j in slot n is allocated
Ψ_i	the set of BSs which are accessible by user i
P_0	transmit power used in the uniform power control method
R_0	number of transmitted bits used in the reverse channel power control method
$P_{i,\max}$	maximum transmit power of user i
$U_i[n-1]$	number of tasks in the ith queue at the beginning of slot n
$X_{i,j}[n]$	number of tasks offloaded from the ith queue to BS j in slot n
$A_i[n]$	number of newly arriving tasks into the ith queue at the end of slot n
F_j	number of VMs in BS j
$u_i(g)$	number of slots required to execute a task of user i given g VMs are allocated to this task
$g_{\max}$	maximum number of VMs which can be allocated to a task
$g_{j,i,e}$	number of VMs allocated to the task e of user i in BS j
$v_{j,i,e}$	index of slot at the beginning of which the value of $g_{j,i,e}$ is decided
$f_{j,i,e}[n]$	number of VMs allocated to the task e of user i in BS j in slot n
$F_{j,i}[n]$	total number of VMs allocated to user i in BS j in slot n
$Q_{j,i}[n]$	number of tasks in the computing queue of user i in BS j at the end of slot $n+1$
$Y_{j,i}[n]$	number of tasks leaving the computing queue of user i in BS j during slot $n+1$
$W_i[n]$	normalized version of $\Gamma_i[n]$
$Z_{j,i}[n]$	normalized version of $\Xi_{j,i}[n]$
$L[n]$	Lyapunov function
$\Delta[n]$	conditional Lyapunov drift
$C[n]$	feasible user set used in communication resource scheduling
$H_j[n]$	feasible user set of BS j used in computing resource scheduling

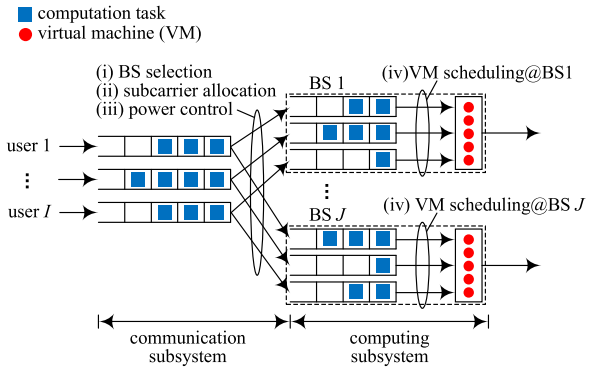


FIGURE 2. The queue model of the system.

where T is the slot length in second, B is the bandwidth of subcarrier in Hz, $P_I = \sum_{i' \neq i}(h_{i',j,k}[n] \cdot \sum_{j'} p_{i',j',k}[n])$ is the interference power, and P_{in} is the power of some background interference and noise. For the eth task of user i, let $d_{i,e}$ denote the index of BS to which this task is offloaded, $n_{i,e}$ denote the index of slot in which the value of $d_{i,e}$ is determined, and $x_{i,j,k}[n]$ denote the index of task of user i to which the subcarrier k from user i to BS j in slot n is allocated. If the following condition holds for the eth task of user i:

$$\sum_{t=n_{i,e}}^{n} \sum_{k:x_{i,d_{i,e},k}[t]=e} R_{i,d_{i,e},k}[t] \geq D_i, \tag{2}$$

this task will be offloaded from user i to the BS $d_{i,e}$ at the end of slot n. Subcarriers will be allocated based on the principle of interference avoidance [37]–[43]. That is, for any two user-BS links that may interfere with each other, they will not be assigned the same subcarrier. Specifically, let Ψ_i denote the set of BSs which are accessible by user i (i.e., can receive the signal from user i with a signal-to-interference-plus-noise-ratio exceeding a threshold). Then, interference avoidance means that for each BS j on each subcarrier k in each slot n, we have the following constraint:

$$\sum_{i:j \in \Psi_i} \mathbf{1}\left\{x_{i,j,k}[n] > 0\right\} \leq 1, \tag{3}$$

where $\mathbf{1}\{\cdot\}$ equals one if the condition is true and zero otherwise.

There are many different power control methods for multi-subcarrier communication systems (e.g., the water-filling power control). To reduce the complexity, this work consider the following two types of power control methods. The first type is the uniform power control method in which the transmit power of each subcarrier is a fixed constant. Let P_0 denote the specified transmit power of each subcarrier in each slot. Then we have:

$$p_{i,j,k}[n] = \begin{cases} 0, & x_{i,j,k}[n] = 0 \\ P_0, & x_{i,j,k}[n] > 0. \end{cases} \tag{4}$$

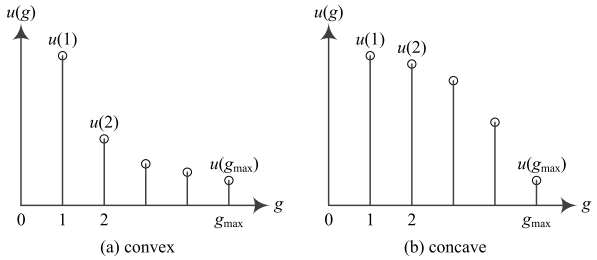


FIGURE 3. The utility function.

The second type is the reverse channel power control method in which the transmit bit of each subcarrier is a fixed constant. Let R_0 denote the specified number of bits transmitted on each subcarrier in each slot. Substituting into (1), we have:

$$p_{i,j,k}[n] = \begin{cases} 0, & x_{i,j,k}[n] = 0 \\ \dfrac{\sigma^2}{h_{i,j,k}[n]}(2^{\frac{R_0}{TB}} - 1), & x_{i,j,k}[n] > 0, \end{cases} \tag{5}$$

Additionally, for each user i in slot n, we have the constraint:

$$\sum_{j=1}^{J}\sum_{k=1}^{K} p_{i,j,k}[n] \le P_{\max}, \tag{6}$$

where $P_{\max}$ is the maximum transmit power of user.

Now we describe the evolution of communication queues. For the communication queue of user i, let $U_i[n-1]$ denote the number of tasks in the queue at the beginning of slot n, $X_{i,j}[n]$ denote the number of tasks offloaded to BS j in slot n, and $A_i[n]$ denote the number of newly arriving tasks at the end of slot n. Using the variables defined previously, the value of $X_{i,j}[n]$ can be calculated as:

$$X_{i,j}[n] = \sum_{e:d_{i,e}=j} \mathbf{1}\{\sum_{t=n_{i,e}}^{n} \sum_{k:x_{i,j,k}[t]=e} R_{i,j,k}[t] \ge D_i\}. \tag{7}$$

Hence, we have:

$$U_i[n] = \left(U_i[n-1] - \sum_{j\in\Psi_i} X_{i,j}[n]\right)^+ + A_i[n], \tag{8}$$

where $(\cdot)^+ = \max(\cdot, 0)$.

Therefore, the communication scheduler must determine the values of the following variables: (i) $d_{i,e}$ and $n_{i,e}$ (i.e., BS selection), (ii) $x_{i,j,k}[n]$ (i.e., subcarrier allocation), and (iii) $p_{i,j,k}[n]$ (i.e., power control).

B. COMPUTING SUBSYSTEM

At the beginning of each slot, the computing scheduler of each BS determines how to allocate VMs to the tasks which are offloaded to this BS. We use a utility function $u(g)$ to represent the number of slots required to execute a task with g VMs, which is a decreasing function with respect to the VM number g. Generally, the more the VM is allocated, the less the execution time of the task. The exact expression of a utility function may depend on the type of task. Fig. 3 plots the utility functions for the two types of tasks, in which we assume at most $g_{\max}$ VMs can be allocated to a task. We leave the work of finding utility functions to computer scientists, and focus on resource scheduling algorithm for a given set of utility functions.

Let $u_i(g)$ denote the utility function of tasks of user i, that is, $u_i(g)$ represents the number of slots needed to execute a task of user i given g VMs are allocated to this task. Then, for the task e of user i, if the scheduler of BS j decides to allocate $g_{j,i,e}$ VMs to this task at the beginning of slot $v_{j,i,e}$, these VMs will be used by this task for consecutive $u_i(g_{j,i,e})$ slots since slot $v_{j,i,e}$ in BS j. That is, letting $f_{j,i,e}[n]$ denote the number of VMs allocated to the task e of user i in BS j in slot n, then we have:

$$f_{j,i,e}[n] = \begin{cases} g_{j,i,e}, & n \in [v_{j,i,e}, v_{j,i,e} + u_i(g_{j,i,e}) - 1] \\ 0, & \text{else.} \end{cases} \tag{9}$$

Further, let $F_{j,i}[n]$ denote the total number of VMs allocated to user i in BS j in slot n, that is:

$$F_{j,i}[n] = \sum_{e:d_{i,e}=j} f_{j,i,e}[n]. \tag{10}$$

Then for each BS j and user i, the value of $F_{j,i}[n]$ is constrained by:

$$\sum_{i=1}^{I} F_{j,i}[n] \le F_j. \tag{11}$$

Additionally, the value of $g_{j,i,e}$ is upper bounded by:

$$g_{j,i,e} \le \min\{F_j, g_{\max}\}. \tag{12}$$

Now we describe the evolution of computing queues. For the computing queue of user i in BS j, let $Q_{j,i}[n]$ denote the number of tasks in the queue at the end of slot $n+1$, and $Y_{j,i}[n]$ denote the number of tasks leaving the queue during slot $n+1$. Using the variables defined previously, the value of $Y_{j,i}[n]$ can be calculated as:

$$Y_{j,i}[n] = \sum_{e:d_{i,e}=j} \mathbf{1}\{v_{j,i,e} + u_i(g_{j,i,e}) - 1 = n+1\}. \tag{13}$$

Recall that a total of $X_{i,j}[n]$ tasks leaves the ith communication queue at the end of slot n, as described in (7). We assume these tasks enter the ith computing queue in BS j at the end of slot $n+1$. Hence, we have:

$$Q_{j,i}[n] = \left(Q_{j,i}[n-1] - Y_{j,i}[n]\right)^+ + X_{i,j}[n], \tag{14}$$

where the expression of $X_{i,j}[n]$ is provided in (7).

Therefore, the computing scheduler must determine the values of the following variables: (i) $g_{j,i,e}$ and $v_{j,i,e}$ (i.e., VM scheduling).

III. THE PROPOSED SCHEDULING ALGORITHM

A. THE DELAY BASED LYAPUNOV FUNCTION

Firstly, we consider the communication delay. For convenience, let $X_i[n] = \sum_{j\in\Psi_i} X_{i,j}[n]$ for each user i and slot n. Let $A_{i,\text{tot}}[n] = \sum_{k=1}^{n} A_i[k]$ and $X_{i,\text{tot}}[n] = \sum_{k=1}^{n} X_i[k]$

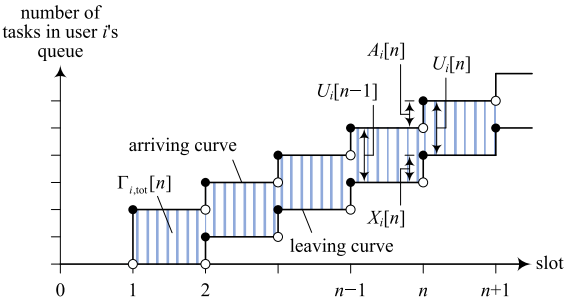


FIGURE 4. The delay model.

denote the total number of arrivals and departures until slot n, respectively. They are also known as the arriving and leaving curves in the literature [44]–[45], as illustrated in Fig. 4. Let $\Gamma_{i,\text{tot}}[n]$ denote the total area between these two curves up to slot $n+1$, which represents the total delay all tasks have experienced in the queue until slot $n+1$. Then we have,

$$\Gamma_{i,\text{tot}}[n] = U_i[1] \cdot T + \cdots + U_i[n] \cdot T, \tag{15}$$

which corresponds to the shadowed area in Fig. 4. Let $\Gamma_i[n]$ denote the time-average of $\Gamma_{i,\text{tot}}[n]$, that is, $\Gamma_i[n] = \Gamma_{i,\text{tot}}[n]/n$. Further, we express $\Gamma_i[n]$ as a virtual queue:

$$\Gamma_i[n] = \Gamma_i[n-1] - \varepsilon\Gamma_i[n-1] + \varepsilon U_i[n]T, \tag{16}$$

where $\varepsilon = 1/n$. Let $W_i[n]$ denote the normalized version of $\Gamma_i[n]$, that is, $W_i[n] = \Gamma_i[n]/T$. Then we have:

$$W_i[n] = W_i[n-1] - \varepsilon W_i[n-1] + \varepsilon U_i[n]. \tag{17}$$

Next, we consider the computing delay. Similar to (15), let $\Xi_{j,i,\text{tot}}[n]$ denote the total delay experienced by all tasks in the computing queue of user i in BS j until slot $n+1$:

$$\Xi_{j,i,\text{tot}}[n] = \sum_{k=1}^{n} Q_{j,i}[k]T. \tag{18}$$

Let $\Xi_{j,i}[n]$ denote the time-average of $\Xi_{j,i,\text{tot}}[n]$, that is, $\Xi_{j,i}[n] = \Xi_{j,i,\text{tot}}[n]/n$. Further, we express $\Xi_{j,i}[n]$ as a virtual queue:

$$\Xi_{j,i}[n] = \Xi_{j,i}[n-1] - \varepsilon\Xi_{j,i}[n-1] + \varepsilon Q_{j,i}[n]T. \tag{19}$$

Let $Z_{j,i}[n]$ denote the normalized version of $\Xi_{j,i}[n]$, that is, $Z_{j,i}[n] = \Xi_{j,i}[n]/T$. Then we have:

$$Z_{j,i}[n] = Z_{j,i}[n-1] - \varepsilon Z_{j,i}[n-1] + \varepsilon Q_{j,i}[n]. \tag{20}$$

Finally, we define the delay based Lyapunov function as:

$$L[n] = \sum_{i=1}^{I} W_i[n]^2 + \sum_{i=1}^{I}\sum_{j=1}^{J} Z_{j,i}[n]^2. \tag{21}$$

B. OPTIMIZATION PROBLEM FORMULATION

According to the Lyapunov optimization technique, the conditional Lyapunov drift $\Delta[n] = \text{E}\{L[n] - L[n-1]|\text{W}[n-1], \text{Z}[n-1]\}$, where $\text{E}\{\cdot\}$ is the expectation operation, $\text{W}[n-1] = [W_1[n-1], \ldots, W_I[n-1]]$, and $\text{Z}[n-1] = [Z_{1,1}[n-1], \ldots, Z_{J,I}[n-1]]$. Substituting (21), we have $\Delta[n] = \text{E}\{\sum_i \varepsilon^2 W_i[n-1]^2 + \sum_i \sum_j \varepsilon^2 Z_{j,i}[n-1]^2 + \sum_i \varepsilon^2 U_i[n]^2 + \sum_i \sum_j \varepsilon^2 Q_{j,i}[n]^2 - \sum_i 2\varepsilon W_i[n-1]^2 - \sum_i \sum_j 2\varepsilon Z_{j,i}[n-1]^2 + \sum_i 2\varepsilon(1-\varepsilon)W_i[n-1]U_i[n] + \sum_i \sum_j 2\varepsilon(1-\varepsilon)Z_{j,i}[n-1]Q_{j,i}[n]|\text{W}[n-1], \text{Z}[n-1]\}$, where the first six terms can be upper bounded by a constant under the expectation operation. According to the Lyapunov optimization technique, the above expression is minimized by the algorithm that obtains the values of $\text{W}[n-1]$ and $\text{Z}[n-1]$ and minimize $\sum_i W_i[n-1]U_i[n] + \sum_i \sum_j Z_{j,i}[n-1]Q_{j,i}[n]$. Further, substituting (8) and (14), this objective can be further upper bounded as $\sum_i W_i[n-1]U_i[n-1] + \sum_i W_i[n-1]A_i[n] + \sum_i \sum_j Z_{j,i}[n-1]Q_{j,i}[n-1] - \sum_i \sum_j W_i[n-1]X_{i,j}[n] + \sum_i \sum_j Z_{j,i}[n-1]X_{i,j}[n] - \sum_i \sum_j Z_{j,i}[n-1]Y_{j,i}[n]$, where the first three terms can also be upper bounded by a constant. So the final form of the optimization problem is:

$$\min \sum_{i=1}^{I}\sum_{j=1}^{J} \left(-W_i[n-1] + Z_{j,i}[n-1]\right) X_{i,j}[n] - \sum_{i=1}^{I}\sum_{j=1}^{J} Z_{j,i}[n-1]Y_{j,i}[n] \tag{22}$$

where $X_{i,j}[n]$ and $Y_{j,i}[n]$ are determined by scheduling variables $(x_{i,j,k}[n], p_{i,j,k}[n], d_{i,e}, n_{i,e})$ and $(g_{j,i,e}, v_{j,i,e})$ which must satisfy the constraints in (6)(3) and (11)(12), respectively.

C. SCHEDULING ALGORITHM

1) SUBCARRIER ALLOCATION, POWER CONTROL, AND BS SELECTION

The subproblem of optimizing communication resource scheduling variables can be written as:

$$\min \sum_{i=1}^{I}\sum_{j=1}^{J} \left(-W_i[n-1] + Z_{j,i}[n-1]\right) X_{i,j}[n], \tag{23}$$

where $X_{i,j}[n]$ is determined by scheduling variables $x_{i,j,k}[n]$, $p_{i,j,k}[n]$, $d_{i,e}$, and $n_{i,e}$ which must satisfy the constraints in (6)(3).

Before presenting the scheduling algorithm, we need to define the feasible user set. For any user i, firstly, if its communication queue is empty, we will not allocate any subcarrier to this user; secondly, if the constraint in (6) holds with equality, we will also not allocate any subcarrier to this user; thirdly, if the constraint in (3) holds with equality for each BS and each subcarrier, we will also not allocate any subcarrier to this user. Hence, we define the feasible user set as:

$$\begin{aligned} C[n] = \{i : U_i[n-1] > 0,\ & \textstyle\sum_{j=1}^{J}\sum_{k=1}^{K} p_{i,j,k}[n] < P_{\max}, \\ & \text{and exists } j \text{ and } k \text{ so that } \mathbf{1}\left\{x_{i,j,k}[n] > 0\right\} \\ & + \textstyle\sum_{i':j\in\Psi_{i'}} \mathbf{1}\left\{x_{i',j,k}[n] > 0\right\} = 0 \text{ where } j \in \Psi_i\}. \end{aligned} \tag{24}$$

Then, for each n, the communication resource scheduling algorithm (including subcarrier allocation, power control,

and BS selection) is executed, which is summarized in **Algorithm 1**. Initially, we set $x_{i,j,k}[n] = 0$, $p_{i,j,k}[n] = 0$, $P_{i,\text{rem}}[n] = P_{\max}$ which represents the remaining transmit power of user i, $B_{i,e} = D_i$ which represents the remaining bits of task e in the communication queue of user i,

$$\tilde{U}_i = U_i[n-1] - \sum_e \frac{B_{i,e}}{D_i}, \tag{25}$$

$$\tilde{W}_i = (1-\varepsilon)W_i[n-1] + \varepsilon \tilde{U}_i, \tag{26}$$

$$\tilde{Q}_{j,i} = Q_{j,i}[n-1] + \sum_{e:d_{i,e}=j} \mathbf{1}\{B_{i,e}=0\}, \tag{27}$$

and

$$\tilde{Z}_{j,i} = (1-\varepsilon)Z_{j,i}[n-1] + \varepsilon \tilde{Q}_{j,i}. \tag{28}$$

At the beginning (line 1), the value of $C[n]$ is calculated by substituting $\tilde{U}_i$ into (24). If $C[n]$ is empty, the algorithm stops. The first step is BS selection (line 4–12), in which the values of $d_{i,e}$ and $n_{i,e}$ are determined. The second step is subcarrier allocation (line 14–15), in which the value of $x_{i,j,k}[n]$ is determined. The third step is power control (line 17), in which the value of $p_{i,j,k}[n]$ is determined. There are two types of power control methods: for the uniform power control method, set:

$$p_{i^*,j^*,k^*}[n] = \min\{P_0, P_{i^*,\text{rem}}[n]\}; \tag{29}$$

for the reverse channel power control method, set:

$$p_{i^*,j^*,k^*}[n] = \min\left\{\frac{\sigma^2(2^{\frac{R_0}{TB}}-1)}{h_{i^*,j^*,k^*}[n]}, P_{i^*,\text{rem}}[n]\right\}. \tag{30}$$

Finally, update all relevant parameters (line 19–28).

2) VM SCHEDULING

The subproblem of optimizing computing resource scheduling variables can be written as:

$$\min -\sum_{i=1}^{I}\sum_{j=1}^{J} Z_{j,i}[n-1]Y_{j,i}[n], \tag{31}$$

where $Y_{j,i}[n]$ is determined by scheduling variables $g_{j,i,e}$ and $v_{j,i,e}$ which must satisfy the constraints in (11)(12).

Before presenting the scheduling algorithm, we need to define the feasible user set. For any user i in BS j, firstly, if its computing queue is empty, BS j will not allocate any new VM to user i; secondly, if the constraint in (11) holds with equality, BS j will also not allocate any new VM to user i. Hence, we define the feasible user set in BS j as:

$$H_j[n] = \{i : Q_{j,i}[n-1] > 0 \text{ and } F_{j,i}[n] + \sum\nolimits_{i' \neq i} F_{j,i'}[n] < F_j\}. \tag{32}$$

Additionally, for each task e in the computing queue of user i in BS j, let $\lambda_{j,i,e}[n]$ denote the normalized remaining execution slots of this task at the end of slot $n+1$, which is defined as:

$$\lambda_{j,i,e}[n] = \begin{cases} 1, & \delta_{j,i,e}[n] < 0 \\ 1 - \dfrac{\delta_{j,i,e}[n]+1}{u_i(g_{j,i,e})}, & \delta_{j,i,e}[n] \in [0, u_i(g_{j,i,e})-1] \\ 0, & \delta_{j,i,e}[n] \geq u_i(g_{j,i,e}), \end{cases} \tag{33}$$

where

$$\delta_{j,i,e}[n] = n + 1 - v_{j,i,e}. \tag{34}$$

Finally, let $L_{j,i,\text{new}}[n]$ denote the set of tasks in the ith computing queue in BS j which has not been allocated any VM yet until the beginning of slot $n+1$.

Then, for each n in BS j, the computing resource scheduling algorithm (i.e., VM scheduling) is executed, which is

Algorithm 1 Communication Resource Scheduling for Each n

Input: The initial values of $x_{i,j,k}[n]$, $p_{i,j,k}[n]$, $P_{i,\text{rem}}[n]$, $B_{i,e}$, $\tilde{U}_i$, $\tilde{W}_i$, $\tilde{Q}_{j,i}$, and $\tilde{Z}_{j,i}$ set according to (25)-(28);
Output: The values of $x_{i,j,k}[n]$, $p_{i,j,k}[n]$, $d_{i,e}$, and $n_{i,e}$.

1: Calculate $C[n]$ by substituting $\tilde{U}_i$ into (24);
2: **while** $C[n]$ is not empty **do**
3: // Step 1: BS selection
4: Select $i^* = \arg\max \tilde{W}_i$ over all $i \in C[n]$;
5: Select e^* which is the task of user i^* arriving earliest with $B_{i^*,e^*} > 0$;
6: **if** d_{i^*,e^*} and n_{i^*,e^*} do not exist **then**
7: Select $j^* = \arg\min \tilde{Z}_{j,i^*}$ over all $j \in \Psi_{i^*}$;
8: Set $d_{i^*,e^*} = j^*$;
9: Set $n_{i^*,e^*} = n$;
10: **else**
11: Set $j^* = d_{i^*,e^*}$;
12: **end if**
13: // Step 2: Subcarrier allocation
14: Select $k^* = \arg\max h_{i^*,j^*,k}[n]$ over all subcarriers which are available to the link from user i^* to BS j^*;
15: Set $x_{i^*,j^*,k^*}[n] = e^*$;
16: // Step 3: Power control
17: Set $p_{i^*,j^*,k^*}[n]$ according to (29) or (30);
18: // Update parameters
19: Update $P_{i^*,\text{rem}}[n] \leftarrow P_{i^*,\text{rem}}[n] - p_{i^*,j^*,k^*}[n]$;
20: Calculate $R_{i^*,j^*,k^*}[n]$ according to (1);
21: Update $B_{i^*,e^*} \leftarrow \max(0, B_{i^*,e^*} - R_{i^*,j^*,k^*}[n])$;
22: Update $\tilde{U}_{i^*}$ according to (25);
23: Update $\tilde{W}_{i^*}$ according to (26);
24: **if** $B_{i^*,e^*} = 0$ **then**
25: Update $\tilde{Q}_{j^*,i^*}$ according to (27);
26: Update $\tilde{Z}_{j^*,i^*}$ according to (28);
27: **end if**
28: Update $C[n]$;
29: **end while**

Algorithm 2 Computing Resource Scheduling for Each n

Input: The initial values of $\hat{Q}_{j,i}$, $\hat{Z}_{j,i}$, $F_{j,i,\text{ini}}$, $F_{j,i,\text{new}}$, and $F_{j,i}$ set according to (35)-(38);
Output: The values of $g_{j,i,e}$ and $v_{j,i,e}$.
1: Calculate $H_j[n]$ by substituting $\hat{Q}_{j,i}$ into (32);
2: **while** $H_j[n]$ is not empty **do**
3: // VM scheduling
4: Select $i^* = \arg\min(-\tilde{Z}_{j,i})$ over all $i \in H_j[n]$;
5: Select a task e^* from $L_{j,i^*,\text{new}}[n]$;
6: **if** There exists $e^* \in L_{j,i^*,\text{new}}[n]$ with $g_{j,i^*,e^*} > 0$ **then**
7: Update $g_{j,i^*,e^*} \leftarrow g_{j,i^*,e^*} + 1$;
8: **else**
9: Pick the task $e^* \in L_{j,i^*,\text{new}}[n]$ which arrives earliest;
10: Set $g_{j,i^*,e^*} = 1$;
11: Set $v_{j,i^*,e^*} = n + 1$;
12: **end if**
13: **if** $g_{j,i^*,e^*} = g_{\max}$ **then**
14: Remove this task from $L_{j,i^*,\text{new}}[n]$;
15: **end if**
16: // Update parameters
17: Update $F_{j,i^*,\text{new}} \leftarrow F_{j,i^*,\text{new}} + 1$;
18: Update F_{j,i^*} according to (37);
19: Update $\lambda_{j,i^*,e^*}[n]$ according to (33);
20: Update $\hat{Q}_{j,i^*}$ according to (35);
21: Update $\hat{Z}_{j,i^*}$ according to (36);
22: Update $H_j[n]$;
23: **end while**

summarized in **Algorithm 2**. Initially, we set

$$\hat{Q}_{j,i} = Q_{j,i}[n-1] + \sum_e \lambda_{j,i,e}[n], \tag{35}$$

$$\hat{Z}_{j,i} = (1-\varepsilon)Z_{j,i}[n-1] + \varepsilon \hat{Q}_{j,i}, \tag{36}$$

and

$$F_{j,i} = F_{j,i,\text{ini}} + F_{j,i,\text{new}}, \tag{37}$$

where $F_{j,i,\text{new}} = 0$ and

$$F_{j,i,\text{ini}} = \sum\nolimits_e g_{j,i,e} \cdot \mathbf{1}\{\delta_{j,i,e}[n] \in [0, u_i(g_{j,i,e}) - 1]\}. \tag{38}$$

At the beginning (line 1), the value of $H_j[n]$ is calculated by substituting $\hat{Q}_{j,i}$ into (32). If $H_j[n]$ is empty, the algorithm stops. The next step is VM scheduling (line 4–15), in which the values of $g_{j,i,e}$ and $v_{j,i,e}$ are determined. Finally, update all relevant parameters (line 16–22).

The analysis of the computation complexity of the proposed scheduling algorithm is as follows. The proposed scheduling algorithm has two parts **Algorithm 1** and **Algorithm 2**. We first discuss the first part. For this part, the key steps are to calculate and compare the values of $\tilde{W}_i$, $\tilde{Z}_{j,i^*}$, and $h_{i^*,j^*,k}$ over all feasible i, j, and k. Thus, the computational complexity of these key steps is $O(I+J+K)$. Further, according to the procedure in **Algorithm 1**, these key steps will be executed until there is no assignable subcarrier. Since the number of subcarriers is limited, the number of times of executing these key steps will be no more than a constant K. Therefore, the computational complexity of the first part of the proposed scheduling algorithm is $O(IK + JK + K^2)$. We then discuss the second part. For this part, for each BS j, the key step is to calculate and compare the value of $\tilde{Z}_{j,i}$ over all feasible i. Thus, the computational complexity of this key step is $O(I)$ for each BS. Further, according to the procedure in **Algorithm 2**, this key step will be executed until there is no assignable VM. Since the number of VMs in each BS is limited, the number of times of executing this step will be no more than a constant $F \triangleq \max_j F_j$. Therefore, the computational complexity of the second part of the proposed scheduling algorithm is $O(IF)$ for each BS. Since there are a total of J BSs, the total computation complexity of the second part is $O(IJF)$. Putting together, the overall computational complexity of the proposed scheduling algorithm is $O(IK + JK + K^2 + IJF)$.

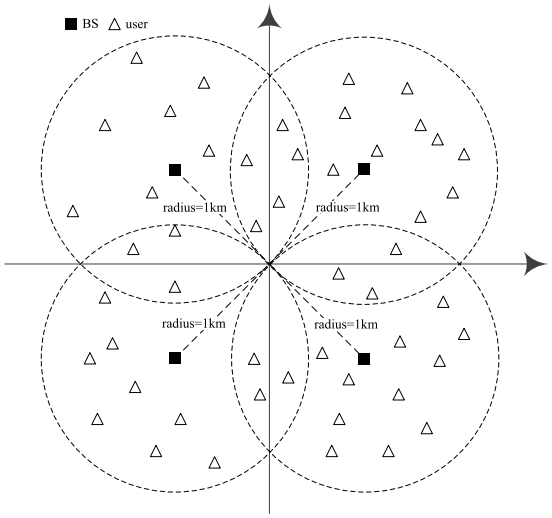


FIGURE 5. The scenario used in simulations.

IV. PERFORMANCE EVALUATION

A. SIMULATION PARAMETERS

Consider a multi-cell system, as shown in Fig. 5. There are $J = 4$ BSs and the radius of each cell is 1km. Thus, the geographical positions of these BSs are: $(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}})$, $(-\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}})$, $(-\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}})$, and $(\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}})$ in km. A total of $I = 50$ users are randomly distributed over the area covered by these BSs. For each user i, assume his tasks arrive according to a Poisson process with the average inter-arrival time of T_i seconds. The slot duration T is 1ms. Unless otherwise stated, we set $T_i = 10$ slots. Let d_{ij} represents the distance between user i and BS j. In the simulation, we let user i be a neighbor of BS j, that is, $j \in \Psi_i$, if $d_{ij} < 1.1 \cdot \max_{i'} \min_{j'} d_{i'j'}$. Thus the set Ψ_i can be determined for each user i. The maximum transmit power of each user is 400mW. For the uniform power control, the value of P_0 is set to be 200mW; for the reverse channel power control, the value of R_0 is set to be 200b. The value of

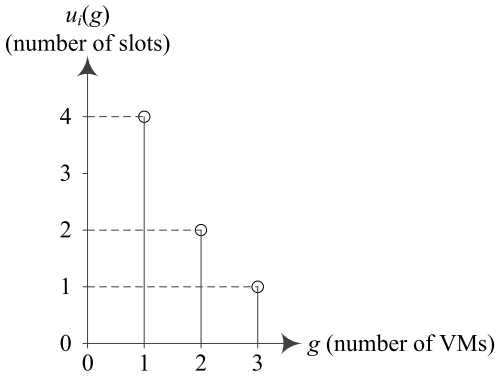


FIGURE 6. The utility function used in simulations.

P_{in} is set to be lower than the maximum transmit power by -120dB. The link gain between the BS and a user is given as the product of path loss, shadowing, and fast fading effect. For path loss, we adopt the modified Hata urban propagation model [46], which equals to $122 + 38\log_{10}(d)$, where d (in km) is the distance between the BS and a user, and every user has the same path loss value within the distance of 50m. The shadowing component follows lognormal distribution with mean value of 0dB and standard deviation of 8dB. The fast fading follows Rayleigh distribution with mean of 2. For each application of user i, the value of D_i is set to be 1000 bits. The bandwidth of each subcarrier is 100kHz. Unless otherwise stated, the number of subcarriers is set as $K = 16$. For the utility function, the value of $g_{\max}$ is set as 3 and the function $u_i(g)$ is set as: $u_i(1) = 4$, $u_i(2) = 2$, and $u_i(3) = 1$, as shown in Fig. 6. The value of F_j is set as $\rho_j \times 10$, where ρ_j is a coefficient of BS j in the range of $[0, 1]$. Unless otherwise stated, for even-numbered j, the value of ρ_j is set as 1; for odd-numbered j, the value of ρ_j is set as 0.5. Main parameters used in simulations are summarized in Tab. 2. The performance considered in this paper is the total delay which is the sum of the average communication delay and computing delay. A total of four different scheduling algorithms are considered in simulations, which are summarized in Tab. 3. Among them, the outline of the traditional queue length based scheduling algorithm can be found in the Appendix. Given the parameter configuration, the simulation experiment is repeated 100 times and then averaged as the final result.

B. SIMULATION RESULTS

Fig. 7 shows the convergence performance of all the considered scheduling algorithms. In this set of simulation experiments, the number of subcarriers is set to $K = 16$. The convergence of the average communication delay is shown in Fig. 7(a). As can be seen from this figure, the average communication delay of each considered scheduling algorithm can quickly converge to a stable value. For example, for the 'Dly+Uni' scheduling algorithm, its average communication delay has been taken as 2.08ms after 100 slots, while the value after 2000 slots is stable at about 2.26ms, with a deviation of only -7%. On the other hand, the convergence of the average computing delay is shown in Fig. 7(b). The curves in this figure show that the convergence performance of the delay based scheduling algorithms is better than that of the queue length based scheduling algorithms. For example, the 'Dly+Uni' scheduling algorithm is stable after about 100 slots, and the 'Que+Uni' scheduling algorithm needs to be stabilized after about 600 slots. Therefore, it can be concluded from these figures that all the considered scheduling algorithms are convergent, and the delay based scheduling algorithms can converge faster than the queue length based scheduling algorithms.

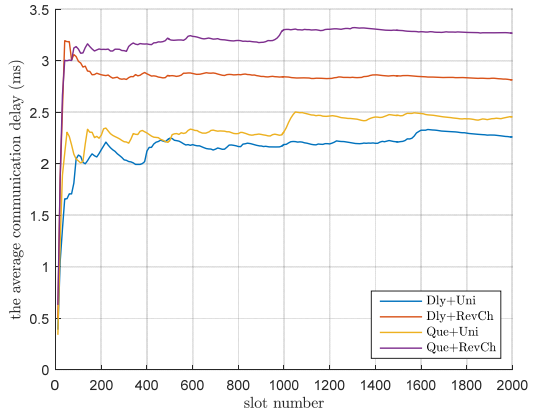
Fig. 8 shows the probability distribution of the delays of all the considered scheduling algorithms. In this set of simulation experiments, the number of subcarriers is also set to $K = 16$. The probability distribution of the communication delay is shown in Fig. 8(a). From this figure, it can be seen that for each scheduling algorithm, the value of the communication delay experienced by transferring each computing request is within a relatively narrow range. For example, for the 'Dly+Uni' scheduling algorithm, the communication delay of each computing request is 2ms, 3ms, 4ms, 5ms with a probability of 43.18%, 29.45%, 12.73%, and 6.21%, respectively. Therefore, the communication delay for each computing request takes a value within the range $[2, 4]$ms with a probability of 85.38%, and then takes a value within the range $[2, 5]$ms with a probability of 91.60%. Therefore,

TABLE 2. Main parameters used in simulations.

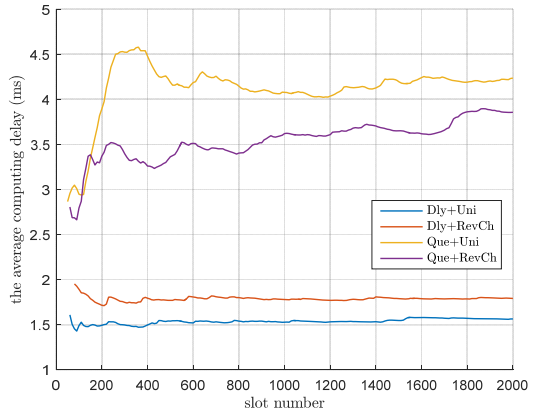
Parameter	Value
I	50
J	4 or 3
K	$16 \sim 30$
B	100kHz
T	1ms
T_i	$10 \sim 16$ms
D_i	1000b
$P_{\max}$	400mW
P_0	200mW
R_0	200b
F_j	$3 \sim 10$
$u_i(g)$	$[4, 2, 1]$ (as shown in Fig. 6)
$g_{\max}$	3
P_{in}	lower than $P_{\max}$ by -120dB
channel model	path loss: the modified Hata urban propagation model [46]; shadow: lognormal distribution with mean of 0dB and standard deviation of 8dB; fast fading: Rayleigh distribution with mean of 2

TABLE 3. Scheduling algorithms used in simulations.

Algorithm	Description
'Dly+Uni'	the proposed delay based scheduling algorithm + uniform power control method
'Dly+RevCh'	the proposed delay based scheduling algorithm + reverse channel power control method
'Que+Uni'	the traditional queue length based scheduling algorithm + uniform power control method
'Que+RevCh'	the traditional queue length based scheduling algorithm + reverse channel power control method

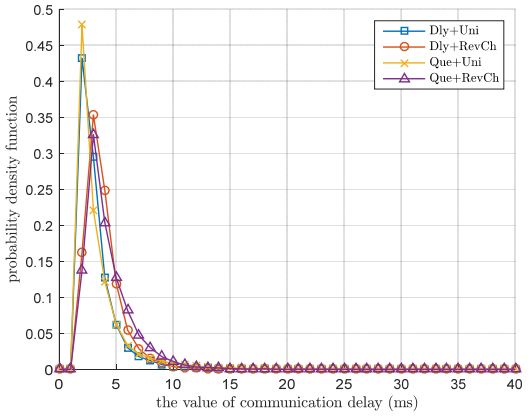


(a)

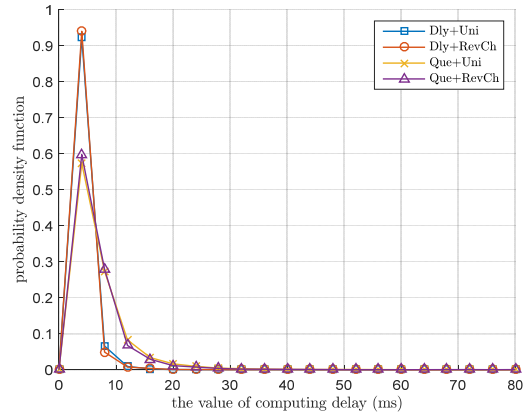


(b)

FIGURE 7. The convergence performance of scheduling algorithms: (a) the average communication delay and (b) the average computing delay.



(a)



(b)

FIGURE 8. The probability distribution of the delays: (a) the communication delay and (b) the computing delay.

if the threshold is 90% probability, the communication delay for each computing request will fluctuate within a range of intervals where 2.26ms for the mean and 0.90ms for the variance. On the other hand, the probability distribution of the computing delay is shown in Fig. 8(b). It can be seen from this figure that the ranges of the computing delay of the delay based scheduling algorithms is narrower than those of the queue length based scheduling algorithms. For example, most of the computing delays of the 'Dly+Uni' scheduling algorithm fall within the range [1, 10]ms, while most of the computing delays of the 'Que+Uni' scheduling algorithm fall within the range [1, 20]ms. Therefore, it can be concluded from these figures that the delay variance of the delay based scheduling algorithms is smaller than that of the scheduling algorithms based on the queue length.

Fig. 9 shows the simulation results with different K (i.e., the total number of subcarriers in the system) for different scheduling algorithms. In this set of simulation experiments, we set $\rho_j = 0.5$ for even-numbered j and change K from 16 to 30. Firstly, we can observe the total delay decreases with the increase of K. For example, for the proposed delay based scheduling algorithm with uniform power

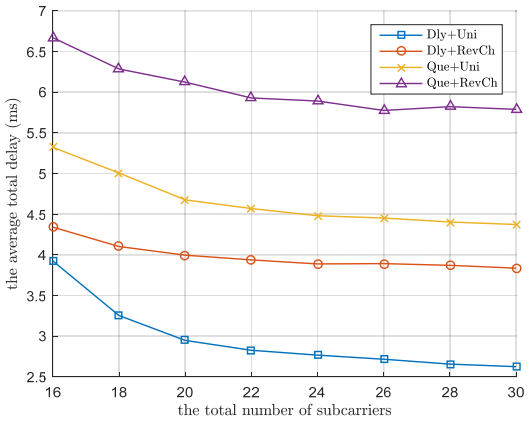


FIGURE 9. Impact of the total number of subcarriers.

control (i.e., the 'Dly+Uni' curve with square mark), when K increases from 16 to 30, the average total delay decreases from 3.92ms to 2.62ms. Additionally, when the value of K is large, all curves become flat. This is due to the maximum transmit power of each user is limited so that the redundant subcarrier resource cannot be fully utilized by users. Further,

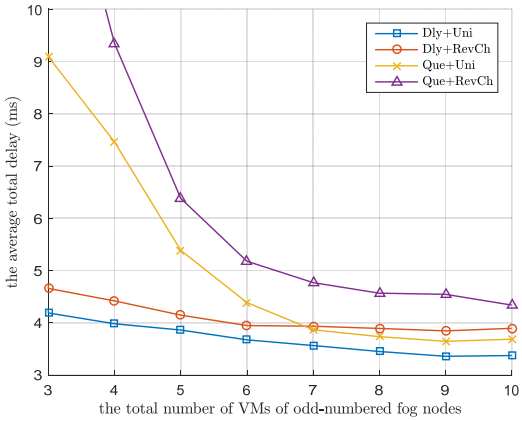


FIGURE 10. Impact of the amount of computing resources.

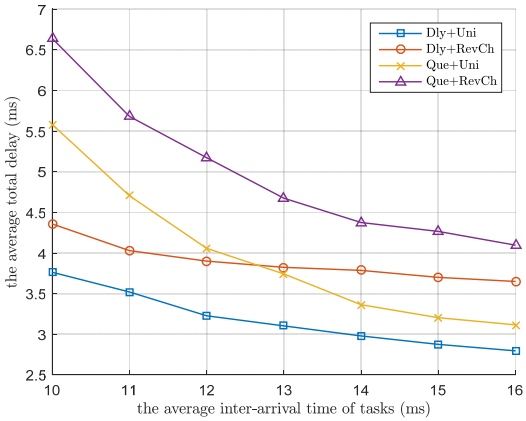


FIGURE 11. Impact of the average inter-arrival time of tasks.

we can observe that, the delay of the 'Dly+Uni' scheduling algorithm is better than that of the other scheduling algorithms when the system is stable. For example, when $K = 20$, the average total delay of the 'Que+Uni' algorithm is 4.67ms, while the average total delay of the 'Dly+Uni' scheduling algorithm is 2.94ms, with a 37.04% off. This is due to that the proposed scheduling algorithm controls delay directly, while the traditional one does not. Additionally, we can observe that, the delay of the scheduling algorithm with uniform power control method is better than that of the scheduling algorithm with reverse channel power control method. For example, when $K = 20$, the average total delay of the 'Dly+RevCh' scheduling algorithm is 3.99ms, while the average total delay of the 'Dly+Uni' scheduling algorithm is 2.94ms, with a 26.31% off. This is due to that the revers channel power control waste more power when the channel is not good than the uniform power control method. Therefore, we can conclude that the delay performance of the 'Dly+Uni' scheduling algorithm is better than the traditional queue-length based one.

Fig. 10 shows the simulation results for different values of the number of VMs F_j. In this set of simulation experiments, we change the value of F_j for odd-numbered j from 3 to 10 (i.e., change the value of ρ_j for odd-numbered j from 0.3 to 1). We can observe that, with the decrease of F_j for odd-numbered j, the heterogeneity of BSs increase, then the differences between the 'Dly+Uni' scheduling algorithm and other scheduling algorithms increase. For example, when F_j for odd-numbered j is 4, the average total delay of the 'Que+Uni' scheduling algorithm is 7.45ms, while the average total delay of the 'Dly+Uni' scheduling algorithm is 3.98ms, with a 46.57% off. It can be noted that the 'Dly+Uni' scheduling algorithm has a very similar performance when compared with the queue-based one for higher values of total number of VMs. For this observation, we would like to emphasize that, the delay performance of the 'Dly+Uni' scheduling algorithm is similar to that of the queue-based one only when the computing resources (i.e., the number of VMs) are fully supplied. In this case, since the computing resources are fully supplied, it is normal for the existing queue-based scheduling algorithm to achieve good performance, and there is no strong need to further improve it. However, when the computing resources are not fully supplied, the delay performance of the existing queue-based scheduling algorithm begins to increase. At this point, it is very meaningful to study how to improve the existing queue-based scheduling algorithm to reduce the delay. In fact, as shown in Fig. 10, when the computing resources are not fully supplied, the superiority of the 'Dly+Uni' scheduling algorithm to the queue-based one becomes much more significant. Therefore, although the right side of the performance curve of the 'Dly+Uni' scheduling algorithm is similar to the queue-based one, this does not prevent us from getting the conclusion that the 'Dly+Uni' scheduling algorithm is an improvement to the queue-based one.

Fig. 11 shows the simulation results for different values of T_i (i.e., the average inter-arrival time of tasks). In this set of simulation experiments, we change the value of T_i from 10 to 16 slots. We can observe that, with the increase of T_i, the delay decreases. For example, for the proposed 'Dly+Uni' scheduling algorithm, when T_i increases from 10 to 16, the average total delay decreases from 3.76ms to 2.79ms. Further, we can observe that, the delay of the 'Dly+Uni' scheduling algorithm is always better than other algorithms. For example, when $T_i = 10$, the average total delay of the 'Que+Uni' scheduling algorithm is 5.58ms, while the average total delay of the 'Dly+Uni' scheduling algorithm is 3.76ms, with a 32.61% off. This is also due to that the 'Dly+Uni' scheduling algorithm controls delay directly, while the traditional queue-length based one does not.

Fig. 12 shows the simulation results for different number of BSs. In this set of simulation experiments, when changing the number of BSs, for fair comparison, the number of users in the system is always set to 30, the number of subcarriers is always set to 30, and then the geographic coordinates of the three base stations are set to $(0, 1)$, $(\frac{\sqrt{3}}{2}, -\frac{1}{2})$, and $(-\frac{\sqrt{3}}{2}, -\frac{1}{2})$ in km, respectively. As can be seen from the results in this figure, given the number of users and the number of subcarriers, the average total delay in the case of three base stations is significantly longer than the average

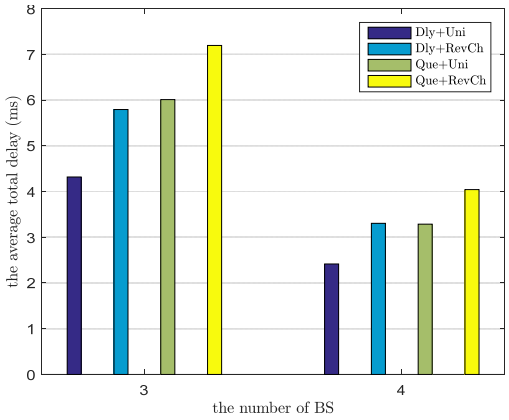


FIGURE 12. Impact of the number of BSs.

total delay in the case of four base stations. For example, for the 'Dly+Uni' scheduling algorithm, the average total delay in the case of three base stations is 4.31ms, and the average total delay in the case of four base stations is shortened to 2.41ms, a decrease of 26.49%. Similarly, for the 'Que+Uni' scheduling algorithm, the average total delay in the case of three base stations is 6.00ms, and the average total delay in the case of four base stations is shortened to 3.28ms, a decrease of 28.16%. Further, it can be seen from this figure that the delay performance of the 'Dly+Uni' scheduling algorithm is always superior to the delay performance of the remaining scheduling algorithms regardless of the number of base stations. Therefore, the results of this set of simulation experiments again verify the performance superiority of the proposed delay based 'Dly+Uni' scheduling algorithm.

V. CONCLUSION AND FUTURE WORK

This paper studies the problem of how to schedule subcarrier, BS, power, and VM for users in multi-cell cellular computing systems. Firstly, the model of the communication delay and computing delay was established and expressed as virtual queues. Then, using the Lyapunov optimization method, a novel joint subcarrier allocation, BS selection, power control, and VM scheduling algorithm was proposed to stabilize the virtual delay queues. Simulation results showed that the delay performance of the proposed scheduling algorithm is better than that of the traditional queue length based one.

In our future work, we will extend this work from the following aspects. Firstly, we will extend the definition of the utility function to include situations where multiple computation tasks can be completed in a single slot. Secondly, while minimizing delay, we will also consider how to design scheduling algorithm with the goal of minimizing energy consumption. Thirdly, we will consider the situation where computing task execution may fail. Fourthly, we will consider the situation where predictive caching of data is allowed. Finally, we will extend to the situation where users are constantly moving in the cellular system and solve the corresponding wireless computing handover problem.

APPENDIX
THE TRADITIONAL ALGORITHM

The traditional Lyapunov optimization based scheduling algorithm is presented here which is queue length based. In the literature, such traditional scheduling algorithms can be found in [26]–[33]. These algorithms vary greatly in detail due to different optimization goals. We extract the commonalities of these algorithms, which are summarized as follows and used as the benchmark in this work. Firstly, define the queue length based Lyapunov function as:

$$L[n] = \sum_{i=1}^{I} U_i[n]^2 + \sum_{j=1}^{J} \sum_{j=1}^{J} Q_{j,i}[n]^2. \quad (39)$$

Then, calculate the conditional Lyapunov drift $\Delta[n] = \mathrm{E}\{L[n] - L[n-1] | \mathrm{U}[n-1], \mathrm{Q}[n-1]\}$, where $U[n-1] = [U_1[n-1], \ldots, U_I[n-1]]$ and $Q[n-1] = [Q_{1,1}[n-1], \ldots, Q_{J,I}[n-1]]$. After similar derivations [16], the algorithm solves the following optimization problem for each n:

$$\min \sum_{i=1}^{I} \sum_{j=1}^{J} \left(-U_i[n-1] + Q_{j,i}[n-1]\right) X_{i,j}[n] - \sum_{i=1}^{I} \sum_{j=1}^{J} Q_{j,i}[n-1] Y_{j,i}[n], \quad (40)$$

where $X_{i,j}[n]$ and $Y_{j,i}[n]$ are determined by scheduling variables $(x_{i,j,k}[n], p_{i,j,k}[n], d_{i,e}, n_{i,e})$ and $(g_{j,i,e}, v_{j,i,e})$ which must satisfy the constraints in (6)(3) and (11)(12). We use the same procedure proposed in the last section to solve the optimization problem in (40) except that the queue length is used instead.

ACKNOWLEDGMENT

The authors thank reviewers for pointing out future research directions to them.

REFERENCES

[1] P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture and computation offloading," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1628–1656, 3rd Quart., 2017.
[2] 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.
[3] S. Wang, X. Zhang, Y. Zhang, L. Wang, J. Yang, and W. Wang, "A survey on mobile edge networks: Convergence of computing, caching and communications," *IEEE Access*, vol. 5, pp. 6757–6779, 2017.
[4] T. X. Tran, A. Hajisami, P. Pandey, and D. Pompili, "Collaborative mobile edge computing in 5G networks: New paradigms, scenarios, and challenges," *IEEE Commun. Mag.*, vol. 55, no. 4, pp. 54–61, Apr. 2017.
[5] Z. Jiang, C. Xu, J. Guan, and D. Niyato, "Stochastic profit maximization of service provider in millimeter-wave high-speed railway networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 5, pp. 5059–5075, May 2019.
[6] Z. Jiang, S. Chen, S. Zhou, and Z. Niu, "Joint user scheduling and beam selection optimization for beam-based massive MIMO downlinks," *IEEE Trans. Wireless Commun.*, vol. 17, no. 4, pp. 2190–2204, Apr. 2018.
[7] M. Bennis, M. Debbah, and H. V. Poor, "Ultrareliable and low-latency wireless communication: Tail, risk, and scale," *Proc. IEEE*, vol. 106, no. 10, pp. 1834–1853, Oct. 2018.
[8] J. Ren, G. Yu, Y. He, and G. Y. Li, "Collaborative cloud and edge computing for latency minimization," *IEEE Trans. Veh. Technol.*, vol. 68, no. 5, pp. 5031–5044, May 2019.

[9] T. Q. Dinh, Q. D. La, T. Q. S. Quek, and H. Shin, "Learning for computation offloading in mobile edge computing," *IEEE Trans. Commun.*, vol. 66, no. 12, pp. 6353–6367, Dec. 2018.

[10] 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.

[11] F. Guo, H. Zhang, H. Ji, X. Li, and V. C. M. Leung, "An efficient computation offloading management scheme in the densely deployed small cell networks with mobile edge computing," *IEEE/ACM Trans. Netw.*, vol. 26, no. 6, pp. 2651–2664, Dec. 2018.

[12] J. Ren, G. Yu, Y. Cai, and Y. He, "Latency optimization for resource allocation in mobile-edge computation offloading," *IEEE Trans. Wireless Commun.*, vol. 17, no. 8, pp. 5506–5519, Aug. 2018.

[13] Y. Wu, K. Ni, C. Zhang, L. P. Qian, and D. H. K. Tsang, "Noma-assisted multi-access mobile edge computing: A joint optimization of computation offloading and time allocation," *IEEE Trans. Veh. Technol.*, vol. 67, no. 12, pp. 12244–12258, Dec. 2018.

[14] J. Zhou, X. Zhang, and W. Wang, "Joint resource allocation and user association for heterogeneous services in multi-access edge computing networks," *IEEE Access*, vol. 7, pp. 12272–12282, 2019.

[15] X. Yu, M. Guan, M. Liao, and X. Fan, "Pre-migration of vehicle to network services based on priority in mobile edge computing," *IEEE Access*, vol. 7, pp. 3722–3730, 2019.

[16] A. Khalili, S. Zarandi, and M. Rasti, "Joint resource allocation and offloading decision in mobile edge computing," *IEEE Commun. Lett.*, vol. 23, no. 4, pp. 684–687, Apr. 2019.

[17] T. X. Tran and D. Pompili, "Joint task offloading and resource allocation for multi-server mobile-edge computing networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 1, pp. 856–868, Jan. 2019.

[18] Y. Zhang, X. Lan, Y. Li, L. Cai, and J. Pan, "Efficient computation resource management in mobile edge-cloud computing," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 3455–3466, Apr. 2019.

[19] B. P. Rimal, D. P. Van, and M. Maier, "Cloudlet enhanced fiber-wireless access networks for mobile-edge computing," *IEEE Trans. Wireless Commun.*, vol. 16, no. 6, pp. 3601–3618, Jun. 2017.

[20] S. Guo, D. Wu, H. Zhang, and D. Yuan, "Resource modeling and scheduling for mobile edge computing: A service provider's perspective," *IEEE Access*, vol. 6, pp. 35611–35623, 2018.

[21] L. Chen, S. Zhou, and J. Xu, "Computation peer offloading for energy-constrained mobile edge computing in small-cell networks," *IEEE/ACM Trans. Netw.*, vol. 26, no. 4, pp. 1619–1632, Aug. 2018.

[22] S. Ko, K. Han, and K. Huang, "Wireless networks for mobile edge computing: Spatial modeling and latency analysis," *IEEE Trans. Wireless Commun.*, vol. 17, no. 8, pp. 5225–5240, Aug. 2018.

[23] Q. Fan and N. Ansari, "Application aware workload allocation for edge computing-based IoT," *IEEE Internet Things J.*, vol. 5, no. 3, pp. 2146–2153, Jun. 2018.

[24] L. Yang, H. Zhang, M. Li, J. Guo, and H. Ji, "Mobile edge computing empowered energy efficient task offloading in 5G," *IEEE Trans. Veh. Technol.*, vol. 67, no. 7, pp. 6398–6409, Jul. 2018.

[25] M. Liu, F. R. Yu, Y. Teng, V. C. M. Leung, and M. Song, "Distributed resource allocation in blockchain-based video streaming systems with mobile edge computing," *IEEE Trans. Wireless Commun.*, vol. 18, no. 1, pp. 695–708, Jan. 2019.

[26] X. Lyu, W. Ni, H. Tian, R. P. Liu, X. Wang, and G. B. Giannakis, and A. Paulraj, "Optimal schedule of mobile edge computing for Internet of Things using partial information," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 11, pp. 2606–2615, Nov. 2017.

[27] 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.

[28] H. Feng, J. Llorca, A. M. Tulino, and A. F. Molisch, "Optimal control of wireless computing networks," *IEEE Trans. Wireless Commun.*, vol. 17, no. 12, pp. 8283–8298, Dec. 2018.

[29] X. Yang, Z. Chen, K. Li, Y. Sun, N. Liu, W. Xie, and Y. Zhao, "Communication-constrained mobile edge computing systems for wireless virtual reality: Scheduling and tradeoff," *IEEE Access*, vol. 6, pp. 16665–16677, 2018.

[30] Y. Kim, J. Kwak, and S. Chong, "Dual-side optimization for cost-delay tradeoff in mobile edge computing," *IEEE Trans. Veh. Technol.*, vol. 67, no. 2, pp. 1765–1781, Feb. 2018.

[31] J. Du, F. R. Yu, X. Chu, J. Feng, and G. Lu, "Computation offloading and resource allocation in vehicular networks based on dual-side cost minimization," *IEEE Trans. Veh. Technol.*, vol. 68, no. 2, pp. 1079–1092, Feb. 2019.

[32] Y. Kim, H.-W. Lee, and S. Chong, "Mobile computation offloading for application throughput fairness and energy efficiency," *IEEE Trans. Wireless Commun.*, vol. 18, no. 1, pp. 3–19, Jan. 2019.

[33] C.-F. Liu, M. Bennis, M. Debbah, and H. V. Poor, "Dynamic task offloading and resource allocation for ultra-reliable low-latency edge computing," *IEEE Trans. Commun.*, vol. 67, no. 6, pp. 4132–4150, Jun. 2019.

[34] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," *IEEE Trans. Autom. Control*, vol. 37, no. 12, pp. 1936–1948, Dec. 1992.

[35] M. J. Neely, *Stochastic Network Optimization With Application to Communication and Queueing Systems*. San Rafael, CA, USA: Morgan & Claypool, 2010.

[36] Y. Zhang, P. Du, J. Wang, T. Ba, R. Ding, and N. Xin, "Resource scheduling for delay minimization in multi-server cellular edge computing systems," *IEEE Access*, vol. 7, pp. 86265–86273, 2019.

[37] G. J. Pottie, "System design choices in personal communications," *IEEE Pers. Commun.*, vol. 2, no. 5, pp. 50–67, Oct. 1995.

[38] C. Rose, S. Ulukus, and R. D. Yates, "Wireless systems and interference avoidance," *IEEE Trans. Wireless Commun.*, vol. 1, no. 3, pp. 415–428, Jul. 2002.

[39] D. López-Pérez, A. Valcarce, G. de la Roche, and J. Zhang, "OFDMA femtocells: A roadmap on interference avoidance," *IEEE Commun. Mag.*, vol. 47, no. 9, pp. 41–48, Sep. 2009.

[40] K. Son, S. Chong, and G. Veciana, "Dynamic association for load balancing and interference avoidance in multi-cell networks," *IEEE Trans. Wireless Commun.*, vol. 8, no. 7, pp. 3566–3576, Jul. 2009.

[41] Z. Hu, Z. Chen, R. Susitaival, I.-F. Fu, S. K. Baghel, and P. Dayal, "Interference avoidance for in-device coexistence in 3GPP LTE-Advanced: Challenges and solutions," *IEEE Commun. Mag.*, vol. 50, no. 11, pp. 60–67, Nov. 2012.

[42] N. Gresset, H. Halbauer, J. Koppenborg, W. Zirwas, and H. Khanfir, "Interference-avoidance techniques: Improving ubiquitous user experience," *IEEE Veh. Technol. Mag.*, vol. 7, no. 4, pp. 37–45, Dec. 2012.

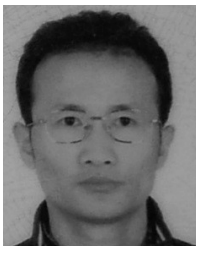
[43] C. Kosta, B. Hunt, A. U. Quddus, and R. Tafazolli, "On interference avoidance through inter-cell interference coordination (ICIC) based on OFDMA mobile systems," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 3, pp. 973–995, 3rd Quart., 2013.

[44] L. Kleinrock, *Queueing Systems: Theory*, vol. I. New York, NY, USA: Wiley, 1975.

[45] J.-Y. Le Boudec and P. Thiran, *Network Calculus: A Theory of Deterministic Queuing Systems for the Internet*. Berlin, Germany: Springer, 2001.

[46] K. Kim, Y. Han, and S.-L. Kim, "Joint subcarrier and power allocation in uplink OFDMA systems," *IEEE Commun. Lett.*, vol. 9, no. 6, pp. 526–528, Jun. 2005.

YUAN ZHANG (M'16) received the Ph.D. degree from Southeast University, Nanjing, China, in 2004. Since 2005, he has been with the National Mobile Communications Research Laboratory, Southeast University, as a Faculty Member. His research interests include areas of wireless communications, networking, and computing. He has co-received the Best Paper Award from ICC 2014.



PENG DU received the Ph.D. degree from Southeast University, Nanjing, China, in 2004. Since 2004, he became a member of the Technical Staff with Lucent Technologies, Nanjing, China. Since 2008, he has been with the Nanjing University of Posts and Telecommunications, Nanjing. His research interests include wireless communication system design and digital signal processing.