Delay-Sensitive Multi-Period Computation Offloading with Reliability Guarantees in Fog Networks

Junhua Wang, Kai Liu, *Member, IEEE*, Bin Li, *Senior Member, IEEE*, Tingting Liu, *Member, IEEE*, Ruoguang Li, *Student Member, IEEE*, and Zhu Han, *Fellow, IEEE*

Abstract—Computation offloading over fog computing has the potential to improve reliability and reduce latency in future networks. This paper considers a scenario where roadside units (RSUs) are installed for offloading tasks to the computation nodes including nearby fog nodes and a cloud center. To guarantee the reliable communication, we formulate the first subproblem of power allocation, and leverage the conditional value-at-risk approach to analyze the successful transmission probability in the worse-case channel condition. To complete computation tasks with low latency, we formulate the second subproblem of task allocation into a multi-period generalized assignment problem (MPGAP), which aims at minimizing the total delay by offloading tasks to the 'right' fog nodes at 'right' period. Then, we propose a modified branch-and-bound algorithm to derive the optimal solution and a heuristic greedy algorithm to obtain approximate performance. In addition, the master problem is proposed as a non-convex optimization problem, which considers both the reliability-guaranteed and delay-sensitive requirements. We design the Lagreedy algorithm by combining the subgradient algorithm and the heuristic algorithm. Comprehensive evaluations demonstrate that the Lagreedy is able to obtain the shortest delay with a high power consumption, while the branch-and-bound algorithm can achieve both shorter delay and lower power consumption with reliability guarantees.

Index Terms—Computation offloading, delay-sensitive, reliability-guaranteed, fog computing

1 Introduction

The upcoming fifth generation (5G) wireless system is promising to support a wide range of applications. The strict requirement of low latency and high reliability are becoming the most appealing features for emerging applications, e.g., virtual reality, intelligent transportation, real-time control of cyber physical systems and smart factory, etc [1].

However, the latency and reliability in the cellular network varies with multiple factors, e.g., transmitter-receiver distance, wireless communication technology, network architecture, and the number of network users, etc. [2]. New PHY techniques have been adopted to reduce end-to-end latency and improve reliability, such as adopting multiple-input multiple-output (MIMO) antenna schemes, shorter transmission time interval (TTI), and prioritized scheduling policy to minimize queuing delay, etc [3]. Some other techniques based on the diverse interfaces and multiple channel access can also improve reliability and reduce latency [4]. In addition, mobile edge computing (MEC), fog computing and device-to-device (D2D) communication [5] are considered as promising technologies to process computationally intensive tasks [6]. Among which, fog computing is capable of providing more usage flexibility by coordinating the use of geographically distributed network edge devices. Through offloading tasks to fog network, the long-distance propagation delay as well as users' queuing delay at core networks could be avoided. Thereby, it is promising to achieve higher reliability and lower latency.

Fog computing was first proposed by Cisco in 2014 to address the challenges of phenomenal data growth in IoT applications [7]. Compared with conventional cloud computing, fog nodes have geo-distribution awareness, fast responses, and high availability to end users. Numerous work has considered the tradeoff of workload assignment between fog and cloud. However, to the best of our knowledge, the majority of work on computation offloading aimed at achieving energy efficient offloading strategies [8], and to optimize the energy consumption and delay performance [9], etc. Few work considered to satisfy the reliability constraint [10], or the latency and reliability-aware computation offloading strategy [11]. However, the efficient utilizations of communication and computation resource capacities are not discussed.

- *J. Wang is with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China. E-mail: jhua1207@nuaa.edu.cn.*
- *K. Liu is with the College of Computer Science, Chongqing University, Chongqing 400030, China. E-mail: liukai0807@cqu.edu.cn.*
- *B. Li is with the School of Electrical Engineering and Information, Sichuan University, Chengdu 610065, China. E-mail: bin.li@scu.edu.cn.*
- *T. Liu is with the School of Information and Communication Engineering, Nanjing Institute of Technology, Nanjing 211167, China. E-mail: liutt@njit.edu.cn.*
- *R. Li is with the Beijing Key Laboratory of Work Safety Intelligent Monitoring, School of Electronic Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China. E-mail: ruoguangli@bupt.edu.cn.*
- *Z. Han is with the University of Houston, Houston, TX 77004, and also with the Department of Computer Science and Engineering, Kyung Hee University, Seoul 446-701, South Korea. E-mail: zhan2@uh.edu.*

Manuscript received 25 Oct. 2018; revised 3 May 2019; accepted 13 May 2019. Date of publication 27 May 2019; date of current version 4 Aug. 2020. (Corresponding author: Kai Liu.)
Digital Object Identifier no. 10.1109/TMC.2019.2918773

In this paper, we consider both the reliability-guaranteed and delay-sensitive requirements under the computation offloading scenario with heterogenous fog nodes and a remote cloud. Our work is distinguished from others in the following three aspects. First, the usage of communication and computation resources of fog nodes are modeled as two dynamic resource queues, with the constraints that the consumed resources for task transmission and computation should not exceed the idle resource amount in the queue at any time. Second, since the available communication and computation resources of fog nodes are limited and dynamic, and computation tasks require distinct resources for different fog nodes, we consider to schedule computation tasks in a time sequence to improve overall performance. Third, we aim to minimize the total delay of all tasks by satisfying the communication reliability at the worst-case channel condition. The reliability constraint is analyzed using a risk theoretical approach. The main contributions of this work are outlined as follows:

- We present a computation offloading architecture consisting of multiple fogs nodes with heterogeneous communication and computation capabilities and a cloud center. Further, we model the dynamic communication and computation resources as two dynamic queues with the purpose of recording the resource availability and improve resource utilization.
- We formulate the first subproblem of power allocation for achieving the reliability-guaranteed communication, and adopt the conditional value-at-risk approach to compute the required transmission powers for any pair of task and fog node.
- We formulate the second subproblem of task allocation into a multi-period generalized assignment problem (MPGAP), which aims at minimizing the total delay of all tasks by allocating them to the 'right' nodes at 'right' periods. MPGAP is an integer programming problem, and can be considered as a combination of multi-period knapsack problem (MPKP) and a GAP. Each of them is NP-hard.
- We propose a modified branch-and-bound algorithm to derive the optimal solution for MPGAP. The lower bound at the branch node is first computed by allocating tasks to the most beneficial computation nodes and periods without considering resource constraints. Then, it is refined by checking the feasibility of resource capacities, and solving a sequence of minimization knapsack problems (minKPs). Moreover, we design a heuristic greedy algorithm to obtain approximate performance.
- We formulate a non-convex optimization problem which considers both the reliability-guaranteed and delay-sensitive requirements. The Lagreedy algorithm is designed by combining the subgradient algorithm and heuristic greedy algorithm.
- We compare the two-step solution and the Lagreedy solution. Comprehensive studies show that the Lagreedy algorithm can obtain the lowest average delay by setting a larger weight to the delay metric, and the branch-and-bound algorithm can always obtain better performance on both the delay and power consumption.

The reminder of this paper is organized as follows. Section 2 reviews the related work. Section 3 presents the system model. In Section 4, the MPGAP is formulated. In Section 5, we propose a modified branch-and-bound algorithm and a heuristic greedy algorithm. The simulation results are given in Section 6. We conclude the paper in Section 7.

2 Related Work

In order to meet the high-reliability and low-latency requirement in future 5G applications, Nielsen et al. [12] used multiple interfaces to achieve ultra-reliable communications. To reduce latency, an optimization problem as well as the generic evaluation method are proposed to split the total amount of information to transmit across different interfaces. Sutton et al. [13] discussed the multi-channel strategies for achieving low-latency LTE unlicensed band access, including predicting traffic profiles, implementing request-to-send/clear-to-send (RTS/CTS), etc. They demonstrate that the transmission latency can be decreased by allowing a packet to be transmitted on either the unlicensed or the licensed spectrum.

On the other side, fog computing, MEC [14] and D2D communication [15] attract much research attention recently. Yu et al. [16] considered the power allocation in fog network to maximize the utility function from the energy efficiency perspective. They formulate a mix integer nonlinear programming problem, and adopt the Benders decomposition algorithm to separate the integer variables and continuous variables. Deng et al. [17] formulated a workload assignment problem between fog and cloud considering the tradeoff between power consumption and transmission delay. An approximate approach is developed to solve the primal problem through decomposing and solving subproblems with optimization techniques. Zhao et al. [18] aimed to maximize the probability that the delay bounds of computation tasks are satisfied by designing offloading policy between edge cloud and remote cloud. They find that the tasks with stringent delay bounds are offloaded to edge cloud and tasks with loose bounds are mainly offloaded to remote cloud.

Few works considered both the reliability and latency requirements. The work in [10] studied the edge computing and proactive caching problem, where cloudlets exploit both computing and storage capabilities by proactively caching popular task computation results to reduce computing delay. The task distribution problem is considered as a matching game which aims to minimize the computing delay at a required reliability. The work in [19] analyzed the tradeoff between latency and reliability when dividing a large task into sub-tasks and offloaded to multiple nearby edge nodes. The objective is to minimize the product of latency and failure probability of offloading links. However, the resource constraints of edge nodes are not considered.

3 System Model

3.1 Computation Offloading Scenario

This work considers a typical network service scenario, in which a set of roadside units (RSUs) can communicate with various users and heterogenous computation nodes [20]. As shown in Fig. 1, the computation nodes include heterogenous fog nodes, and a cloud center. Due to the limited computation capabilities and energy budgets, the users (e.g.,

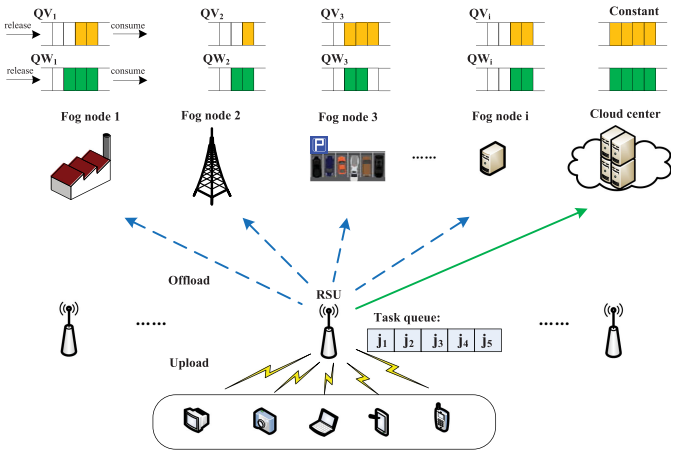


Fig. 1. System model.

mobile phone user, roadside camera, nearby laptop user, etc.) may choose to offload computation tasks to nearby RSUs. Then, the tasks will be stored in a queue and scheduled to the fog nodes based on certain scheduling algorithm. Note that in practical deployment, the edge servers with powerful computation capacities can be configured at the RSU side for making scheduling decisions. The potential fog nodes may consist of the parking lot, smart building and mobile edge computing server, which have idle computation resources during their off-peak hours. Fog nodes can connect with RSUs via various wireless network interfaces. For example, the cellular base station can communicate with RSUs via the LTE interface [21], and some smart buildings may communicate with RSUs via WiFi. Therefore, the communication environment changes notably in different cases. Besides fog nodes, RSU can offload computation tasks to the remote cloud center via the reliable high-speed wired fiber system. The cloud is considered to have unlimited resources, and the total delay to complete any computation task can be assumed as a constant [22].

3.2 Resource Queue Model

Each fog node has limited communication bandwidth. In order to enhance the utilization of available bandwidth resource of fog nodes, we use a dynamic resource queue to record the usage of communication bandwidth at each period. Once the current task is completely uploaded to fog node, the occupied channels will be released to the resource queue, and the available bandwidth will increase accordingly. Similarly, each fog node has a dynamic computation resource queue (e.g., QV_1 shows the computation resource queue of fog node 1). Before a set of tasks being offloaded to these fog nodes, the resources are gradually increasing in a 'short duration' with the completion of current transmission and computation processes. The length of a 'short duration' depends on several critical factors, such as the task arrival rate, the delay requirement and the resource availability of the fog nodes, etc. In this paper, a 'short duration' refers to several to dozens of milliseconds. If we divide a short duration into multiple periods, there is an increase of resource queues at each period. Assume that the fog node can predict the incremental resource amount at each period. At the beginning of a short scheduling duration, fog nodes inform the RSU about the predicted variety of resource queues. When the RSU starts to offload tasks, the resource queues will be gradually occupied by this set of tasks. At the beginning of next short scheduling duration, the occupied resources will be gradually released and then, occupied by newly arrived tasks.

3.3 Task Allocation Model

As described, the available sources of fog nodes are dynamic, and the tasks may consume different amount of resources when they are offloaded to different fog nodes. The scheduling policy for allocating tasks to different computation nodes at different periods will have significant impact on system performance. Note that the tasks are not instantly allocated to the nearest fog node since its communication or computation resources may be insufficient at the current period. They can choose to either wait at the task queue, or be offloaded to other fog nodes. Therefore, we consider the multi-period task allocation mechanism as follows. Given a period $m = 0$, RSU has a non-empty task queue. Within the subsequent periods from $m = 1$ to $m = t$, there is an increment of available resources at each period due to the completion of ongoing tasks. To minimize the total delay of all tasks, the RSU makes scheduling decisions by allocating tasks to different fog nodes at different periods, with the constraint that the occupied communication and computation resources must not exceed the cumulative resource amount at the fog node at each period $m \in [1, t]$.

In addition, it is possible that it may take multiple time periods to complete a computation task. The task can occupy the allocated communication and computation resources from the offloading period until it is completed. However, the fog nodes can predict when the task will be completed, which is the time when the occupied resources will be released. To execute multi-period task allocation, the fog nodes will inform in advance the RSU about their current available resources and the newly available resources during each period.

4 Problem Formulation

4.1 Notations

Denote $I = \{0, \ldots, |I|\}$ as the set of computation node indices. Let $i = 0$ represent the cloud center, and $i \in I \& i > 0$ represent the fog nodes. For each fog node $i (i > 0)$, the total communication bandwidth (i.e., *Hz*) is denoted by W_i, and the maximum computation rate (i.e., *the number of cycles per second*) is denoted by V_i. Denote $J = \{1, \ldots, |J|\}$ as the set of task indices. For task $j \in J$, let b_j represent the size of input data, and c_j represent the size of computation task. The measurement of b_j is the *bit*, and the measurement of c_j is the *cycle*. The allocated communication bandwidth for task j from fog node i is denoted by $w_{i,j}$, and the allocated computation rate is denoted by $v_{i,j}$. We consider that tasks are offloaded in different periods. The set of periods are denoted by $T = \{1, \ldots, |T|\}$. Each fog node i has two resource queues. One is communication resource/bandwidth queue QW_i, and the other is computation resource queue QV_i. At period t, the resource queues may increase due to the completion of current tasks. The increased communication bandwidth is denoted by $\varphi_{i,t}$. The increased computation rate is denoted by $\nu_{i,t}$. In addition, the maximum available bandwidth of fog node i at period t is calculated as $W_{i,t} = \sum_{m=1}^{t} \varphi_{i,m}$. $V_{i,t} = \sum_{m=1}^{t} \nu_{i,m}$ gives the cumulative achievable computation rate of fog node i at period t.

Within duration $[1,t]$, the total required communication bandwidths and computation rates of all tasks offloaded to i should not exceed the resource amount $W_{i,t}$ and $V_{i,t}$, respectively. In addition, the cumulative available communication and computation resources at any period t (i.e., $W_{i,t}$ and $V_{i,t}$) can not exceed the total amount of resources (i.e., W_i and V_i) for any fog node i. We introduce a binary variable $x_{i,j,t}$, where $x_{i,j,t}=1$ means task j is offloaded to computation node i at period t; otherwise, it is 0. $\boldsymbol{x}$ includes all decision variables.

4.2 Problem Formulation

4.2.1 Reliability-Guaranteed Communication

If task j is to be processed at fog node i, the input data will be transmitted to fog node i before being computed. Assume that tasks are transmitted in orthogonal channels, and the bandwidth allocated to task j is $w_{i,j}$. Denote $h_{i,j}$ as the channel gain between fog node i and the RSU from which task j is transmitted. The achievable transmission rate of task j is [23]:

$$r_{i,j} = w_{i,j} \cdot \log_2\left(1+\gamma_{i,j}\right), \tag{1}$$

where $\gamma_{i,j}$ is the received signal-to-noise-ratio (SNR) at fog node i for task j, and it is computed by [24]:

$$\gamma_{i,j} = \frac{\left|h_{i,j}\right|^2 \cdot K \cdot s_{i,j}^{-\alpha} \cdot p_{i,j}}{N_0}, \tag{2}$$

where K is a system constant; $s_{i,j}$ is the transmission distance of task j to fog node i and α is the path loss exponent. $p_{i,j}$ is the transmission power, and N_0 is noise power.

Due to the varying wireless communication environments between RSU and the heterogenous fog nodes, the channel uncertainty becomes a key factor to hinder the transmission reliability. To guarantee the reliable communication under the worst-case channel condition, the assigned transmission power should be sufficient to offload tasks to corresponding fog nodes with the fast changing channel gains. In this paper, we assume that the channel fading $\left|h_{i,j}\right|^2$ follows a class of distribution with the mean μ and variance Σ. The distribution set is represented as

$$\tilde{p} = \left\{\mathbb{P} : \mathbb{E}_{\mathbb{P}}\left[\left|h_{i,j}\right|^2\right] = \mu, \mathbb{E}_{\mathbb{P}}\left[\left|h_{i,j}\right|^2 - \mu\right]^2 = \Sigma\right\}. \tag{3}$$

A successful transmission is referred to the event that the received SNR is above a certain threshold. The transmission reliability is measured by the possibility that a successful transmission probability is beyond a reliability threshold. That is,

$$\Pr_{[\mathbb{P}]}\left(\gamma_{i,j} \geq \gamma_i^{tgt}\right) \geq 1-\varepsilon, \tag{4}$$

where γ_i^{tgt} is the target SNR threshold, $1-\varepsilon$ represents the reliability threshold, and ε is the maximum tolerant transmission error rate.

The reliability-guaranteed communication focuses on the transmission reliability under the worst-case distribution of channel fading. To guarantee the reliable transmission of task j to fog node i, we compute the minimum transmission power $p_{i,j}$ by formulating the problem as follows.

$$\begin{aligned}
&\min_{p_{i,j}} \sum_{i\in I}\sum_{j\in J} p_{i,j},\\
&\text{s.t. } \inf_{\mathbb{P}\in\tilde{p}} \Pr_{[\mathbb{P}]}\left(\gamma_{i,j} \geq \gamma_i^{tgt}\right) \geq 1-\varepsilon, \forall i \in I \& i \neq 0, j \in J,\\
&\mathbb{P} = \left\{\left|h_{i,j}\right|^2 : \mathbb{E}\left[\left|h_{i,j}\right|^2\right] = \mu, \mathbb{E}\left[\left|h_{i,j}\right|^2 - \mu\right]^2 = \Sigma\right\}.
\end{aligned} \tag{5}$$

The inequality constraint requires that the probability of successful transmission at the worst-case channel condition should be larger than the reliability threshold $1-\varepsilon$. The channel fading $\left|h_{i,j}\right|^2$ comes from the distribution set $\mathbb{P}$.

4.2.2 Delay-Sensitive Communication

Since tasks may not be offloaded instantly, they need to wait at the RSU before being transmitted to corresponding fog nodes. Therefore, the delay of an offloading operation includes the waiting delay, transmission delay and computation delay. In addition, the cloud center is assumed to have unlimited resources, and the total delay to complete any task is assumed as a constant τ. Therefore, when task j is offloaded to computation node i at period t, the total delay is computed by:

$$d_{i,j,t} = \begin{cases} t_j + t - 1 + b_j/\ r_{i,j} + c_j/\ v_{i,j}, & i = 1,\ldots,|I|,\\ \tau, & i = 0, \end{cases} \tag{6}$$

where t_j is the waiting time of task j before RSU making scheduling. $t-1$ is the waiting time of task j before being offloaded. Since RSU makes scheduling in each short duration, the tasks arrived during this short duration will wait for the next-time scheduling. $b_j/\ r_{i,j}$ is the transmission delay, and $c_j/\ v_{i,j}$ is the computation delay.

We assume that all tasks are offloaded with the minimum transmission power computed in (5). To minimize the total delay of all offloaded computation tasks, we formulate the multi-period generalized assignment problem (MPGAP) as follows.

$$\begin{aligned}
&\min_{x_{i,j,t}} \sum_{i\in I}\sum_{j\in J}\sum_{t\in T} d_{i,j,t}\cdot x_{i,j,t}\\
&\text{s.t. } \text{C1}: \sum_{m\in[1,t]}\sum_{j\in J} w_{i,j}\cdot x_{i,j,m} \leq \sum_{m\in[1,t]} \varphi_{i,m} = W_{i,t},\\
&\qquad \forall i \in I \& i \neq 0, \forall t \in T,\\
&\qquad \text{C2}: \sum_{m\in[1,t]}\sum_{j\in J} v_{i,j}\cdot x_{i,j,m} \leq \sum_{m\in[1,t]} \nu_{i,m} = V_{i,t},\\
&\qquad \forall i \in I \& i \neq 0, \forall t \in T,\\
&\qquad \text{C3}: \sum_{i\in I}\sum_{t\in T} x_{i,j,t} = 1, \forall j \in J,\\
&\qquad \text{C4}: x_{i,j,t} \in \{0,1\}, \forall i \in I, \forall j \in J, \forall t \in T.
\end{aligned} \tag{7}$$

The constraint C1 requires that the sum of occupied communication bandwidth of all tasks offloaded to fog node i from period 1 to t cannot exceed the cumulative available communication resources $W_{i,t}$ for all periods $t \in T$. Similarly, C2 requires that the sum of required computation rates of all tasks for fog node i from period 1 to t cannot exceed the cumulative

available computation resources $V_{i,t}$ for all periods $t \in T$. C3 states that each task j can be assigned to exactly one fog node i at only one period t. C4 is the binary variable constraint.

5 Proposed Algorithm

In Section 5.1, we first introduce the risk theoretical approach, which computes the minimum transmission power for each pair of task and fog node under the reliability constraint. The minimum transmission power is used when allocating tasks to different computation nodes. Then, to minimize the total delay, we propose the modified branch-and-bound algorithm in Section 5.2.1, and the heuristic greedy algorithm in Section 5.2.2 for task allocation. Finally, in Section 5.3, we consider both the reliability-guaranteed and delay-sensitive requirements by formulating a non-convex problem. The Lagreedy algorithm is proposed to iteratively adjust the transmission power, as well as the task allocation decisions to obtain lower delay.

5.1 Risk Theoretical Approach

For the first subproblem in (5), we aim to assign the transmission power for any pair of task j and fog node i while reducing the risk of violating the reliability constraint. As a special class of risk measure approach, the conditional value-at-risk (CVaR) is treated as an alternative for solving value-at-risk (VaR) problems in financial applications [25], [26], [27]. Thus, we adopt CVaR to transform the chance constraint in (5) into a convex approximation. For a random variable ξ, its CVaR under distribution $\mathbb{P}$ is defined as

$$\mathbb{P} - \mathrm{CVaR}_{\varepsilon}[\xi] = \inf_{\beta \in \mathbb{R}} \left\{ \beta + \frac{1}{\varepsilon} \mathbb{E}_{\mathbb{P}}[\max(\xi - \beta), 0] \right\}, \tag{8}$$

where $\mathbb{R}$ denotes the set of all real numbers. According to [28], for a continuous loss function that is either concave or quadratic in ξ, the following equivalence holds,

$$\inf_{\mathbb{P} \in \tilde{p}} \Pr_{[\mathbb{P}]}[L(\xi) \le 0] \ge 1 - \varepsilon \Leftrightarrow \sup_{\mathbb{P} \in \tilde{p}} \mathbb{P} - \mathrm{CVaR}_{\varepsilon}[L(\xi)] \le 0. \tag{9}$$

In this paper, we define the measurable loss function as:

$$L\left(\left|h_{i,j}\right|^2\right) = \gamma_i^{tgt} - \frac{\left|h_{i,j}\right|^2 \cdot K \cdot s_{i,j}^{-\alpha} \cdot p_{i,j}}{N_0}. \tag{10}$$

Since the expectation in (8) is hard to calculate because of integration, we use the semi-definite programming (SDP) [29] to represent the worst-case CVaR. The feasible set of (8) can be written as

$$\left\{ p_{i,j} \in \mathbb{R} : \begin{array}{c} \boldsymbol{M_{i,j}} \succeq 0, \beta_{i,j} + \frac{1}{\varepsilon} \mathbf{tr}\left(\boldsymbol{\Omega} \boldsymbol{M_{i,j}}\right) \le 0, \\ \boldsymbol{M_{i,j}} - \begin{bmatrix} 0 & -\frac{K \cdot s_{i,j}^{-\alpha} \cdot p_{i,j}}{2 \cdot N_0} \\ -\frac{K \cdot s_{i,j}^{-\alpha} \cdot p_{i,j}}{2 \cdot N_0} & \gamma_i^{tgt} - \beta_{i,j} \end{bmatrix} \succeq 0 \end{array} \right\}, \tag{11}$$

where $\mathbf{tr}(\cdot)$ denotes the matrix trace, and

$$\boldsymbol{\Omega} = \begin{bmatrix} \Sigma + \mu^2 & \mu \\ \mu & 1 \end{bmatrix}. \tag{12}$$

Then the problem in (5) can be reformulated as the following conic optimization problem

$$\begin{aligned} & \min_{\beta_{i,j}, \boldsymbol{M_{i,j}}, p_{i,j}} p_{i,j}, \\ \text{s.t. } & \mathrm{C5}: \beta_{i,j} + \frac{1}{\varepsilon} \mathbf{tr}\left(\boldsymbol{\Omega} \boldsymbol{M_{i,j}}\right) \le 0, \\ & \mathrm{C6}: \boldsymbol{M_{i,j}} \succeq 0, \\ & \mathrm{C7}: \boldsymbol{M_{i,j}} - \begin{bmatrix} 0 & -\frac{K \cdot s_{i,j}^{-\alpha} \cdot p_{i,j}}{2 \cdot N_0} \\ -\frac{K \cdot s_{i,j}^{-\alpha} \cdot p_{i,j}}{2 \cdot N_0} & \gamma_i^{tgt} - \beta_{i,j} \end{bmatrix} \succeq 0, \end{aligned} \tag{13}$$

where $\boldsymbol{\Omega}$ is defined in (12), $\boldsymbol{M_{i,j}}$ is a combined variable and $\boldsymbol{M_{i,j}} \in \mathbb{S}^2$. The deduction process can be referred to [28].

5.2 Two Solutions for MPGAP

For the second subproblem in (7) which addresses the task allocation, we consider two strategies. One is the branch-and-bound algorithm, and the other is the heuristic greedy algorithm.

5.2.1 Branch-and-Bound Algorithm

As proved in [30], the generalized assignment problem (GAP) is an NP-hard problem. We obtain the optimal solution of (7) by modifying a branch-and-bound algorithm, which was intended for GAP. The lower bound at each branch node is calculated by solving a set of minimization 0-1 knapsack problems (MinKPs).

First, we consider a relaxed problem (i.e., MPGAP-R) by removing the capacity constraints of C1 and C2 in (7). Each task will freely select the computation node and corresponding offloading period on which the total delay of completing the task itself is the shortest. For example, given a task j, we find computation node i and offloading period t such that

$$d_{i_j,j,t_j} = \min_{\forall i,t} \left\{ d_{i,j,t} \right\}. \tag{14}$$

By setting $x_{i_j,j,t_j} = 1$ and $x_{i,j,t} = 0$ for all $i \ne i_j$ and $t \ne t_j$, we get a loose lower bound as:

$$Z = \sum_{j \in J} d_{i_j,j,t_j}. \tag{15}$$

Then, the resource capacity constraints of C1 and C2 are checked for feasibility of the relaxed problem. Denote $J_{i,t}$ as the set of tasks which are assigned to computation node i at period t. We have

$$J_{i,t} = \left\{ \forall j \middle| x_{i,j,t} = 1 \right\}. \tag{16}$$

For any fog node i, the resource capacity constraints may be violated at some periods. Denote T_i as the set of periods that the allocated tasks to fog node i exceed the resource capacities at corresponding periods.

$$T_i = \left\{ \forall t \middle| \sum_{j \in J_{i,t}} w_{i,j} \cdot x_{i,j,t} > W_{i,t} \middle| \sum_{j \in J_{i,t}} v_{i,j} \cdot x_{i,j,t} > V_{i,t} \right\} \tag{17}$$

Let $I' = \left\{ T_1, \ldots, T_{|I|} \right\}$, the lower bound Z will be increased by the sum of objective values obtained by solving for each T_i the problem

$$\begin{aligned} \min_{y_{i,j,t}} \ & z_{i,t} = \sum_{j \in J_{i,t}} \phi_j \cdot y_{i,j,t}, \\ \text{s.t.} \ & \sum_{j \in J_{i,t}} w_{i,j} \cdot y_{i,j,t} \le W^0_{i,t}, \\ & \sum_{j \in J_{i,t}} v_{i,j} \cdot y_{i,j,t} \le V^0_{i,t}, \\ & y_{i,j,t} \in \{0,1\}, \end{aligned} \tag{18}$$

where

$$\phi_j = \min_{i' \in I - \{i_j\}, t' \in T - \{t_j\},} \Big\{ d_{i',j,t'} - d_{i_j,j,t_j} \Big\}, \tag{19}$$

and

$$\begin{aligned} W^0_{i,t} &= \sum_{j \in J_{i,t}} w_{i,j} \cdot x_{i,j,t} - W_{i,t}, \\ V^0_{i,t} &= \sum_{j \in J_{i,t}} v_{i,j} \cdot x_{i,j,t} - V_{i,t}. \end{aligned} \tag{20}$$

$W^0_{i,t}$ and $V^0_{i,t}$ represent the excess part of communication and computation resource capacitites. ϕ_j represents the minimum penalty caused by the reassignment of task j. The problem in (18) is an MinKP and can be transformed into a traditional multi-dimensional KP. The optimal solution can be obtained by the dynamic programming or the branch-and-bound algorithm. Denote the optimal solution of (18) as $y^*_{i,j,t}$, which indicates the reassignment operations that lead to a minimum increase (i.e., $z^*_{i,t}$) on the initial lower bound Z. Therefore, the lower bound is revised as:

$$LB = Z + \sum_{T_i \in I'} \sum_{t \in T_i} z^*_{i,t}. \tag{21}$$

The initial solution is also revised according to the values of $y^*_{i,j,t}$. If $y^*_{i,j,t}$ equals zero, the initial assignment $x_{i,j,t}$ keeps one. If $y^*_{i,j,t}$ equals one, it means task j should be removed from fog node i at period t. Thus we set $x_{i,j,t}$ to zero, and the corresponding variable $x_{i',j,t'}$ whose total delay satisfies $\phi_j = d_{i',j,t'} - d_{i_j,j,t_j}$ should be set to one. As proved in [30], the relaxation method to compute lower bound can be viewed as a special case of the Lagrange relaxation, by setting $d_{i',j,t'}$ as the values of relaxation multipliers. After revising the lower bound, the branch process is conducted. We compute the match degree of task j when it is kept with fog node i at period t:

$$u_{i,j,t} = \phi_j \Big/ \Big(w_{i,j} \Big/ W^1_{i,t} + v_{i,j} \Big/ V^1_{i,t} \Big), \tag{22}$$

where $W^1_{i,t} = W_{i,t} - \sum_{j \in F_{i,t}} w_{i,j} \cdot x_{i,j,t}$ and $V^1_{i,t} = V_{i,t} - \sum_{j \in F_{i,t}} v_{i,j} \cdot x_{i,j,t}$. $F_{i,t}$ represents the set of tasks allocated to fog node i at period t. Note that $F_{i,t}$ is the allocation results after revising. A larger value of $u_{i,j,t}$ indicates that task j is better kept with fog node i at period t with the consideration of both the penalty for reallocating task j and the available resources of fog node. We choose the branch variable x_{i^*,j^*,t^*} such that

$$u_{i^*,j^*,t^*} = \max \Big\{ u_{i,j,t} \Big| \forall y^*_{i,j,t} = 0 \Big\}. \tag{23}$$

Then, $x_{i^*,j^*,t^*} = 1$ is examined, and the above relaxation method and revising strategy will be proceeded with x_{i^*,j^*,t^*} being fixed to one. In the procedure of branch-and-bound, some feasible solutions may be obtained after the revising operation. Specifically, if $W^1_{i,t}$ and $V^1_{i,t}$ are positive for all i and t, which means the current solution satisfies the

TABLE 1
The Branch-and-Bound Algorithm

Algorithm 1 The branch-and-bound algorithm.

Input: Task set J, computation node set I, period set T, allocated bandwidth $w_{i,j}$ and computation capability $v_{i,j}$ from j to i; resource increment $\varphi_{i,t}$ and $\nu_{i,t}$
Output: Allocation matrix $[x]$ and objective value z^*
Steps:

Step 1: Initialization
Set $F = J$, $e = 0$, $z^* = +inf$, $[x] = 0$.
Set current allocation matrix $[cx] = 0$, and the allocation matrix that have been determined $[sx] = 0$.
Compute $W_{i,t}$ and $V_{i,t}$ for all i and t; and compute possible total delay $d_{i,j,t}$ for all i, j and t.
Step 2: Free selection
Let $[cx] = sx$, and compute the lower bound at current branch as $LB = \sum_{\forall i,j,t} d_{i,j,t} . * sx_{i,j,t}$.
for $j \in F$: **do**
  Find $d_{i_j,j,t_j} = \min_{\forall i,t} \{d_{i,j,t}\}$ where $r_{i,j}$ and $v_{i,j}$ satisfy resource capacities $W_{i,t}$ and $V_{i,t}$.
  Set $cx_{i_j,j,t_j} = 1$, $LB = LB + d_{i_j,j,t_j}$.
end for
Step 3: Revise
for $i \in I$ and $t \in T$: **do**
  Compute $W^0_{i,t}$ and $V^0_{i,t}$ using (20).
  if $W^0_{i,t} > 0$ and $V^0_{i,t} > 0$: **then**
    Find ϕ_j and $d_{i',j,t'}$ from all j which satisfy (19).
  end if
  Solve (18) and obtain the optimal solution $y^*_{i,j,t}$ and value $z^*_{i,t}$. If no solution is found, go to Step 5.
  Update LB using (21).
  Find $y^*_{i,j,t} = 1$ for all j, and set $cx_{i,j,t} = 0$, set $x_{i',j,t'} = 1$ which has a delay difference of ϕ_j with $d_{i,j,t}$.
end for
Step 4: Branch
Compute $W^1_{i,t}$, $V^1_{i,t}$ for all i, j and t such that $cx_{i,j,t} = 1$.
if $W^1_{i,t} > 0$ and $V^1_{i,t} > 0$ for all i and t: **then**
  A feasible solution is found, check to update $[x]$ and z^*.
else
  Find u_{i^*,j^*,t^*} using (23) such that $sx_{i^*,j^*,t^*} \neq 1$.
  Set $W_{i^*,t} = W_{i^*,t} - w_{i^*,j^*}$, $V_{i^*,t} = V_{i^*,t} - v_{i^*,j^*}$ from t^* to T. Set $F = F - \{j^*\}$, $sx_{i^*,j^*,t^*} = 1$, $e = e + 1$.
end if
Step 5: Backtrack
if $sx_{i^*,j^*,t^*} = 0$: **then**
  Continue backtracking until finding a branch variable $sx_{i,j,t}$ which equals one.
  Set $W_{i,t} = W_{i,t} + w_{i,j}$, $V_{i,t} = V_{i,t} + v_{i,j}$ from t to T. Set $F = F + \{j\}$, $sx_{i,j,t} = 0$.
end if
if $sx_{i^*,j^*,t^*} = 1$: **then**
  Set $W_{i^*,t} = W_{i^*,t} + w_{i^*,j^*}$, $V_{i^*,t} = V_{i^*,t} + v_{i^*,j^*}$ from t^* to T. Set $F = F + \{j^*\}$, $sx_{i^*,j^*,t^*} = 0$.
end if
Go to Step 2.

communication and computation resource constraints of any fog nodes at any periods, therefore, a feasible solution is found. As stated in [30], the algorithm will produce multiple feasible solutions before finding the optimal solution.

Table 1 shows the branch-and-bound algorithm. In Step 2, all tasks independently select their best assignments of computation nodes and offloading periods. In Step 3, the computation nodes check the feasibility of task allocation at each period. The lower bound is revised accordingly. Step 4 checks the feasibility of produced solution in Step 3. If it is infeasible, the branch variable is selected which has the highest match degree being kept with current computation node and period. In Step 5, the backtracking procedure is conducted to prune the unfeasible solutions. After branching, the algorithm goes to Step 2 for the next allocation and bound procedure.

5.2.2 Heuristic Greedy Algorithm

A basic greedy idea to solve the knapsack problem is to select the item with the highest profit weight ratio. The first step of the proposed greedy algorithm is inspired from this idea. Denote $g_{i,j,t}$ as the weight of task j being allocated to computation node i at period t, which is computed by:

$$g_{i,j,t} = u^0_{i,t} \cdot w_{i,j} + u^1_{i,t} \cdot v_{i,j}, \tag{24}$$

where $u^0_{i,t}$ and $u^1_{i,t}$ are the coefficients of communication and computation resources. We simply set them as the used resource amount of computation node i at period t. They equal one if the resources are not yet be occupied. The greedy algorithm first computes all values of delay-weight-ratio. Then, it selects the allocation operation x_{i^*,j^*,t^*} with the smallest value δ_{i^*,j^*,t^*} (As computed in (25)). If the resource capacity constraints can not be met (i.e., $w_{i^*,j^*} > W_{i^*,t^*}$ or $v_{i^*,j^*} > V_{i^*,t^*}$), x_{i^*,j^*,t^*} is set to zero; Otherwise, it is set to one. If $x_{i^*,j^*,t^*} = 0$, the assignment x_{i^*,j^*,t^*} will be excluded. If $x_{i^*,j^*,t^*} = 1$, which means the assignment is feasible, all other assignments related to task j will not be further considered. The cumulative communication and computation resources will be updated each time when a feasible assignment is found, and the unallocated tasks will recompute the delay-weight-ratio accordingly. Since the cloud center is assumed to have unlimited resources, there must be a feasible assignment for any task.

$$\delta_{i^*,j^*,t^*} = \min_{\forall i,j,t} \{ d_{i,j,t} / \ g_{i,j,t} \}. \tag{25}$$

The heuristic greedy algorithm is shown in Table 2. From lines 4 to 15, the cumulative communication and computation resources (i.e., $W_{i,t}$ and $V_{i,t}$) are updated. Note that in the greedy procedure, if the resource capacities at certain periods (e.g., $V_{i,t}$) are partially occupied, and the resource capacities at early periods (i.e., $t^* < t$) are larger than $V_{i,t}$. Then, V_{i,t^*} would be considered as the maximum available resource and it is equal to the resource amount at period t (i.e., $V_{i,t^*} = V_{i,t}$). For the first task, the greedy algorithm checks $I \cdot J \cdot T$ delay-weight-ratio values. For the second task, $I \cdot (J-1) \cdot T$ delay-weight-ratio values are checked. For task j, $I \cdot (J-j+1) \cdot T$ delay-weight-ratio values are checked. Finally, the loop will be executed up to $I \cdot T \cdot J \cdot (J+1)/\ 2$ times. The time complexity will be $\mathcal{O}(J^2)$.

TABLE 2
The Heuristic Greedy Algorithm

Algorithm 2 The heuristic greedy algorithm.

Input: Task set J, computation node set I, period set T, allocated bandwidth $w_{i,j}$ and computation capability $v_{i,j}$ from j to i; resource increment $\varphi_{i,t}$ and $\nu_{i,t}$
Output: Allocation matrix $[x]$ and objective value z^*
Steps:

1: Compute $W_{i,t}$ and $V_{i,t}$ for all i and t; compute possible total delay $d_{i,j,t}$ for all i, j and t. Set $z^* = 0$, $F = J$.
2: **while** 1: **do**
3: Set $[\delta] = 0$.
4: **for** $t \in T$ and $i \in I$: **do**
5: Set $W'_{i,t} = W_{i,t}$, $V'_{i,t} = V_{i,t}$.
6: **for** $t' \in [1, t]$ such that $x_{i,j,t'} = 1$: **do**
7: Set $W'_{i,t} = W'_{i,t} - w_{i,j}$, $V'_{i,t} = V'_{i,t} - v_{i,j}$.
8: **end for**
9: **end for**
10: **for** $i \in I$ and t from T to 1: **do**
11: **for** t^* from t to 1: **do**
12: If $W'_{i,t} < W'_{i,t^*}$, then set $W'_{i,t} = W'_{i,t^*}$.
13: If $V'_{i,t} < V'_{i,t^*}$, then set $V'_{i,t} = V'_{i,t^*}$.
14: **end for**
15: **end for**
16: **for** $t \in T$ and $i \in I$: **do**
17: Compute multipliers $u^0_{i,t}$ and $u^1_{i,t}$.
18: Compute $g_{i,j,t}$ using (24) for all $j \in F$ such that $x_{i,j,t} \neq 0$.
19: **end for**
20: Find x_{i^*,j^*,t^*} and δ_{i^*,j^*,t^*} which satisfy (25).
21: **if** $w_{i^*,j^*} < W'_{i^*,t^*}$ and $v_{i^*,j^*} < V'_{i^*,t^*}$: **then**
22: Set $x_{i^*,j^*,t^*} = 1$, $z^* = z^* + d_{i^*,j^*,t^*}$, $F = F - \{j^*\}$.
23: **else**
24: Set $x_{i^*,j^*,t^*} = 0$.
25: **end if**
26: If F is empty, stop.
27: **end while**

5.3 Non-Convex Optimization and Lagreedy Algorithm

Recall that in Section 5.1, the first subproblem computed the minimum transmission powers which guarantee the reliable communication. In Section 5.2, the second subproblem (i.e., task offloading problem) minimizes the total delay by allocating tasks to different fog nodes with the minimum transmission powers. Apparently, if we use larger transmission powers, then, the allocation decisions and the corresponding total delay of the second subproblem will change accordingly. It is interesting to explore the influence of different transmission powers on the reliability probability, and the total delay. In the following, we formulate the master problem by combining the constraints of the two subproblems and their objective functions by giving a weight coefficient to each objective.

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{p},\boldsymbol{\beta},\boldsymbol{M}} \ & \sum_{i\in I}\sum_{j\in J}\sum_{t\in T} \left(\theta_1 \cdot d_{i,j,t} + \theta_2 \cdot p_{i,j}\right) \cdot x_{i,j,t} \\ \text{s.t.} \ & \ \text{C1} \sim \text{C7}, \end{aligned} \tag{26}$$

$$\begin{aligned} L(\boldsymbol{x},\boldsymbol{p},\beta,\boldsymbol{M},\boldsymbol{\lambda}) = & \sum_{i\in I}\sum_{j\in J}\sum_{t\in T}\left(\theta_1\cdot d_{i,j,t}+\theta_2\cdot p_{i,j}\right)\cdot x_{i,j,t} + \sum_{j\in J}\left(\lambda_j^5\cdot\left(\sum_{i\in I}\sum_{t\in T}x_{i,j,t}-1\right)\right) \\ & + \sum_{i\in I}\sum_{j\in J}\left(\lambda_{i,j}^0\cdot\left(\beta_{i,j}+\frac{1}{\varepsilon}\mathbf{tr}\big(\boldsymbol{\Omega}\boldsymbol{M_{i,j}}\big)\right)-\mathbf{tr}\Big(\boldsymbol{M_{i,j}}\lambda_{i,j}^1\Big)-\mathbf{tr}\Big(\big(\boldsymbol{M_{i,j}}-\boldsymbol{U}\big)\lambda_{i,j}^2\Big)\right) \\ & + \sum_{i\in I}\sum_{t\in T}\left(\lambda_{i,t}^3\cdot\left(\sum_{m\in[1,t]}\sum_{j\in J}w_{i,j}\cdot x_{i,j,m}-W_{i,t}\right)+\lambda_{i,t}^4\cdot\left(\sum_{m\in[1,t]}\sum_{j\in J}v_{i,j}\cdot x_{i,j,m}-V_{i,t}\right)\right). \end{aligned} \tag{27}$$

where θ_1 and θ_2 are the weights of the delay metric and the power consumption metric, and $\theta_1+\theta_2=1$. By changing the weight coefficients in the objective function, we can trace the relationship between the power consumption, the reliability and the total delay. For example, by setting an extremely large weight to the delay metric, the main objective of the master problem will be to minimize the total delay. Then, the proposed algorithm may prone to use larger transmission power to allocate task. On the contrary, by setting an extremely large weight to the power consumption metric, it will prone to use the minimum transmission power to allocate task, then, the master problem will approximate to a combination of the two subproblems.

Since that the objective function contains binary decision variables and the product of two variables, and that the fraction expression is neither convex or concave, (26) is thus a non-convex problem. We consider to update the transmission powers using the subgradient descent algorithm. The lagrange relaxation function of the master problem is given in (27), where $\boldsymbol{\lambda}=\{\lambda^0,\lambda^1,\lambda^2,\lambda^3,\lambda^4,\lambda^5\}$. $\lambda^0\in\mathbb{R}^{I\times J}$, λ^1, $\lambda^2\in\tilde{\mathbb{S}}^{I\times J}$, $\lambda^1\succeq 0$, $\lambda^2\succeq 0$, $\tilde{\mathbb{S}}=\mathbb{S}^2$, $\lambda^3,\lambda^4\in\mathbb{R}^{I\times T}$, $\lambda^5\in\mathbb{R}^J$ and

$$\boldsymbol{U}=\begin{bmatrix} 0 & -\frac{1}{2}\cdot\frac{K\cdot s_{i,j}^{-\alpha}\cdot p_{i,j}}{N_0} \\ -\frac{1}{2}\cdot\frac{K\cdot s_{i,j}^{-\alpha}\cdot p_{i,j}}{N_0} & \gamma_i^{tgt}-\beta_{i,j}\end{bmatrix}, \tag{28}$$

for all $i\in I$, $j\in J$, $t\in T$.

The lagrange dual function is given as [31]:

$$g(\boldsymbol{\lambda})=\inf_{\boldsymbol{x},\boldsymbol{p},\beta,\boldsymbol{M}} L(\boldsymbol{x},\boldsymbol{p},\beta,\boldsymbol{M},\boldsymbol{\lambda}). \tag{29}$$

We consider the lagrange dual problem by maximizing $g(\boldsymbol{\lambda})$ with the domain constraints of dual variables:

$$\begin{aligned} &\max_{\boldsymbol{\lambda}} g(\boldsymbol{\lambda})=\max_{\boldsymbol{\lambda}}\inf_{\boldsymbol{x},\boldsymbol{p},\beta,\boldsymbol{M}} L(\boldsymbol{x},\boldsymbol{p},\beta,\boldsymbol{M},\boldsymbol{\lambda}) \\ &\text{s.t. } \boldsymbol{\lambda}^0\in\mathbb{R}^{I\times J},\boldsymbol{\lambda}^1,\boldsymbol{\lambda}^2\in\mathbb{S}^{I\times J},\mathbb{S}=\mathbb{S}^2, \\ &\quad \boldsymbol{\lambda}^1\succeq 0,\boldsymbol{\lambda}^2\succeq 0,\boldsymbol{\lambda}^3,\boldsymbol{\lambda}^4\in\mathbb{R}^{I\times T},\boldsymbol{\lambda}^5\in\mathbb{R}^J. \end{aligned} \tag{30}$$

Then, we propose the Lagreedy algorithm to solve the dual problem in (30). The algorithm iteratively adopts the subgradient projection and the heuristic greedy algorithm to obtain a feasible solution. In the subgradient projection, the values of lagrange multipliers are updated at certain step sizes, and the transmission powers are also updated according to KKT conditions. Once the transmission powers are updated, the heuristic greedy algorithm solves the integer programming problem (i.e., task allocation). Then, the obtained solutions are used to update the values of multipliers in the next iteration.

5.3.1 Inner Minimization Solution

By differentiating $L(\boldsymbol{x},\boldsymbol{p},\beta,M,\boldsymbol{\lambda})$ with respect to $\boldsymbol{p}$, $\boldsymbol{x}$, β and M, and let them equal to zero, we have (39), (40), (31) and (32).

$$\lambda_{i,j}^0+\lambda_{i,j}^2(2,2)=0 \tag{31}$$

$$\lambda_{i,j}^0\cdot\frac{1}{\varepsilon}\cdot\Omega-\left(\lambda_{i,j}^1\right)^T-\left(\lambda_{i,j}^2\right)^T=0. \tag{32}$$

The transmission powers $\boldsymbol{p}$ can be computed from (39). Note that the updated transmission power $\boldsymbol{p}$ in each iteration must be not lower than the the minimum transmission power computed in Section 5.1. Therefore, the reliability constraint in the master problem is definitely guaranteed. Then, with the updated transmission power, the task offloading solution (i.e., $\boldsymbol{x}$) can be obtained by solving the integer programming problem. Consider the high computation complexity of the branch-and-bound algorithm, we use the heuristic greedy algorithm in Section 5.2.2 to compute $\boldsymbol{x}$. The constraints of communication and computation resources are also satisfied in the task allocation. That is, the reliability constraint and the resource constraint are respectively satisfied in updating the transmission power, and the task allocation solution. Therefore, the Lagreedy algorithm obtains a feasible solution of the master problem.

5.3.2 Outer Maximization Solution

The subgradient of $g(\boldsymbol{\lambda})$ are given as

$$\lambda_{i,j}^0(t+1)=\left[\lambda_{i,j}^0(t)-l_{i,j}^0\cdot\left(\beta_{i,j}+\frac{1}{\varepsilon}\mathbf{tr}\big(\boldsymbol{\Omega}\boldsymbol{M_{i,j}}\big)\right)\right]^+, \tag{33}$$

$$\lambda_{i,j}^1(t+1)=\left[\lambda_{i,j}^1(t)-l_{i,j}^1.*\left(-\boldsymbol{M_{i,j}}\right)^T\right]^+, \tag{34}$$

$$\lambda_{i,j}^2(t+1)=\left[\lambda_{i,j}^2(t)-l_{i,j}^2.*\left(\boldsymbol{U}-\boldsymbol{M_{i,j}}\right)^T\right]^+, \tag{35}$$

$$\lambda_{i,t}^3(t+1)=\left[\lambda_{i,t}^3(t)-l_{i,t}^3\cdot\left(\sum_{m\in[1,t]}\sum_{j\in J}w_{i,j}\cdot x_{i,j,m}-W_{i,t}\right)\right]^+, \tag{36}$$

$$\lambda_{i,t}^4(t+1)=\left[\lambda_{i,t}^4(t)-l_{i,t}^4\cdot\left(\sum_{m\in[1,t]}\sum_{j\in J}v_{i,j}\cdot x_{i,j,m}-V_{i,t}\right)\right]^+, \tag{37}$$

$$\lambda_j^5(t+1)=\lambda_j^5(t)-l_j^5\cdot\left(\sum_{i\in I}\sum_{t\in T}x_{i,j,t}-1\right), \tag{38}$$

where $x_{i,j,m}$, $\boldsymbol{M_{i,j}}$, $p_{i,j}$ and β (which are implicitly shown in $\boldsymbol{U}$) are the solutions of inner minimization problem in

$$p_{i,j} = N_0 \cdot \left(\frac{\sum_{t \in T} x_{i,j,t} \cdot b_j \cdot w_{i,j}}{\ln 2 \cdot r_{i,j}^2 \cdot \left(\lambda_{i,j}^2(1,2) \cdot K \cdot s_{i,j}^{-\alpha} + \sum_{t \in T} x_{i,j,t} \cdot N_0 \cdot \theta \right)} - \frac{1}{\left| h_{i,j} \right|^2 \cdot K \cdot s_{i,j}^{-\alpha}} \right) \tag{39}$$

current iteration. For real variable π, $[\pi]^+ = \max(\pi, 0)$; for a real matrix $\boldsymbol{\pi}$, $[\boldsymbol{\pi}]^+ = \max(\boldsymbol{\pi}, \mathbf{0})$.

$$\left(\left(t - 1 + \frac{c_j}{v_{i,j}} \right) + \frac{b_j}{\left(w_{i,j} \log_2 \left(1 + \left| h_{i,j} \right|^2 \cdot K \cdot s_{i,j}^{-\alpha} \cdot p_{i,j} \Big/ N_0 \right) \right)} + \theta \cdot p_{i,j} \right) + (T - t + 1) \cdot \left(\lambda_{i,t}^3 \cdot w_{i,j} + \lambda_{i,t}^4 \cdot v_{i,j} \right) + \lambda_j^5 = 0 \tag{40}$$

The algorithm starts with an initialization of power allocation $\boldsymbol{p}$, auxiliary variables M and β, as well as the lagrange multipliers. Then, the decision variable $\boldsymbol{x}$ is computed using the heuristic greedy algorithm. In the next iteration, the values of lagrange multipliers are adjusted with the subgradient process. $\boldsymbol{p}$, M and β are recomputed, based on which, $\boldsymbol{x}$ is updated using the heuristic greedy algorithm. The iteration process continues until converging to a preset threshold.

6 Simulations

The parameter settings are summarized in Table 3 [18], [24], [32], [33]. To compare the efficiency of the Lagreedy algorithm, we compute a lower bound of the master problem using the subgradient descent algorithm. We first relax the integer constraint C4 in the master problem to the continuous variables. Then we derive the Lagrangian dual problem and solve it with the subgradient descent algorithm. The algorithm stops iterating until the terminal condition is met (i.e., the variation of the multiplier are within a certain range) [23]. After verifying the effectiveness of the Lagreedy algorithm, we compare it with the branch-and-bound algorithm and the heuristic greedy algorithm. For the first two algorithms, we compute the minimum transmission power for each pair of task and computation node under the reliability constraint. Then, the branch-and-bound algorithm and the heuristic greedy algorithm solve the task allocation problem with the assumption that all tasks are offloaded to computation nodes with the minimum transmission power. For the Lagreedy algorithm, if we set a relatively larger value to θ_1 in (26), the delay metric becomes the main concern in the optimization problem. It means that the Lagreedy algorithm may tend to use larger transmission power to get a higher transmission rate, which can shorten the total delay. If we set a relatively larger value to θ_2, the power consumption is seriously considered, which can lead the Lagreedy algorithm to use lower transmission power for obtaining a smaller objective value. In the following experiments, when we set θ_1 to 0.999, the Lagreedy algorithm can be called as the LaDelay, for differentiating with the LaPower algorithm which concerns more about the power consumption and sets θ_2 to 0.999. In addition, since the branch-and-bound algorithm has a higher complexity and can not obtain the optimal solution within relatively shorter time, we give the current optimal solution when the task number reaches up to 25 and 30. As we can verify later, even the currently optimal solution is better than the heuristic greedy algorithm.

Fig. 2 compares the average delay of the LaDelay algorithm (i.e., the Lagreedy algorithm when setting θ_1 to 0.999 in the objective function) and the subgradient descent algorithm. The average delay is defined as the average value of all tasks' delay. As described in Section 5.3, the Lagreedy algorithm computes a feasible solution of the master problem; while the subgradient algorithm obtains a lower bound. As shown, the average delay of the LaDelay algorithm (i.e., green dash dotted line) is slightly longer than the lower bound (i.e., orange dotted line). The difference of average delay is about 0.2 ms when the number of tasks is larger than 10. Therefore, we conclude that the LaDelay algorithm is able to obtain good, feasible solution of the master problem.

Fig. 3 compares the average delay of all algorithms. The red dashed line represents the average delay of the proposed branch-and-bound algorithm, and the black solid line represents the results of the heuristic greedy algorithm. When tasks are offloaded to the computation nodes with the minimum transmission power, the branch-and-bound algorithm always gives shorter average delay with an increasing number of tasks. The blue dotted line and green dash dotted line represent the average delay of the LaPower and the LaDelay algorithms, respectively. When setting θ_1 to 0.999, the objective value is dominated by the delay metric, the LaDelay will iteratively adjust the transmission power to achieve a shorter delay. As shown with green dash dotted line, the LaDelay converges into a better solution than those of the branch-and-bound and the heuristic greedy algorithm. However, when the power consumption becomes the main concern, the LaPower cannot obtain a shorter delay than the other two algorithms. We can explain the results from the following two aspects. First, the LaPower uses the heuristic greedy algorithm to allocate tasks in each iteration, which cannot outperform the optimal solution of the branch-and-bound algorithm. Second, it targets to minimize the power consumption, and hence the influence of delay metric is almost ignored in the converging process. In addition, with an increasing number of tasks, the average delay of all algorithms will increase accordingly. This is because when more tasks arrive at the task queue, they have to wait for longer time due to limited communication and computation resources of fog nodes, which will cause longer total delay.

Fig. 4 compares the average consumed power of all algorithms under different numbers of tasks. As shown, although both the branch-and-bound algorithm and the heuristic greedy algorithm use the minimum transmission power to offload tasks, the branch-and-bound algorithm has lower power consumption. Combined with Fig. 3, the branch-and-bound can achieve both the lower delay and lower power consumption than those of the heuristic greedy algorithm. In addition, the LaPower obtains much similar results with the heuristic greedy algorithm. But as

TABLE 3
Simulation Parameters

Parameter	Value
Noise power	10^{-12} mW
Path loss exponent	3
System parameter	1
Channel fading	2-mean 0.4-variance distribution
Distance between RSU and fog nodes	$[100, 1000]$ meters
SNR threshold	$[1, 10]$
Reliability threshold	0.9
Number of tasks	20
Number of fog nodes	6
Number of periods	6
Length of each period	0.1 ms
Required bandwidth of each task	$[1, 5]$ MHz
Bandwidth increment at each period	$[1, 3]$ MHz
Demanding computation rates of each task	$[1, 15] \times 10^8$ cycles/s
Computation rate increment at each period	$[1, 6] \times 10^8$ cycles/s
Size of input data	$(400, 100^2)$ bytes
Size of computation task	$(100, 20^2)$ Kcycles

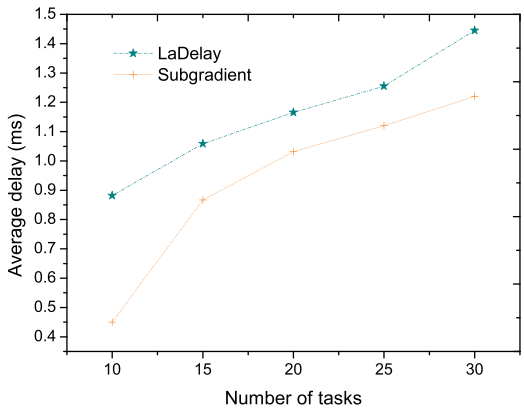


Fig. 2. Comparison of feasible solution and lower bound.

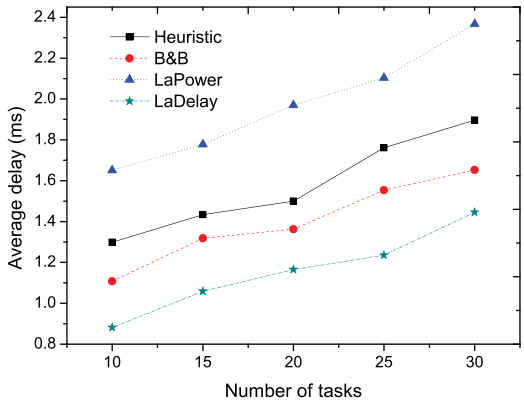


Fig. 3. Average delay under different number of tasks.

described in Fig. 3, the LaPower is reducing the transmission power at the cost of longer delay. On the contrary, the LaDelay achieves the shortest average delay with larger transmission power.

Fig. 5 evaluates the program execution time of all algorithms under different numbers of tasks. When the number of tasks increases, the execution time of the heuristic greedy algorithm increases slowly. As described in Section 5.2.2, the time complexity of the heuristic greedy algorithm is $\mathcal{O}(J^2)$.

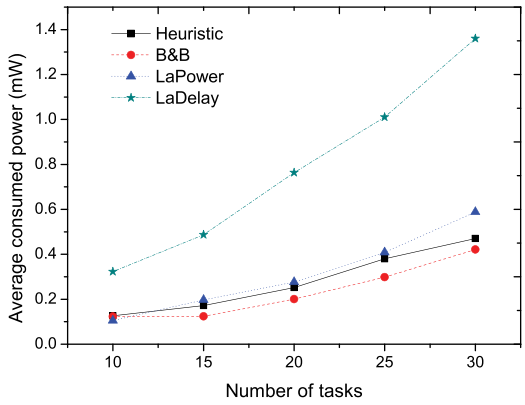


Fig. 4. Average consumed power under different number of tasks.

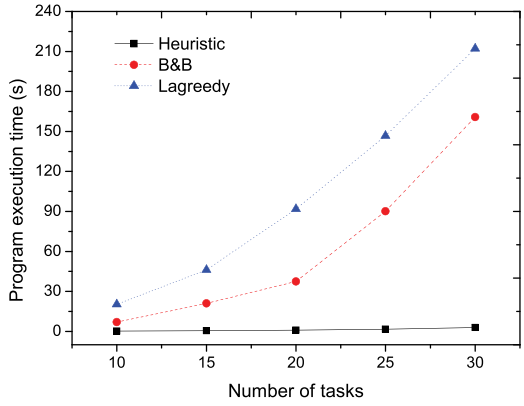


Fig. 5. Average execution time under different number of tasks.

However, the increasing trend of execution time cannot be shown clearly due to large values of y-axis. For the Lagreedy algorithm, since the greedy strategy is applied in each iteration to reallocate tasks, the time complexity is determined by the number of iterations and the execution time of heuristic algorithm. For the branch-and-bound algorithm, it outputs the optimal solution when the number of tasks is lower than 25, and outputs current optimal solution for more than 25 tasks. Since the worst-case time complexity of the branch-and-bound algorithm is exponential, we set the maximum allowable execution time of 90 s for 25 tasks, and 150 s for 30 tasks. From Fig. 3, we can observe that even the current optimal results (i.e., the average delay with 25 and 30 tasks) of the branch-and-bound algorithm keep better than the heuristic greedy algorithm. In addition, although the average execution time of the Lagreedy is higher than the preset time of the branch-and-bound algorithm, the LaDelay (i.e., the Lagreedy algorithm when setting delay weight θ_1 to 0.999) obtains the lowest average delay compared with other algorithms (see Fig. 3). However, as explained above, the average delay of the LaPower (i.e., the Lagreedy algorithm when setting power weight θ_2 to 0.999) is longer than the branch-and-bound algorithm.

Fig. 6 shows the percentage of tasks that are offloaded to cloud center. When the task number is less than 15, all tasks can be offloaded to fog nodes. With an increasing of task number, the percentage of tasks offloaded to cloud center will increase. Compared with the heuristic greedy algorithm, the branch-and-bound algorithm and the LaDelay algorithm offload fewer tasks to the cloud center in all settings. This

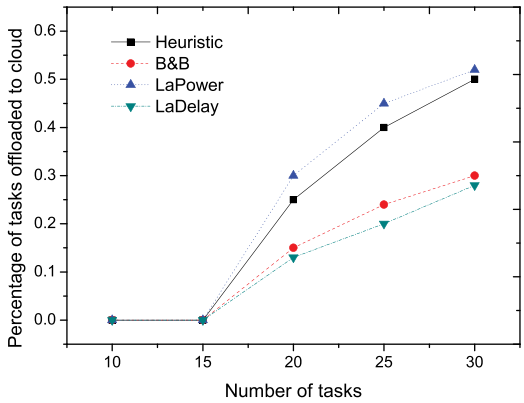


Fig. 6. Percentage of tasks offloaded to cloud.

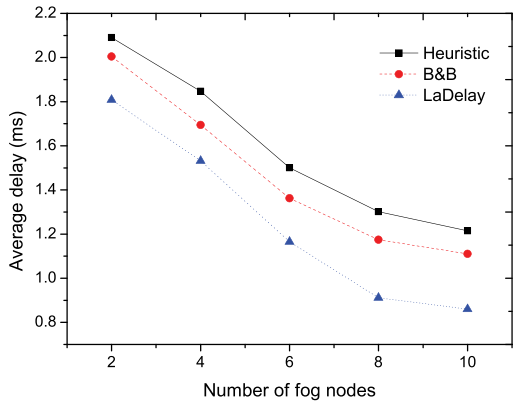


Fig. 8. Average delay under different number of fog nodes.

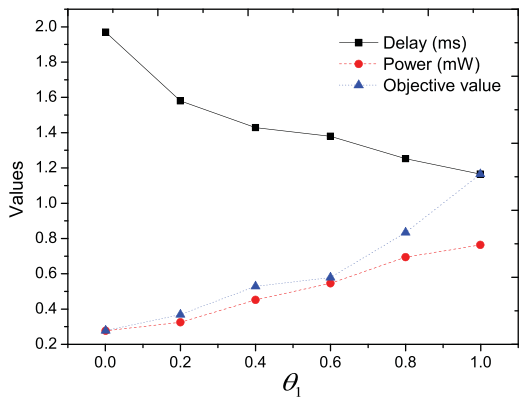


Fig. 7. Comparison of two metrics.

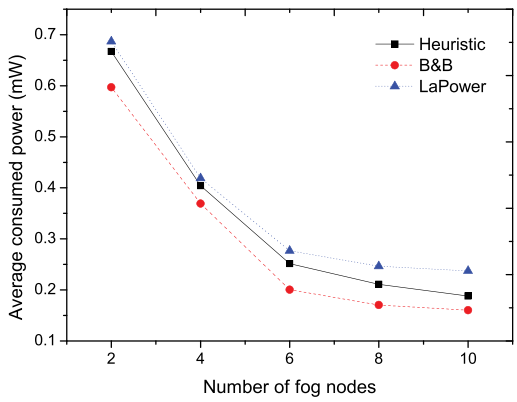


Fig. 9. Average consumed power under different number of fog nodes.

phenomenon also verifies the result of Fig. 3. Since it takes longer time for tasks to be offloaded to the cloud center, the LaDelay achieves the lowest average delay by offloading more tasks to fog nodes. On the contrary, the LaPower will offload relatively more tasks to the cloud center.

Fig. 7 shows the average delay, the consumed power, as well as the objective value of the Lagreedy algorithm under different settings of θ_1. In this set of experiments, we consider 20 tasks. A larger value of θ_1 represents a relatively larger weight of delay metric in the objective function. The blue dashed line represents the varying trend of the average delay under different values of θ_1. With θ_1 increasing from 0.001 to 0.999, the delay metric gradually dominates the value of the objective function. The goal of the Lagreedy algorithm varies from minimizing the power consumption, to finally mainly minimizing the average delay. Therefore, the average power consumption will increase due to a decreasing weight (i.e., θ_2) in the objective function, as shown in red dotted line. The black solid line represent the objective values, which are the weighted sum of the average delay and the power consumption.

Fig. 8 shows the average delay of the heuristic greedy algorithm, the branch-and-bound algorithm and the LaDelay algorithm under different numbers of fog nodes. Since the LaDelay algorithm can obtain the lowest average delay, we only compare its results with the branch-and-bound, and the heuristic greedy algorithm. With an increasing number of fog nodes, the average delay of all algorithms decreases accordingly. On one side, the tasks do not need to wait a long time before being offloaded, and hence the waiting delay is reduced. On the other side, there are more options of fog nodes, and tasks can choose the most suitable fog nodes with lower transmission and computation delay for computation offloading. In this way, both the communication delay and the computation delay will decrease accordingly.

Fig. 9 shows the average consumed power of all algorithms under different numbers of fog nodes. With an increasing number of fog nodes, more tasks can be offloaded to the fog nodes which require less transmission power. For the branch-and-bound and the heuristic greedy algorithms, all tasks are transmitted with the minimum transmission power. The lower bound of minimum transmission power will decrease with more options of fog nodes. For the LaPower algorithm, it will generate increasing pairs of task and fog node, which leads to a better converging result.

The average delay of the three algorithms under different values of the reliability threshold are compared in Fig. 10. As shown, the LaDelay algorithm keeps the lowest average delay compared with the branch-and-bound, and the heuristic algorithm in all settings. With the increasing of reliability threshold, the average delay decreases. This is because a higher reliability threshold requires larger transmission power, which also gives a higher data rate. Thus, the average delay will decrease correspondingly.

The average consumed power under different values of reliability threshold are shown in Fig. 11. With the increasing of reliability threshold, the minimum transmission power to guarantee the reliable communication is increased. Although the offloading solution would change with the updating of transmission power, tasks tend to require a larger transmission power for offloading, the total

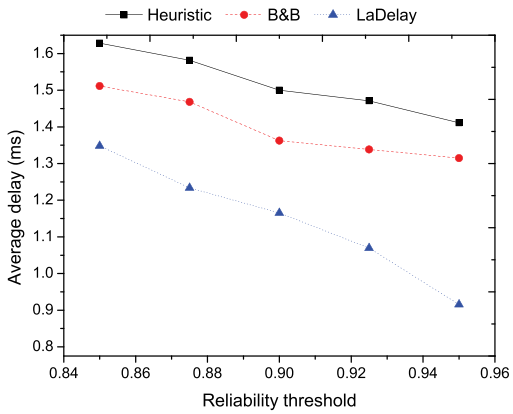


Fig. 10. Average delay under different reliability threshold.

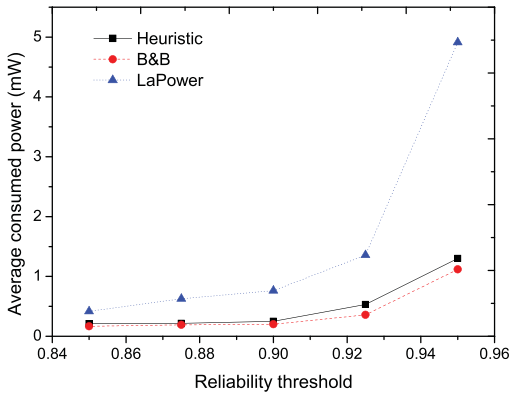


Fig. 11. Average consumed power under different reliability threshold.

consumed transmission power will increase. Specifically, when the reliability threshold increases from 0.92 to 0.95, the total transmission power increases significantly. In addition, in all settings, the branch-and-bound algorithm has a lower average transmission power than that of the heuristic greedy algorithm and the LaPower algorithm.

Fig. 12 shows the delays of all tasks as well as their average delay under different algorithms. We can observe that, the delay distribution of all tasks under the branch-and-bound algorithm and the LaDelay algorithm are very similar. The delay of most of tasks is close to the average delay of all tasks. In addition, the number of tasks with shorter delay (than the average delay) is more than that of tasks with longer delay. This is because it causes longer delay to transmit tasks to the cloud center, even fewer tasks being offloaded to the cloud center will increase the average delay of all tasks.

7 Conclusion

In this paper, we have presented a computation offloading architecture to meet the reliability-guaranteed and delay-sensitive requirements in emerging 5G application. To well explore the limited communication and computation resources of fog nodes, we have considered the dynamic resource queue model, and allocate tasks in a time sequence. In order to guarantee the reliable communication, we compute the minimum transmission power for each pair of task and computation node. The CVaR theory is applied to guarantee a reliable transmission under any worst-case channel

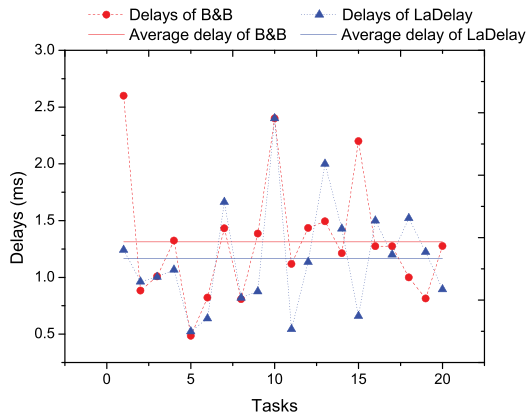


Fig. 12. Individual delay and average delay of different algorithms.

environments. On this basis, we have formulated the MPGAP, which is an integer programming problem to minimize the total delay of all tasks. The branch-and-bound algorithm is modified to obtain the optimal solution with a higher time complexity. In addition, we have proposed a heuristic greedy algorithm to obtain approximate solutions with much lower computation overhead. Furthermore, to analyze the effects of transmission power on the total delay of tasks, we have formulated a non-convex optimization problem, which considered both delay metric and power consumption metric. To solve its lagrange dual problem, the Lagreedy algorithm has been designed by combining the subgradient algorithm and the heuristic greedy algorithm. By assigning a larger weight to the power consumption metric, the Lagreedy algorithm has the approximate objective with the branch-and-bound algorithm, as well as the heuristic greedy algorithm. Through the comprehensive studies, we have demonstrated that the Lagreedy algorithm can only achieve lower average delay at the cost of high power consumption, and the branch-and-bound algorithm is able to achieve better performance on both the total delay and the power consumption.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant Nos. 61872049, 61572088, 61701124 and 61702258; the Frontier Interdisciplinary Research Funds for the Central Universities (Project No. 2018CDQYJSJ0034); the Sichuan Science and Technology Program under No. 2019YJ0105; the China Postdoctoral Science Foundation under Grant No. 2016M591852; and the Postdoctoral research funding program of Jiangsu Province under Grant No. 1601257C; and US MURI AFOSR MURI 18RT0073, NSF CNS-1717454, CNS- 1731424, CNS-1702850, CNS-1646607.

References

[1] J. Rao and S. Vrzic, "Packet duplication for URLLC in 5G: Architectural enhancements and performance analysis," *IEEE Netw.*, vol. 32, no. 2, pp. 32–40, Mar. 2018.

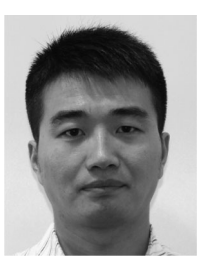
[2] H. Chen, R. Abbas, P. Cheng, M. Shirvanimoghaddam, W. Hardjawana, W. Bao, Y. Li, and B. Vucetic, "Ultra-reliable low latency cellular networks: Use cases, challenges and approaches," *IEEE Commun. Magazine*, vol. 56, no. 12, pp. 119–125, Dec. 2018.

[3] G. Pocovi, H. Shariatmadari, G. Berardinelli, K. Pedersen, J. Steiner, and Z. Li, "Achieving ultra-reliable low-latency communications: Challenges and envisioned system enhancements," *IEEE Netw.*, vol. 32, no. 2, pp. 8–15, Mar. 2018.

[4] K. Liu and V. C. Lee, "On-demand broadcast for multiple-item requests in a multiple-channel environment," *Inf. Sci.*, vol. 180, no. 22, pp. 4336–4352, 2010.
[5] W. Sun, E. G. Strom, F. Brannstrom, Y. Sui, and K. C. Sou, "D2D-based V2V communications with latency and reliability constraints," in *Proc. IEEE Globecom Workshops*, Dec. 2014, pp. 1414–1419.
[6] T. C. Chiu, W. H. Chung, A. C. Pang, Y. J. Yu, and P. H. Yen, "Ultra-low latency service provision in 5G fog-radio access networks," in *Proc. IEEE 27th Annu. Int. Symp. Pers. Indoor Mobile Radio Commun.*, Sep. 2016, pp. 1–6.
[7] F. Bonomi, R. Milito, P. Natarajan, and J. Zhu, "Fog computing: A platform for internet of things and analytics," *Big Data Internet Things: A Roadmap Smart Environments*, vol. 546, pp. 169–186, Mar. 2014.
[8] M. H. Chen, B. Liang, and M. Dong, "Joint offloading and resource allocation for computation and communication in mobile cloud with computing access point," in *Proc. IEEE Conf. Comput. Commun.*, May 2017, pp. 1–9.
[9] T. T. Nguyen and B. L. Long, "Joint computation offloading and resource allocation in cloud based wireless hetnets," in *Proc. IEEE Global Commun. Conf.*, Dec. 2017, pp. 1–6.
[10] M. S. Elbamby, M. Bennis, and W. Saad, "Proactive edge computing in latency-constrained fog networks," in *Proc. Eur. Conf. Netw. Commun.*, Jun. 2017, pp. 1–6.
[11] C. Liu, M. Bennis, and H. V. Poor, "Latency and reliability-aware task offloading and resource allocation for mobile edge computing," in *Proc. IEEE Globecom Workshops*, Dec. 2017, pp. 1–7.
[12] J. J. Nielsen, R. Liu, and P. Popovski, "Ultra-reliable low latency communication using interface diversity," *IEEE Trans. Commun.*, vol. 66, no. 3, pp. 1322–1334, Mar. 2018.
[13] G. J. Sutton, J. Zeng, R. P. Liu, W. Ni, D. N. Nguyen, B. A. Jayawickrama, X. Huang, M. Abolhasan, and Z. Zhang, "Enabling ultra-reliable and low-latency communications through unlicensed spectrum," *IEEE Netw.*, vol. 32, no. 2, pp. 70–77, Mar. 2018.
[14] Z. Zheng, L. Song, Z. Han, G. Y. Li, and H. V. Poor, "A stackelberg game approach to proactive caching in large-scale mobile edge networks," *IEEE Trans. Wireless Commun.*, vol. 17, no. 8, pp. 5198–5211, Aug. 2018.
[15] H. Wang, J. Wang, G. Ding, and Z. Han, "D2D communications underlaying wireless powered communication networks," *IEEE Trans. Veh. Technol.*, vol. 67, no. 8, pp. 7872–7876, Aug. 2018.
[16] Y. Yu, X. Bu, K. Yang, and Z. Han, "Green fog computing resource allocation using joint benders decomposition, dinkelbach algorithm, and modified distributed inner convex approximation," in *Proc. IEEE Int. Conf. Commun.*, May 2018, pp. 1–6.
[17] R. Deng, R. Lu, C. Lai, T. H. Luan, and H. Liang, "Optimal workload allocation in fog-cloud computing toward balanced delay and power consumption," *IEEE Internet Things J.*, vol. 3, no. 6, pp. 1171–1181, Dec. 2016.
[18] T. Zhao, S. Zhou, X. Guo, and Z. Niu, "Tasks scheduling and resource allocation in heterogeneous cloud for delay-bounded mobile edge computing," in *Proc. IEEE Int. Conf. Commun.*, May 2017, pp. 1–7.
[19] J. Liu and Q. Zhang, "Offloading schemes in mobile edge computing for ultra-reliable low latency communications," *IEEE Access*, vol. 6, pp. 12 825–12 837, Feb. 2018.
[20] K. Liu, L. Feng, P. Dai, V. C. S. Lee, S. H. Son, and J. Cao, "Coding-assisted broadcast scheduling via memetic computing in sdn-based vehicular networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 8, pp. 2420–2431, Aug. 2018.
[21] H. Zhang, Q. Zhang, and X. Du, "Toward vehicle-assisted cloud computing for smartphones," *IEEE Trans. Veh. Technol.*, vol. 64, no. 12, pp. 5610–5618, Dec. 2015.
[22] Y. Xiao and M. Krunz, "QoE and power efficiency tradeoff for fog computing networks with fog node cooperation," in *Proc. IEEE Conf. Comput. Commun.*, May 2017, pp. 1–9.
[23] J. Du, L. Zhao, J. Feng, and X. Chu, "Computation offloading and resource allocation in mixed fog/cloud computing systems with min-max fairness guarantee," *IEEE Trans. Commun.*, vol. 66, no. 4, pp. 1594–1608, Apr. 2018.
[24] A. K. Sadek, Z. Han, and K. J. R. Liu, "Distributed relay-assignment protocols for coverage expansion in cooperative wireless networks," *IEEE Trans. Mobile Comput.*, vol. 9, no. 4, pp. 505–515, Apr. 2010.
[25] B. Li, Y. Rong, J. Sun, and K. L. Teo, "A distributionally robust minimum variance beamformer design," *IEEE Signal Process. Lett.*, vol. 25, no. 1, pp. 105–109, Jan. 2018.
[26] B. Li, J. Sun, H. Xu, and M. Zhang, "A class of two-stage distributionally robust games," *J. Ind. Manag. Optimization*, vol. 15, no. 1, pp. 387–400, Jan. 2019.
[27] B. Li, X. Qian, J. Sun, K. L. Teo, and C. Yu, "A model of distributionally robust two-stage stochastic convex programming with linear recourse," *Appl. Math. Modelling*, vol. 58, pp. 86–97, 2018.
[28] S. Zymler, D. Kuhn, and B. Rustem, "Distributionally robust joint chance constraints with second-order moment information," *Math. Program.*, vol. 137, no. 1, pp. 167–198, Feb. 2013.
[29] B. Li, Y. Rong, J. Sun, and K. L. Teo, "A distributionally robust linear receiver design for multi-access space-time block coded mimo systems," *IEEE Trans. Wireless Commun.*, vol. 16, no. 1, pp. 464–474, Jan. 2017.
[30] G. T. Ross and R. M. Soland, "A branch and bound algorithm for the generalized assignment problem," *Math. Program.*, vol. 8, no. 1, pp. 91–103, Dec. 1975.
[31] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.
[32] S. Guo, B. Xiao, Y. Yang, and Y. Yang, "Energy-efficient dynamic offloading and resource scheduling in mobile cloud computing," in *Proc. IEEE Int. Conf. Comput. Commun.*, Apr. 2016, pp. 1–9.
[33] C. Perfecto, J. D. Ser, and M. Bennis, "Millimeter-wave v2v communications: Distributed association and beam alignment," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 9, pp. 2148–2162, Sep. 2017.

Junhua Wang received the BS and PhD degrees in computer science from Chongqing University, in 2014 and 2019, respectively. Since 2019, she has been working with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics. Her research interests include mobile computing, vehicular ad-hoc networks, and wireless networks.

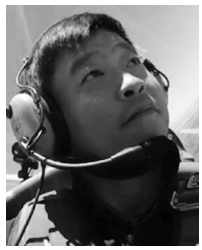
Kai Liu (S'07-M'12) received the PhD degree in computer science from the City University of Hong Kong, in 2011. From December 2010 to May 2011, he was a visiting scholar with the Department of Computer Science, University of Virginia. From 2011 to 2014, he was a postdoctoral fellow with Singapore Nanyang Technological University, the City University of Hong Kong, and Hong Kong Baptist University. He is currently an assistant professor with the College of Computer Science, Chongqing University, China. His research interests include internet of vehicles, mobile computing, pervasive computing, and big data. He is a member of the IEEE.



Bin Li (M'18-SM'18) received the bachelor's degree in automation and the master's degree in control science and engineering from the Harbin Institute of Technology, Harbin, China, in 2005 and 2008, respectively, and the PhD degree in mathematics and statistics from Curtin University, Perth, WA, Australia, in 2011. From 2012 to 2014, he was a research associate with the School of Electrical, Electronic, and Computer Engineering, the University of Western Australia, Perth, WA, Australia. From 2014 to 2017, he was a research fellow with the Department of Mathematics and Statistics, Curtin University. He is currently a research professor with the School of Electrical Engineering and Information, Sichuan University, Chengdu, China. His research interests include signal processing, wireless communications, optimization, and optimal control. He is a senior member of the IEEE.

Tingting Liu (M'12) received the BS degree in communication engineering from the Nanjing University of Science and Technology, Nanjing, China, in 2005, and the PhD degree in information and communication engineering from the Nanjing University of Science and Technology, Nanjing, China, in 2011. Since 2011, she has been with the School of Communication Engineering at the Nanjing Institute of Technology, China. Currently, she is also a postdoctor at the Nanjing University of Science and Technology. Her research interests include game theory, caching-enabled systems, device-to-device networks, and cognitive radio networks. She is a member of the IEEE.

Ruoguang Li (S'15) received the bachelor's degree from the Nanjing University of Posts and Telecommunications (NUPT), Nanjing, China, in 2013. He is currently working towards the PhD degree in the School of Electronic Engineering of Beijing University of Posts and Telecommunications (BUPT). His research interests include the area of resource allocation in next generation networks and physical layer security in cooperative networks. He is a student member of the IEEE.



Zhu Han (S'01-M'04-SM'09-F'14) received the BS degree in electronic engineering from Tsinghua University, in 1997, and the MS and PhD degrees in electrical and computer engineering from the University of Maryland, College Park, in 1999 and 2003, respectively. From 2000 to 2002, he was an R&D engineer of JDSU, Germantown, Maryland. From 2003 to 2006, he was a research associate at the University of Maryland. From 2006 to 2008, he was an assistant professor at Boise State University, Idaho. Currently, he is a John and Rebecca Moores professor with the Electrical and Computer Engineering Department as well as in the Computer Science Department, University of Houston, Texas. He is also a chair professor at the National Chiao Tung University, ROC. His research interests include wireless resource allocation and management, wireless communications and networking, game theory, big data analysis, security, and smart grid. He received an NSF Career Award in 2010, the Fred W. Ellersick Prize of the IEEE Communication Society in 2011, the EURASIP Best Paper Award for the Journal on Advances in Signal Processing in 2015, IEEE Leonard G. Abraham Prize in the field of Communications Systems (best paper award in IEEE JSAC) in 2016, and several best paper awards in IEEE conferences. Currently, he is an IEEE Communications Society Distinguished lecturer from 2015-2018. He has been a 1 percent highly cited researcher since 2017 according to the Web of Science. He is a fellow of the IEEE.