This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCOMM.2019.2920348, IEEE Transactions on Communications

Offloading Optimization and Bottleneck Analysis for Mobile Cloud Computing

Di Han, *Student Member, IEEE*, Wei Chen, *Senior Member, IEEE*,
Bo Bai, *Senior Member, IEEE*, Yuguang Fang, *Fellow, IEEE*

***Abstract*—Mobile cloud computing systems, or simply mobile clouds, have attracted tremendous attention because they allow mobile devices with limited computational resources to offload complex computations. However, due to the channel uncertainty and the complexity of a computation task, mobile computation offloading may suffer from poor outage performance that the offloaded task cannot be completed within the desired delay constraint. Thus, how to efficiently identify and overcome the *outage bottleneck*, which could be used to optimize resource allocation schemes and improve the system performance effectively, is an open problem. In this paper, we shall develop a unified framework that minimizes the overall outage probability in various mobile computation offloading scenarios. More specifically, the *outage bottleneck* is defined and identified by adopting asymptotic analysis, without any need of the accurate outage probabilities in both transmissions and computations. To overcome the *outage bottleneck*, resource pairing, matching, and allocation policies are investigated. Both theoretical analysis and numerical results show that the *outage bottleneck* relies on not only the availability of spectrum and computation resources but also the probability distributions of computation complexities of the computation tasks.**

***Index Terms*—Mobile cloud computing, computation offloading, outage bottleneck, power consumption.**

I. Introduction

In recent years, the demand for mobile devices to execute heavy applications with required delay constraints has exhibited consistent growth. However, mobile devices are usually equipped with limited resources such as computation capability [3] and battery power [4], due to the limited form factor, e.g., physical size. The conflicting design consideration between resource-hungry applications and resource-constrained mobile devices hence poses a significant challenge for the future mobile computing development. To overcome this challenge, mobile computation offloading has emerged as a potential technique, shifting complex computation tasks to more powerful computation facilities such as cloud servers or mobile cloud computation systems, and thus has attracted intensive from both academia and industries [5], [6] and [7].

Unfortunately, to offload a computation task, data may have to be transmitted to the cloud, which may suffer from long latency, and thus mobile clouds in proximity are preferred. On one hand, if a mobile cloud is used, the computation resource may not be enough to complete the computation task on time. Furthermore, the power consumption on mobile devices is also an important factor to consider. Therefore, the computation management is necessary in mobile cloud computing. Then, how to minimize the power consumption in a mobile device while completing the required computation task on time is highly challenging, but of paramount importance. This paper is to tackle this problem.

As mentioned, offloading a task from a mobile device to the computation resources in the cloud does reduce the workload at the device, but will consume transmission power between the device and cloud [4]. Due to the limited battery capacity of mobile devices [8], energy efficiency becomes a critical issue in wireless transmission [9], and computation offloading [10] and [11]. Thus, the performance analysis of the end-to-end computation management between the mobile device and the computation resources is significantly important.

In this paper, we consider a mobile computation offloading system composed of a mobile device connecting with multiple computation resources, simply called *mobile cloud*, through multiple wireless channels. During the computation offloading, each input task in a mobile device, ready for uploading to the mobile cloud, could be divided into multiple subtasks and then be executed in parallel on different computation resources. However, the dynamic environment can cause additional problems [12]. For example, the transmitted task may not reach the computation resources, or the task executed on the server cannot be completed when it has to be returned to the mobile device. In addition, we assume that the input tasks are latency-sensitive. Thus, not only the transmission of each input task and its output results but also the computation of the task must be completed within the required time latencies. Otherwise, the timeout will be triggered, resulting in task resubmission, which we simply state that the system is in outage state. The aforementioned outage event is composed of transmission outage and computation outage due to the uncertainty in wireless channels, and the lack of computation resource in the mobile cloud. The outage probability is considered as the performance metric. Both transmission and computation capabilities could be improved by increasing the power of

This work is supported in part by Beijing Natural Science Foundation (4191001), the National Science Foundation of China under Grant Nos. 61671269 and 61621091, and the National Program for Special Support for Eminent Professionals of China (National 10,000 Talent Program). The work of Y. Fang is supported in part by U.S. National Science Foundation under grant CNS-1717736. The preliminary work has been presented at IEEE GlobalSIP'2016 [1] and IEEE GLOBECOM'2017 [2].

D. Han and W. Chen are with the Department of Electronic Engineering, Tsinghua University, Beijing 100084, China (e-mail: hd15@mails.tsinghua.edu.cn, wchen@tsinghua.edu.cn).

B. Bai was with the Department of Electronic Engineering, Tsinghua University, Beijing 100084 China. He is now with Future Network Theory Lab, Huawei Technologies Co.,Ltd., Shatin, Hong Kong (e-mail: baibo8@huawei.com).

Y. Fang is with the Department of Electrical and Computer Engineering and University of Florida, Gainesville, FL 32611, USA (e-mail: fang@ece.ufl.edu).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCOMM.2019.2920348, IEEE Transactions on Communications

the system in order to lower the outage probability. However, the same level of improvement on outage performance may require different levels of power consumption in transmission and computation. The major factor that restricts the reduction of outage probability of the system even with power increase is considered as the *outage bottleneck*, which could be used to optimize resource allocation schemes to improve the system performance efficiently. Unfortunately, it is difficult to obtain explicit expressions for the outage probabilities of transmission and computation. Thus, how to identify and efficiently overcome the outage bottleneck remains as an open problem. In our previous work, we focus on this problem in a simple system with one channel and one computation resource [1] and then extend the bottleneck into a system with multiple channels and computation resources [2]. However, a theoretical framework for the bottleneck analysis which can be applied in more different scenarios is still required.

The main contribution of this paper is to develop a unified framework from an asymptotic analysis perspective to analyze and minimize the overall outage probability, and identify the *outage bottleneck* of the considered system. Our unified framework jointly considers the task division and channel allocation in the computation offloading, and can be applied in different offloading scenarios, e.g., distributed or centralized computation, and with or without the information of the available computation capabilities. In each considered offloading scenario, the corresponding policy and algorithm are designed to minimize the overall outage probability. Then, by analyzing the features of complexities of input task and background tasks, necessary and sufficient conditions to identify the *outage bottleneck* of the system in each considered offloading scenario can be obtained without knowing the accurate expressions of the outage probabilities of transmission and computation phases, which can be used to evaluate the outage performance and design an efficient strategy to improve the outage performance for existing and future mobile computation offloading systems.

Throughout this paper, $o(x)$ denotes the higher order infinitesimal of x and $\max\{\mathbf{a}\}$ denotes the maximal element in the vector $\mathbf{a}$. The symbol $\sim$ stands for equivalent infinitesimal. The indicator function $\mathbb{1}(a)$ is equal to 1 if $a \geq 1$; otherwise, it is equal to 0. The rest of this paper is organized as follows. The related work is summarized in Section II. Section III presents the system model and the framework for the outage performance analysis. The analysis of outage performance and *outage bottleneck* are given in Section IV and Section V. Numerical results are shown in Section VI to verify the theoretical results. Finally, Section VII concludes the paper. For better illustration, most of the proofs for revealing our analytical results are put in the appendix and the most important notations are listed in Table I.

TABLE I
SUMMARY OF NOTATIONS

Symbol	Definition
$\mathcal{C}$	The set of the computation resources
$\mathcal{S}$	The set of the wireless channels
$\mathcal{C}_{\rm ava}$	The set of the computation resources received the whole task after the upload phase
M, N	The numbers of the computation resources and the wireless channels
$T_{\rm s}$, T_1, T_2	The lengths of the time slot, the upload phase, and the download phase
$l_{\rm in}$, $l_{\rm out}$	The data sizes of the input task and the output result
N_0	The additive white Gaussian noise power
$\xi_{\rm c}$	The power consumption of computation
$\xi_{\rm t}$	The power consumption of transmission
w	The computation requirement of the input task
q_i	The computation requirement of the background task in the computation resource c_i
$C(\xi_{\rm c})$	The computation capability when the computation power consumption is $\xi_{\rm c}$
$\mathbf{H}_{\rm u}\left(\mathbf{h}_{\rm u}^i\right)$	The channel gain matrix (vector for the computation resource c_i) in the upload phase
$\mathbf{H}_{\rm d}\left(\mathbf{h}_{\rm d}^i\right)$	The channel gain matrix (vector for the computation resource c_i) in the download phase
$\mathbf{A}$	The channel allocation result in the download phase
$\mathbf{x}$	The subtask division result in the computation phase
$R_{\rm u}^i$, $R_{\rm c}^i$, $R_{\rm d}^i$	The successful ratios of computation, upload, and download phases in the computation resource c_i
R	The successful offloading ratio, which is the percentage of the task be offloaded successfully
$p_{{\rm Type}k}(\xi_{\rm t},\xi_{\rm c})$	The overall outage probability in the scenario of Type $k \in \{1,2,3,4\}$
$p_{\rm out}^{{\rm c},k}(M,\xi_{\rm c})$, $p_{\rm out}^{{\rm t},k}(N,\xi_t)$	The computation and transmission outage probability partitions in the scenario of Type $k \in \{1,2,3,4\}$
φ_1, ϕ_1, φ_2, ϕ_2	Parameters used to identify the bottlenecks in the four considered scenarios

II. RELATED WORK

There has been a lot of work on the energy efficiency of the computation offloading in the literature. In [10], Lin et al. proposed a framework for computation offloading based on execution time and energy consumption, aiming to shorten response time and reduce energy consumption. In [13], Jiang et al. designed a scheduling scheme for energy-efficient computation offloading for multi-core mobile devices and developed an online algorithm based on Lyapunov optimization. In [14], Zhang et al. proposed a theoretical framework for mobile cloud computing under the stochastic wireless channel to determine the optimal operational regions to save energy. A dynamic computation offloading policy was developed by You et al. in [15] and by Mao et al. in [11] to maximize the energy saving by considering microwave energy transfer and energy harvesting, respectively. In [14], Zhang et al. obtained closed-form solutions to decide the optimal application-execution condition under which either the mobile execution or the cloud execution is more energy-efficient. In [16], Guo et al. integrated dynamic offloading with resource scheduling firstly to minimize the joint energy consumption and application completion time under completion time deadline constraint and task-precedence requirement. The task In this work, we also focus on the energy efficiency of the computation offloading through performance analysis in different scenarios.

More recently, various mobile computation offloading systems have been considered. It was shown that energy efficiency could be improved by jointly optimizing the allocation of computation and transmission resources [17]-[18]. In [17], Salmani and Davidson jointly optimized the allocation of the computational and radio resources to reduce energy consumption by exploiting the capabilities of the multiple access channel. In [19], Guo et al. considered the impact of task dependencies

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCOMM.2019.2920348, IEEE Transactions on Communications

on computation completion time and then provided an energy-efficient dynamic offloading and resource scheduling policy to reduce energy consumption. In [20], Ti and Le studied the joint computation offloading and resource allocation problem in the two-tier wireless heterogeneous network. In [21], Chen et al. aimed to find the local optimum decision on offloading and allocation of computation and communication resources to minimize a weighted cost in a general mobile cloud computing system and an algorithm is given to obtain a local optimum. In [22], Wang et al. offered a joint energy minimization and resource allocation solution that could improve the system performance and save energy in a novel C-RAN architecture with the mobile cloud co-located with the BBU in the cloud pool. In [18], Mao et al. dynamically managed the radio and computational resources to cope with the time-varying computation demands and wireless fading channels. In this paper, we develop a unified framework to optimize the performance and improve energy efficiency of various mobile cloud computing systems from the perspective of outage probability.

Furthermore, by increasing the power consumption of both transmission and computation, the Quality-of-Service (QoS) of mobile users in mobile computation offloading systems could be improved significantly [4][5]. However, the identical increase in power consumption of transmission and computation might result in different improvement of the system performance and QoS. Hence, the transmission-computation power consumption tradeoff is much more important and meaningful, which has the potential to offer the most effective way to improve the system performance and has been studied in [4] and our previous work [1] and [2]. Moreover, Ma et al. presented a survey of energy-efficient computation offloading technologies and summarized that there is a tradeoff in energy consumption between transmission and local computation in [23]. In this paper, to better utilize the resource in computation offloading systems, we define the major limiting factor of the system as *outage bottleneck*.

III. System Model

Consider a mobile computation offloading system that consists of one mobile device and M computation resources, through which the mobile device can offload the input tasks to the mobile cloud as shown in Fig. 1. The computation resources, whose set is denoted by $\mathcal{C} = \{c_1, \ldots, c_M\}$, are distributed across different locations. Assume that the mobile device transmits with each computation resource through $N(N \geq M)$ wireless channels, whose set is denoted by $\mathcal{S} = \{s_1, \ldots, s_N\}$. Due to the lack of computation capability, the input tasks of the mobile device need to be computed through computation offloading with the aid of the mobile cloud. In this way, a significantly improved computation experience can be achieved. Moreover, all control operations of the computation offloading, i.e., transmissions and computations, are managed by a controller [12] in the considered system, which is in charge of processing the requests and provides mobile devices with the corresponding cloud services, e.g., computation resources.

The procedure of the computation offloading can be divided into three phases: upload, computation, and download. Specifically, each input task is set to be uploaded to the computation resource and then be computed in the mobile cloud. After the computation phase, the results of the task is transmitted back to the mobile device.

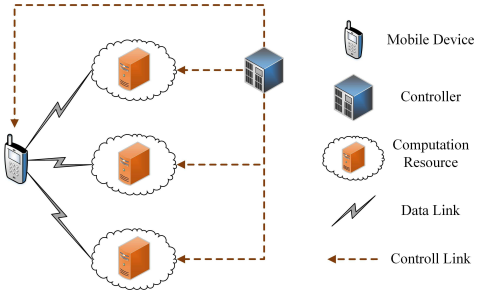


Fig. 1. Illustration of mobile computation offloading system with one mobile device connecting with multiple computation resources, where a controller is responsible for all control plane decisions.

Let R_{s}^i denote the successful offloading ratio of the computation resource c_i, which is the percentage of one input task that can be offloaded successfully in the computation resource c_i under the delay constraints. This successful offloading ratio R_{s}^i is determined by R_{c}^i, R_{u}^i, and R_{d}^i, which are successful ratios of computation, upload, and download phases in the computation resource c_i. In addition, if an input task cannot be uploaded completely, i.e., $R_{\mathrm{u}}^i < 1$, then it cannot be computed in the computation resource c_i, causing $R_{\mathrm{s}}^i = 0$; otherwise $R_{\mathrm{s}}^i = \min\{R_{\mathrm{d}}^i, R_{\mathrm{c}}^i\}$. Therefore, R_{s}^i is given by

$$R_{\mathrm{s}}^i = \min\{\mathbb{1}\left(R_{\mathrm{u}}^i\right), R_{\mathrm{d}}^i, R_{\mathrm{c}}^i\}. \tag{1}$$

We derive next explicit expressions of R_{c}^i, R_{u}^i, and R_{d}^i with different task division and channel allocation methods as follows.

A. Concurrent Computation and Transmissions in Time Domain

We consider a time-slotted system, in which the time is divided into time slots with a fixed length T_{s} and is indexed by an integer k. Assume that each input task of the mobile device arrives at the beginning of the time slot, whose data size (in bits) is a fixed value l_{in}. Moreover, assume that the data size of each task's output result is also a fixed value l_{out}. As depicted in Fig. 2, each time slot can be divided into two parts: the upload phase T_1 and the download phase T_2, where $T_{\mathrm{s}} = T_1 + T_2$.

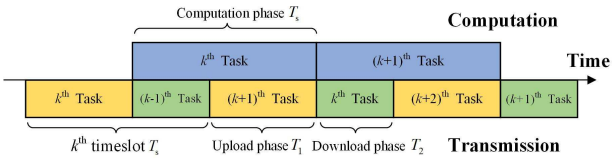


Fig. 2. Time scheduling with concurrent computation and transmissions.

For the task arriving at the beginning of the k-th time slot, it is indexed as k-th task and uploaded from the mobile device to the mobile cloud in the upload phase of the k-th time slot, and then its output result is downloaded from the mobile cloud

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCOMM.2019.2920348, IEEE Transactions on Communications

in the download phase of the $(k+1)$-th time slot. In this way, the wireless channels and computation resources can be well utilized by concurrent computation and transmissions in time domain, which has been used in our previous work [1] and [2]. Therefore, the delay constraints for upload, download, and computation phases are given by T_1, T_2, and T_s, respectively.

Since both the computation and transmission aspects play a key role, we next introduce the computation and transmission models, respectively. Without loss of generality, we focus on the offloading procedure and outage probability analysis for the task arriving in the k-th time slot because the outage events of different tasks are independent and identically distributed.

B. Computation Model

The computation capability required by the task is measured by the number of CPU cycles, denoted by w, which is a random variable with given probability density function $f_w(x)$. In addition, there are also background tasks requiring computation capability in each computation resource. Let q_i denote the computation capability required by background tasks in the computation resource c_i during the computation phase, which is also a random variable with given probability density function $f_q(x)$. Assume that w and q_i $(1 \leq i \leq M)$ are independent with each other.

The computation capability of each computation resource is a finite value C, which is the total number of CPU cycles supported by the computation resource during one computation phase, i.e., one time slot T_s. Let f_c denote the clock frequency of the CPU in each computation resource. Thus, the computation capability of each computation resource is given by

$$C = f_c T_s. \tag{2}$$

According to [24] and [25], the power consumption of computation can be expressed as

$$\xi_c = \kappa {f_c}^3, \tag{3}$$

where κ is the effective switched capacitance depending on the chip architecture of CPU. Then, the computation capability C can be rewritten as

$$C(\xi_c) = k_c \xi_c^{\frac{1}{3}}, \tag{4}$$

where $k_c = T_s \kappa^{-\frac{1}{3}}$. Consequently, the successful ratio of the computation phase in the computation resource c_i is given by

$$R_c^i(w, q_i, \xi_c) = \frac{C(\xi_c) - q_i}{w}. \tag{5}$$

C. Transmission Model

The considered system is assumed to undergo *Rayleigh block-fading channel* [26]. In each time-slot, the data link between the mobile device and one computation resource through each wireless channel has the independent and identical distribution (i.i.d.) in fading condition, which are assumed to be circularly symmetric Gaussian distribution $\mathcal{CN}(0,1)$. The channel gain keeps fixed during one time-slot, while the channel gains are i.i.d. in different time slots.

Let $\mathbf{h}_u^i = [h_u^{i1} \ldots, h_u^{iN}]^T$ and $\mathbf{h}_d^i = [h_d^{i1} \ldots, h_d^{iN}]^T$ denote the channel gain vectors for the computation resource c_i in upload and download phases, where h_u^{ij} and h_d^{ij} are channel gains of upload and download links between the mobile device and computation resource c_i over channel s_j, respectively. Since computation resources are in different locations, it is reasonable to assume that the channels between the mobile device and different computation resources undergo independent fading. Therefore, both h_u^{ij} and h_d^{ij}, $\forall 1 \leq i \leq M$, are assumed to follow independent and identical distribution $\mathcal{CN}(0,1)$. Furthermore, the channel gain matrices in upload and download phases are denoted by $\mathbf{H}_u = [\mathbf{h}_u^1, \ldots, \mathbf{h}_u^M]$ and $\mathbf{H}_d = [\mathbf{h}_d^1, \ldots, \mathbf{h}_d^M]$.

In the upload phase, to take full advantage of independent fading and maximize the diversity gain of transmission [27], the task is set to be transmitted to each computation resource through the channel with maximal transmission capability, which is the point-to-multipoint communication. The broadcasting approach is not considered in the upload phase to avoid broadcast storm when there are multiple mobile devices in realistic scenarios. In this way, the mobile device can transmit the task to multiple computation resources through one wireless channel simultaneously and the transmission power consumption might be reduced compared to the broadcasting approach. Thus, the successful ratio of the upload phase in the computation resource c_i is given by

$$R_u^i(\mathbf{h}_u^i, \xi_t) = \frac{T_1 \log(1 + |\max\{\mathbf{h}_u^i\}|^2 \frac{\xi_t}{N_0})}{l_{\text{in}}}, \tag{6}$$

where ξ_t is the power consumption of transmission and N_0 is the power of the background noise.

In the download phase, there might have multiple computation resources requiring channels to transmit results back to the mobile device simultaneously. The channels occupied by the computation resource c_i in the download phase are denoted by $\mathbf{a}_i = [a_{i1}, \ldots, a_{iN}]^T$, where $a_{ij} = 1$ if the channel s_j is allocated to the computation resource c_i; otherwise $a_{ij} = 0$. The channel allocation result can be represented by the matrix $\mathbf{A} = [\mathbf{a}_1, \ldots, \mathbf{a}_M]$. We notice that one channel cannot be allocated to multiple computation resources in order to avoid interference. Thus, we have the constraint of channel allocation method, which is given by

$$\sum_{i=1}^{M} a_{ij} \leq 1, \ \forall j = 1, \ldots, N. \tag{7}$$

Then the successful ratio of the download phase in the computation resource c_i is given by

$$R_d^i(\mathbf{h}_d^i, \mathbf{a}_i, \xi_t) = \frac{T_2 \log(1 + |\max\limits_{s_j \in \mathcal{S}}\{a_{ij} h_d^{ij}\}|^2 \frac{\xi_t}{N_0})}{l_{\text{out}}}. \tag{8}$$

D. Theoretical Framework for Outage Analysis

Assume that each input task can be divided into multiple subtasks and then be computed in parallel on multiple computation resources. For the sake of simplicity, the separated results of all subtasks can be combined into a whole result in the mobile device with negligible computation capability

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCOMM.2019.2920348, IEEE Transactions on Communications

consumption [14] and [28], i.e., time delay can be ignored. The fractions of the original task in different computation resources are denoted by vector $\mathbf{x} = [x_1, \ldots, x_M]$, where x_i is the fraction of the input task allocated to the computation resource c_i. The subtask division method $\mathbf{x}$ can be designed by the controller, where the normalization condition holds and thus the sum of subtasks must cover the whole input task, i.e.,

$$\sum_{i=1}^{M} x_i = 1, \tag{9}$$

where $0 \leq x_i \leq 1$. Similar to the task model in [14] and [28], we assume the input task is uniformly splittable, i.e., a fraction of x_i of the original task has $x_i l_{\text{in}}$ input data bits, $x_i l_{\text{out}}$ output data bits, and requires $x_i w$ CPU cycles for processing. For some typical offloading applications, such as image/video/voice recognition and file scanning, can satisfy this assumption [29] and then can be applied in the following framework.

Based on the above analysis, the successful offloading ratio R, which is the percentage of one input task that can be offloaded successfully, is given by

$$R(w, \mathbf{q}, \mathbf{H}_{\text{u}}, \mathbf{H}_{\text{d}}, \mathbf{x}, \mathbf{A}, \xi_{\text{c}}, \xi_{\text{t}}) = \sum_{i=1}^{M} R_{\text{s}}^{i} = \sum_{i=1}^{M} \min\{\mathbb{1}\left(R_{\text{u}}^{i}(\mathbf{h}_{\text{u}}^{i}, \xi_{\text{t}})\right), R_{\text{d}}^{i}(\mathbf{h}_{\text{d}}^{i}, \mathbf{a_i}, \xi_{\text{t}}), R_{\text{c}}^{i}(w, q_i, \xi_{\text{c}}), x_i\}. \tag{10}$$

Thus, the system is in outage state if

$$R(w, \mathbf{q}, \mathbf{H}_{\text{u}}, \mathbf{H}_{\text{d}}, \mathbf{x}, \mathbf{A}, \xi_{\text{c}}, \xi_{\text{t}}) < 1, \tag{11}$$

which represents that the offloading of the input task cannot be completed fully under the required delay constraints. From Eqs. (9)-(11), the system is in outage state if

$$\mathbb{1}\left(R_{\text{u}}^{i}(\mathbf{h}_{\text{u}}^{i}, \xi_{\text{t}})\right) < x_i \tag{12}$$

or

$$R_{\text{d}}^{i}(\mathbf{h}_{\text{d}}^{i}, \mathbf{a_i}, \xi_{\text{t}}) < x_i \tag{13}$$

or

$$R_{\text{c}}^{i}(w, q_i, \xi_{\text{c}})\} < x_i \tag{14}$$

holds for any computation resource $c_i \in \mathcal{C}$. Then, a unified framework for outage performance analysis, where efficiency and fairness are both taken into consideration, has already been presented. In this framework, $\mathbf{x}$ is a design parameter, which can be designed by the controller based on $\mathbf{q}$ and $\mathbf{H}_{\text{u}}$ after the upload phase. Next, $\mathbf{A}$ is also a design parameter, which can be determined based on $\mathbf{x}$ and $\mathbf{H}_{\text{d}}$ after the computation phase. The feedback of the above information will also cause the signaling overhead.

IV. Outage Probability Analysis

In this section, we focus on outage probability analysis in different offloading scenarios based on the unified framework in Eq. (10).

In the upload phase of each considered offloading scenario, the input task is set to be transmitted to all computation resources over all available channels. If the input task cannot be received by any one of the computation resources successfully, the upload phase in this computation resource fails with probability given by

$$p_{\text{br}}^{\text{out}}(\xi_{\text{t}}) = \Pr\left\{R_{\text{u}}^{i}(\mathbf{h}_{\text{u}}^{i}, \xi_{\text{t}}) < 1\right\}. \tag{15}$$

Since the channel gains between the mobile device and different computation resources over different channels are independent with each other, the probability that the input task received by m $(0 \leq m \leq M)$ computation resources successfully is given by

$$p_{\text{br}}^{\text{up}}(m, \xi_{\text{t}}) = \binom{M}{m} \left\{1 - p_{\text{br}}^{\text{out}}(\xi_{\text{t}})\right\}^{m} \left\{p_{\text{br}}^{\text{out}}(\xi_{\text{t}})\right\}^{(M-m)}. \tag{16}$$

If no computation resource can receive the task successfully, i.e., $m = 0$, the computation offloading is considered to be a failure, i.e., in outage state, with probability given by

$$p_{\text{br}}^{\text{up}}(0, \xi_{\text{t}}) = \left\{p_{\text{br}}^{\text{out}}(\xi_{\text{t}})\right\}^{M}. \tag{17}$$

For each offloading scenario, the operation in the upload phase is identical, while it might be different in computation and download phases, which are introduced next.

A. Classification of Computation Offloading Scenarios

Consider the condition that the whole task has been received by m $(1 \leq m \leq M)$ computation resources after the upload phase, whose set is denoted by $\mathcal{C}_{\text{ava}}$, where $\mathcal{C}_{\text{ava}} \subseteq \mathcal{C}$ and $|\mathcal{C}_{\text{ava}}| = m$. Thus, the task can be computed on at most m computation resources in parallel. Then, there are multiple offloading scenarios should be considered, which are different in the computation and download phases. Intuitively, the offloading scenarios can be classified into two major categories according to either centralized or distributed computation, which are summarized in Table II.

TABLE II
Two Computation Offloading Categories

Upload	Computation	Download
Point-to-multipoint	Centralized	Point-to-point
Point-to-multipoint	Distributed	Multiple access

Furthermore, the available computation capability of the computation resource c_i for computation offloading is given by $C(\xi_t) - q_i$, which can be used by the controller to design the subtask division method $\mathbf{x}$. Hence, we define the value $C(\xi_t) - q_i$ for $\mathcal{C}_i \in \mathcal{C}_{\text{ava}}$ as the *computation capability information* in the computation offloading. It should be noticed that the value of $C(\xi_{\text{c}})$ and ξ_{c} are given, while q_i is a random variable, which can be obtained through the feedback from the computation resource c_i to the controller. It might not be guaranteed in practical systems due to the constraint on signaling overhead. Thus, another factor that should be considered is whether the controller has computation capability information for all $c_i \in \mathcal{C}_{\text{ava}}$. We consider two aforementioned factors jointly and then four feasible scenarios are summarized in Table III and clarified as follows:

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCOMM.2019.2920348, IEEE Transactions on Communications

TABLE III
CLASSIFICATION OF FOUR TYPICAL SCENARIOS IN COMPUTATION OFFLOADING

Computation	Without computation capability information	With computation capability information
Centralized	Type 1	Type 2
Distributed	Type 4	Type 3

1) Type 1: One computation resource c_{i^*} is selected from $\mathcal{C}_{\rm ava}$ randomly to compute the whole task. In this scenario, we have $x_i = 1$ if $i = i^*$; otherwise $x_i = 0$.
2) Type 2: One computation resource c_{i^*} with the maximal available computation capability, i.e., correspondingly the minimal background task q_{i^*}, is selected from $\mathcal{C}_{\rm ava}$ to compute the whole task according to the given computation capability information. In this scenario, we have $x_i = 1$ if $i = i^*$; otherwise $x_i = 0$.
3) Type 3: The whole task is divided into m subtasks to each available computation resources in $\mathcal{C}_{\rm ava}$. Intuitively, the computation resource with high available computation capability should be allocated to a relatively high portion of the whole task. Then the subtasks can be computed in parallel on multiple computation resources. The task division method $\mathbf{x}$ can be obtained according to Algorithm 1 to minimize the outage probability of the offloading.
4) Type 4: The whole task is divided into m subtasks with identical size to each available computation resource in $\mathcal{C}_{\rm ava}$. In this scenario, we have $x_i = \frac{1}{m}$ if $c_i \in \mathcal{C}_{\rm ava}$; otherwise $x_i = 0$.

Next, we conduct the outage probability analysis with respect to the aforementioned scenarios.

B. Type 1: Centralized Computation without Computation Capability Information

In this scenario, the controller selects one computation resource c_{i^*} from available computation resource set $\mathcal{C}_{\rm ava}$ randomly. Hence the failure probability of computing is given by

$$p^{\rm comp}_{\rm Type1}(m, \xi_{\rm c}) = \Pr\left\{w + q_i > C(\xi_{\rm c})\right\}, \tag{18}$$

where q_{i^*} is replaced with q_i since q_i $(1 \le i \le M)$ are i.i.d. random variables.

In the download phase, the result of the task is transmitted from c_{i^*} back to the mobile device, which is the point-to-point transmissions over N channels, i.e., $\mathbf{a}_{i^*} = [1, \ldots, 1]$. Since $x_{i^*} = 1$, the failure probability of downloading is given by

$$p^{\rm down}_{\rm Type1}(\xi_{\rm t}) = \Pr\left\{R^{i^*}_{\rm d}(\mathbf{h}^{i^*}_{\rm d}, \mathbf{a}_{\mathbf{i^*}}, \xi_{\rm t}) < 1\right\}. \tag{19}$$

The outage events in the computation and download phases are independent of each other since they depend on parameters and random variables of computations and transmissions separately. Hence the outage probability during the computation and download phases can be approximated by

$$\begin{aligned} &p_{\rm Type1}(m, \xi_{\rm t}, \xi_{\rm c}) \\ &= p^{\rm comp}_{\rm Type1}(m, \xi_{\rm c}) + p^{\rm down}_{\rm Type1}(\xi_{\rm t}) - p^{\rm comp}_{\rm Type1}(m, \xi_{\rm c}) p^{\rm down}_{\rm Type1}(\xi_{\rm t}) \\ &\approx p^{\rm comp}_{\rm Type1}(m, \xi_{\rm c}) + p^{\rm down}_{\rm Type1}(\xi_{\rm t}), \end{aligned} \tag{20}$$

when $p^{\rm comp}_{\rm Type1}(m, \xi_{\rm c})$ and $p^{\rm down}_{\rm Type1}(\xi_{\rm t})$ are sufficiently small, i.e., $\xi_{\rm c}$ and $\xi_{\rm t}$ are sufficiently large. In practical systems, the approximation error in Eq. (20) is limited. This is simply due to the fact that the outage probability in any practical systems must be small enough, or otherwise the QoS cannot be guaranteed, resulting in this scenario is unpractical. Therefore, the neglected part in approximation $p^{\rm comp}_{\rm Type1}(m, \xi_{\rm c}) p^{\rm down}_{\rm Type1}(\xi_{\rm t})$ is much less than the overall outage probability of the systems, which can be neglected since the accuracy will not be reduced too much.

Then, From Eqs. (16), (17), and (20), the overall outage probability in the scenario of Type 1 can be approximated by

$$\begin{aligned} &p_{\rm Type1}(\xi_{\rm t}, \xi_{\rm c}) \approx p^{\rm up}_{\rm br}(0, \xi_{\rm t}) + \\ &\sum_{m=1}^{M} \left\{ p^{\rm up}_{\rm br}(m, \xi_{\rm t}) \left(p^{\rm comp}_{\rm Type1}(m, \xi_{\rm c}) + p^{\rm down}_{\rm Type1}(\xi_{\rm t}) \right) \right\}. \end{aligned} \tag{21}$$

Thus, we obtain the following result.

Theorem 1. *When $\xi_{\rm c}$ and $\xi_{\rm t}$ are sufficiently large, we have*

$$p_{\rm Type1}(\xi_{\rm t}, \xi_{\rm c}) \sim k_1 \xi_{\rm t}^{-N} + p^{\rm comp}_{\rm Type1}(M, \xi_{\rm c}), \tag{22}$$

where $k_1 \in \left[{N_0}^N 2^{N R_2}, {N_0}^N \left(2^{N R_1} M + 2^{N R_2} \right) \right]$ with $R_1 = \frac{l_{\rm in}}{T_1}$ and $R_2 = \frac{l_{\rm out}}{T_2}$.

Proof. See Appendix A. □

C. Type 2: Centralized Computation with Computation Capability Information

Given the computation capability information of all available computation resources in $\mathcal{C}_{\rm ava}$, the controller can select the computation resource c_{i^*} with minimal background tasks from $\mathcal{C}_{\rm ava}$. Hence the failure probability of computing is given by

$$\begin{aligned} p^{\rm comp}_{\rm Type2}(m, \xi_{\rm c}) &= \Pr\left\{ \max_{c_i \in \mathcal{C}_{\rm ava}} R^i_{\rm c}(w, q_i, \xi_{\rm c}) < x_i \right\} \\ &= \Pr\left\{ w + \min_{i=1,\ldots,m} q_i > C(\xi_{\rm c}) \right\}, \end{aligned} \tag{23}$$

which is different from that in the scenario of Type 2 given in Eq. (18). During the offloading, the operation in the computation phase in the scenarios of Type 1 and Type 2 are different, while that in the upload and download phases are identical. Thus, similar to the proof of Theorem 1, we can obtain the following result for the overall outage probability $p_{\rm Type2}(\xi_{\rm t}, \xi_{\rm c})$ in this scenario.

Theorem 2. *When $\xi_{\rm c}$ and $\xi_{\rm t}$ are sufficiently large, we have*

$$p_{\rm Type2}(\xi_{\rm t}, \xi_{\rm c}) \sim k_2 \xi_{\rm t}^{-N} + p^{\rm comp}_{\rm Type2}(M, \xi_{\rm c}), \tag{24}$$

where ${N_0}^N 2^{N R_2} \le k_2 \le {N_0}^N \left(2^{N R_1} M + 2^{N R_2} \right)$ with $R_1 = \frac{l_{\rm in}}{T_1}$ and $R_2 = \frac{l_{\rm out}}{T_2}$.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCOMM.2019.2920348, IEEE Transactions on Communications

D. Type 3: Distributed Computation with Computation Capability Information

Given the computation capability information of all computation resources in $\mathcal{C}_{\rm ava}$, the computation capability can be fully exploited by finding the optimal subtask division method $\mathbf{x}^*$ at the controller. In the ideal case, the task cannot be completed only if the sum of available computation capabilities of all computation resources in $\mathcal{C}_{\rm ava}$ cannot satisfy the computation requirement of the task. From Eqs. (5) and (14), the probability of the task cannot be completed during the computation phase in above ideal case is given by

$$p_{\rm Type3}^{\rm comp}(m,\xi_{\rm c}) = \Pr\left\{ w > \sum_{i=1}^{m} \max\{C(\xi_{\rm c}) - q_i, 0\} \right\}, \quad (25)$$

where $\max\{C(\xi_{\rm c}) - q_i, 0\}$ represents the available computation capability in computation resource $c_i \in \mathcal{C}_{\rm ava}$. Since the computation resource will be fully loaded when $C(\xi_{\rm c}) \le q_i$, the available computation capability of each computation resource cannot be negative. To obtain the optimal subtask division method $\mathbf{x}^*$ minimizing the overall outage probability of offloading, an algorithm with low computation complexity is proposed in Algorithm 1. Based on this, we have the following theorem.

Algorithm 1 Subtask Division Algorithm

1: initialize the sets $\mathcal{C}_{\rm ava}$ and $\mathbf{x} = \mathbf{0}$.
2: **repeat**
3: $m = |\mathcal{C}_{\rm ava}|$.
4: $q_{i\max} = \max\limits_{c_i \in \mathcal{C}_{\rm ava}} \{q_i\}$ and $i\max = \arg \max\limits_{c_i \in \mathcal{C}_{\rm ava}} \{q_i\}$.
5: if $C(\xi_{\rm c}) - q_{i\max} > \frac{w}{m}$, then $x_{i\max} = \frac{1}{m}$.
6: else $x_{i\max} = \frac{C(\xi_{\rm c}) - q_{i\max}}{w}$, $w = (1 - x_{i\max})w$ and delete $c_{i\max}$ from $\mathcal{C}_{\rm ava}$.
7: **until** $\mathcal{C}_{\rm ava} = \emptyset$.
8: output $\mathbf{x}^* = \mathbf{x}$, then stop.

Theorem 3. *By implementing the subtask division method $\mathbf{x}^*$ obtained from Algorithm 1, the overall outage probability of computation offloading can be minimized if $\sum\limits_{c_i \in \mathcal{C}_{\rm ava}} x_i = 1$. Otherwise if $\sum\limits_{c_i \in \mathcal{C}_{\rm ava}} x_i < 1$, the offloading will be in outage in the computation phase for any subtask division methods.*

Proof. See Appendix B. □

Then, in the download phase, the result of each subtask is transmitted from the corresponding computation resource in $\mathcal{C}_{\rm ava}$ back to the mobile device. From Eqs. (9) and (14), the failure probability of downloading subtask from c_i through channel s_j is given by

$$\begin{aligned} p_{\rm Type3}^{{\rm down},i}(x_i,\xi_{\rm t}) &= \Pr\left\{ \frac{T_2 \log(1 + |h_{\rm d}^{ij}|^2 \frac{\xi_{\rm t}}{N_0})}{l_{\rm out}} < x_i \right\} \\ &= \Pr\left\{ R_{\rm d}^i(\mathbf{h}_{\rm d}^i, [1,0,\ldots,0], \xi_{\rm t}) < x_i \right\}, \end{aligned} \quad (26)$$

because the channel gains $h_{\rm d}^{i1}$ and $h_{\rm d}^{ij}$, $j = 2,\ldots,N$, are i.i.d. random variables.

The download phase in this scenario is to conduct multiple access transmissions of m computation resources over N channels ($m \le M \le N$), and optimal channel allocation can be formulated as a matching problem in the random bipartite graph (RBG) [30] and then be solved by the bipartite graph matching. The RBG model can be constructed in the following steps.

First, all the vertices in one partition class are used to represent the available computation resources $c_i \in \mathcal{C}_{\rm ava}$, and the vertices in another partition class are used to represent all the channels $s_n \in \mathcal{S}$. Only if the result of the subtask in $c_i \in \mathcal{C}_{\rm ava}$ can be downloaded through channel $s_n \in \mathcal{S}$ successfully can we add an edge between the corresponding vertices in the RBG sample. A sample of this RBG model with $m = 4$ and $N = 6$ is shown in Fig. 3. By applying the RBG model, the optimal channel allocation can be obtained according to $\mathbf{R^2HK}$ Algorithm with the complexity of $O(N^{2.5})$ [30].

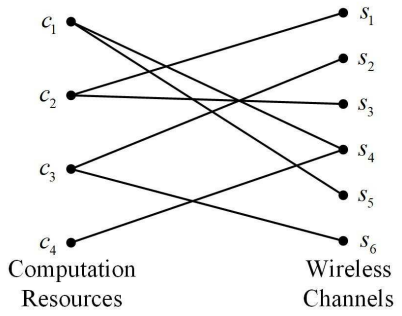


Fig. 3. A sample of RBG model with 4 available computation resources and 6 wireless channels.

Let $p_{\rm type3}^{\rm down}(m,\xi_{\rm t})$ denote the failure probability of downloading in the scenario of Type 3, which is the probability that not all results can be downloaded successfully with the subtask division method $\mathbf{x}^*$ according to Algorithm 1. The following theorem can be obtained.

Theorem 4. *For sufficiently large $\xi_{\rm t}$, the failure probability of downloading $p_{\rm type3}^{\rm down}(m,\xi_{\rm t})$ under channel allocation scheme based on $\mathbf{R^2HK}$ algorithm is given by*

$$\begin{aligned} p_{\rm type3}^{\rm down}(m,\xi_{\rm t}) &= \sum_{c_i \in \mathcal{C}_{\rm ava}} \left\{ p_{\rm Type3}^{{\rm down},1}(x_i,\xi_{\rm t}) \right\}^N + a_0(m) \cdot \\ & N \prod_{c_i \in \mathcal{C}_{\rm ava}} p_{\rm Type3}^{{\rm down},1}(x_i,\xi_{\rm t}) + o\left(\sum_{c_i \in \mathcal{C}_{\rm ava}} \left\{ p_{\rm Type3}^{{\rm down},1}(x_i,\xi_{\rm t}) \right\}^N \right), \end{aligned} \quad (27)$$

where

$$a_0(m) = \begin{cases} 0, & m < N \\ 1, & m = N. \end{cases} \quad (28)$$

Proof. See Appendix C. □

From this Theorem, we can easily obtain the following result.

Corollary 1. *For sufficiently large $x_{\rm t}$, $p_{\rm type3}^{\rm down}(m,\xi_{\rm t})$ can be approximated by*

$$p_{\rm type3}^{\rm down}(m,\xi_{\rm t}) \approx k_{\rm d} {\xi_{\rm t}}^{-N}, \quad (29)$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCOMM.2019.2920348, IEEE Transactions on Communications

where $k_{\rm d} \in \left[m2^{x_{\min}R_2N},(m+N)2^{x_{\max}R_2N}\right]\cdot {N_0}^N\xi_{\rm t}^{-N}$, $x_{\max}=\max\limits_{c_i\in\mathcal{C}_{\rm ava}} x_i$, *and* $x_{\min}=\min\limits_{c_i\in\mathcal{C}_{\rm ava}} x_i$.

Proof. See Appendix D. □

When $\xi_{\rm c}$ and $\xi_{\rm t}$ are sufficiently large, the overall outage probability in this scenario can be approximated by

$$p_{\rm Type3}(\xi_{\rm t},\xi_{\rm c})\approx p_{\rm br}^{\rm up}(0,\xi_{\rm t}) + \sum_{m=1}^{M}\left\{p_{\rm br}^{\rm up}(m,\xi_{\rm t})\left(p_{\rm Type3}^{\rm comp}(m,\xi_{\rm c})+p_{\rm Type3}^{\rm down}(m,\xi_{\rm t})\right)\right\}, \tag{30}$$

and we can obtain the following theorem.

Theorem 5. *When $\xi_{\rm c}$ and $\xi_{\rm t}$ are sufficiently large, we have*

$$p_{\rm Type3}(\xi_{\rm t},\xi_{\rm c})\sim k_3\xi_{\rm t}^{-N}+p_{\rm Type3}^{\rm comp}(M,\xi_{\rm c}), \tag{31}$$

where $k_3\in\left[k_{\rm d},{N_0}^N2^{NR_1}M+k_{\rm d}\right]$ with $R_1=\frac{l_{\rm in}}{T_1}$ and $R_2=\frac{l_{\rm out}}{T_2}$.

E. Type 4: Distributed Computation without Computation Capability Information

Without known computation capability information of computation resources $c_i\in\mathcal{C}_{\rm ava}$ in the controller, a simple but reasonable method for the controller is to divide the whole task into subtasks with identical size and then the subtasks are computed in each computation resource in the available computation resource set, i.e., $x_i=\frac{1}{m}$ for $c_i\in\mathcal{C}_{\rm ava}$. From Eqs. (5) and (14), the failure probability of computing is given by

$$\begin{aligned}p_{\rm Type4}^{\rm comp}(m,\xi_{\rm c})&=\Pr\left\{\min_{c_i\in\mathcal{C}_{\rm ava}} R_i^{\rm c}(w,q_i,\xi_{\rm c})<x_i\right\}\\&=\Pr\left\{\frac{w}{m}+\max_{i=1,\ldots,m}q_i>C(\xi_{\rm c})\right\}.\end{aligned} \tag{32}$$

In contrast to this scenario, the scenario of Type 3 without the constraint of equal task division is more general. Thus, the task division method in the scenario of Type 4 can be viewed as one case of that in the scenario of Type 3. By substituting $x_i=\frac{1}{m}$ for $c_i\in\mathcal{C}_{\rm ava}$ in Eq. (26), the failure probability of downloading subtask from c_i through channel s_j is given by

$$\begin{aligned}p_{\rm Type4}^{\rm down}(\xi_{\rm t})&=p_{\rm Type3}^{{\rm down},i}(\frac{1}{m},\xi_{\rm t})\\&=\Pr\left\{R_i^{\rm d}(\mathbf{h}_i^{\rm d},[1,0,\ldots,0],\xi_{\rm t})<\frac{1}{m}\right\}.\end{aligned} \tag{33}$$

Then, from Eq. (27) in Theorem 4, the failure probability of downloading is given by

$$p_{\rm type4}^{\rm down}(m,\xi_{\rm t})=b(m)\left\{p_{\rm Type4}^{\rm down}(\xi_{\rm t})\right\}^N+o\left(p_{\rm Type4}^{\rm down}(\xi_{\rm t})^N\right), \tag{34}$$

where $b(m)=a(m)+1$. When $\xi_{\rm c}$ and $\xi_{\rm t}$ are sufficiently large, the overall outage probability in this scenario can be approximated by

$$p_{\rm Type4}(\xi_{\rm t},\xi_{\rm c})\approx p_{\rm br}^{\rm up}(0,\xi_{\rm t}) + \sum_{m=1}^{M}\left\{p_{\rm br}^{\rm up}(m,\xi_{\rm t})\left(p_{\rm Type4}^{\rm comp}(m,\xi_{\rm c})+p_{\rm Type4}^{\rm down}(m,\xi_{\rm t})\right)\right\}, \tag{35}$$

and the following theorem can be obtained.

Theorem 6. *When $\xi_{\rm c}$ and $\xi_{\rm t}$ are sufficiently large, we have*

$$p_{\rm Type4}(\xi_{\rm t},\xi_{\rm c})\sim k_4\xi_{\rm t}^{-N}+p_{\rm Type4}^{\rm comp}(M,\xi_{\rm c}), \tag{36}$$

where $k_4\in\left[b(M)M{N_0}^N\lambda_2,b(M)M{N_0}^N\left(\lambda_1+\lambda_2\right)\right]$ with $\lambda_1=2^{\frac{Nl_{\rm in}}{T_1}}$ and $\lambda_2=2^{\frac{Nl_{\rm in}}{T_2}}$.

V. Outage Bottleneck Analysis

According to Theorems 1, 2, 5, and 6, the outage probabilities with high power consumption of computation and transmission in all considered scenarios can be divided into two parts, which only depend on transmission parameters N and $\xi_{\rm t}$, and computation parameters M and $\xi_{\rm c}$, respectively. Thus, we define these two partitions as the computation and transmission outage probability partitions. For the sake of simplicity, we set the constant coefficients in transmission outage probability in Theorems 1, 2, 5 and 6, i.e., k_1,k_2,k_3 and k_4, to 1. Therefore, we have the following definition.

Definition 1. The computation and transmission outage probability partitions of four considered scenarios are defined by

$$\begin{cases}p_{\rm out}^{{\rm c},k}(M,\xi_{\rm c})=p_{{\rm Type}k}^{\rm comp}(M,\xi_{\rm c}),\\ p_{\rm out}^{{\rm t},k}(N,\xi_t)=\xi_{\rm t}^{-N},\end{cases} \tag{37}$$

where $k\in\{1,2,3,4\}$ that correspond to Type 1, 2, 3, and 4 scenarios.

Then, for given M and N, the power consumptions with $p_{\rm out}^{{\rm c},k}(M,\xi_c)=p_{\rm out}^{{\rm t},k}(N,\xi_t)=\varepsilon$ can be rewritten as

$$\begin{cases}\xi_{\rm c}(\varepsilon)={p_{\rm out}^{{\rm c},k}}^{-1}(\varepsilon)=f_k(\varepsilon),\\ \xi_{\rm t}(\varepsilon)={p_{\rm out}^{{\rm t},k}}^{-1}(\varepsilon)=g_k(\varepsilon),\end{cases} \tag{38}$$

where $p^{-1}(x)$ is the inverse function of $p(x)$. The major part of total power consumption as $\varepsilon\to 0$ is considered as the *outage bottleneck* of the mobile computation offloading system, which is the major limiting factor to the decrease of outage probability of the system by increasing total power consumption. Then, according to Eq. (38), the *outage bottleneck* of the computation offloading can be found by comparing power consumption ratio

$$h_k(\varepsilon)=\frac{\xi_{\rm c}(\varepsilon)}{\xi_{\rm t}(\varepsilon)}=\frac{f_k(\varepsilon)}{g_k(\varepsilon)} \tag{39}$$

with $\varepsilon\to 0$ and $k\in\{1,2,3,4\}$, which is summarized in the following definitions.

Definition 2. Define[1]

$$\Xi_k=\lim_{\varepsilon\to 0}h_k(\varepsilon)=\lim_{\varepsilon\to 0}\frac{f_k(\varepsilon)}{g_k(\varepsilon)}=\lim_{\xi_{\rm c}\to\infty}\frac{\xi_{\rm c}}{g_k({f_k}^{-1}(\xi_{\rm c}))}, \tag{40}$$

where $k\in\{1,2,3,4\}$. If $\Xi_k=0$ or ∞, the *outage bottleneck* of corresponding offloading scenario is either transmission or computation capability, respectively. Otherwise, if $\Xi_k\in(0,\infty)$, the *outage bottleneck* is both transmission and computation capability.

[1]In this paper, we do not consider the scenario that the limitation $\lim\limits_{\varepsilon\to 0}h_k(\varepsilon)$ does no exist.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCOMM.2019.2920348, IEEE Transactions on Communications

The rest of this paper focuses on the analysis of the features and acquisition method of Ξ_k to obtain *outage bottleneck*, which are summarized in the following theorems.

Theorem 7. *Ξ_k defined in Eq. (40) can be rewritten as*

$$\Xi_k = W \lim_{x\to\infty} l_k(x), \tag{41}$$

where $k \in \{1,2,4\}$, W is a constant satisfying $W = \frac{1}{{k_c}^3\{3N\}^{\frac{1}{N}}} \in [0,\infty]$, and $l_k(x)$ is given by

$$l_k(x) = \left\{\frac{f_{u_k}(x)}{x^{-(3N+1)}}\right\}^{\frac{1}{N}} \tag{42}$$

with $f_{u_k}(t)$ being the probability density function of u_k, which is given by

$$u_k = \begin{cases} w + q_i, & k = 1; \\ w + \min\limits_{i=1,\ldots,M} q_i, & k = 2; \\ \dfrac{w}{M} + \max\limits_{i=1,\ldots,M} q_i, & k = 4. \end{cases} \tag{43}$$

Proof. See Appendix E. □

According to Theorem 7, the *outage bottleneck* in the scenarios of Type 1, 2, and 4 can be obtained by analyzing $\lim\limits_{x\to\infty} l_k(x)$. However, it is sometimes difficult to obtain explicit expression of $l_k(x)$. To solve this problem and obtain the *outage bottleneck* more efficiently, we first analyze distributions of w, q_i and u_k.

Definition 3. Define

$$\begin{cases} \varphi_1 = \lim\limits_{x\to\infty} x^{3N+1} f_w(x); \\ \phi_1 = \lim\limits_{x\to\infty} x^{3N+1} f_q(x), \end{cases} \tag{44}$$

where $f_w(x)$ and $f_q(x)$ are the probability density functions of w and q_i, respectively.

Theorem 8. *In the scenarios of Type 1 and Type 4, if $\max\{\varphi_1, \phi_1\} = 0$ or ∞, the outage bottleneck of this system is either transmission or computation capability, respectively. Otherwise, if $\max\{\varphi_1, \phi_1\} \in (0,\infty)$, the outage bottleneck is both transmission and computation capability.*

Proof. See Appendix F. □

Similar to Definition 3 and Theorem 8 for the scenarios of Type 1 and Type 4, we have the following definitions for the scenarios of Type 2 and Type 3.

Definition 4. Define

$$\begin{cases} \varphi_2 = \lim\limits_{x\to\infty} x^{3N+1} f_w(x); \\ \phi_2 = \lim\limits_{x\to\infty} x^{\frac{3N}{M}+1} f_q(x), \end{cases} \tag{45}$$

where $f_w(x)$ and $f_q(x)$ are the probability density functions of w and q_i, respectively.

Theorem 9. *In the scenarios of Type 2 and 3, if $\max\{\varphi_2, \phi_2\} = 0$ or ∞, the outage bottleneck of this system is either transmission or computation capability, respectively. Otherwise, if $\max\{\varphi_2, \phi_2\} \in (0,\infty)$, the outage bottleneck is both transmission and computation capability.*

Proof. For the scenario of Type 2, the proof is similar to that for the scenario of Type 1 in Appendix F. In addition, for the scenario of Type 3, the proof is given in Appendix G. □

Remark 1. According to [31], the number of required CPU cycles w depends on the input data size l_{in}. For a given input data size l_{in}, the number of required CPU cycles can be derived as

$$w = L l_{\text{in}}, \tag{46}$$

where L indicates the number of CPU cycles per bit, which can be modeled by a Gamma distribution with shape parameter α and scale parameter β in [31]. In this case, we have

$$f_w(x) = \frac{1}{\beta\Gamma(\alpha) l_{\text{in}}} \left(\frac{x}{\beta l_{\text{in}}}\right)^{\alpha-1} e^{-\frac{x}{\beta l_{\text{in}}}}, x > 0, \tag{47}$$

Then, φ_1 and φ_2 can be calculated as

$$\varphi_1 = \varphi_2 = \lim_{t\to\infty} x^{(3N+1)} f_w(x) = 0, \quad \forall N \geq 1. \tag{48}$$

According to Theorem 8 and Theorem 9, the *outage bottleneck* of four considered offloading scenarios only depends on ϕ_1 or ϕ_2. If q_i also follows a Gamma distribution, i.e., $\phi_1 = \phi_2 = 0$, the *outage bottlenecks* of four offloading scenarios are transmission capability.

Remark 2. If w and q_i are light-tailed distributions as defined in [32], we have $\varphi_1 = \phi_1 = 0$ and $\varphi_2 = \phi_2 = 0$, and the *outage bottlenecks* of four considered offloading scenarios are transmission capability. However, if w and q_i are heavy-tailed distribution as defined in [32], the *outage bottlenecks* can also be obtained according to Theorem 8 and Theorem 9.

VI. Numerical Results

In this section, we validate the theoretical analysis via Monte-Carlo simulation results. Throughout this section, we set $k_c = 1$, the additive white Gaussian noise $N_0 = 1$, data size of the input task $l_{\text{in}} = 5$, and data size of the output result $l_{\text{out}} = 0.5$. The lengths of the time slot, the upload phase, and the download phase are assumed to be normalized, i.e., $T_1 = 1$, $T_2 = 1$, and $T_{\text{s}} = 2$. In addition, the channel gains h_{u}^{ij} and h_{d}^{ij} are assumed to follow independent and identical distribution $\mathcal{CN}(0,1)$. Moreover, the distributions of computation requirements w and q_i are given by

1) Case 1: w and q_i follow the identical independent Gamma distribution with $\alpha = \beta = 1$, i.e., the exponential distribution with parameter $\lambda = 1$.
2) Case 2: w and q_i follow the independent power-law distribution, whose probability density functions are $f_w(x) = \frac{10}{x^{11}}$ and $f_q(x) = \frac{5}{x^6}$ for $x \geq 1$, respectively.

The controller collects the above information and schedules the offloading based on the proposed approach. Then, not only the overall outage probability of offloading, but also the computation and transmission outage probability partitions defined in Eq. (37) can be obtained.

For two cases, the φ_1, ϕ_1, φ_2, and ϕ_2 defined in Definitions 3 and 4 are summarized in Table IV. According to Theorem 8 and Theorem 9, we can further obtain the *outage bottleneck* in four considered scenarios with case 1 and case 2, which is summarized in Table V.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCOMM.2019.2920348, IEEE Transactions on Communications

TABLE IV
$\varphi_{1/2}$ AND $\phi_{1/2}$ IN TWO CASES

f_w, f_q	Case 1: exponential	Case 2: power-law
φ_1	0	0 ($N \leq 3$) and ∞ ($N \geq 4$)
φ_2	0	0 ($N \leq 3$) and ∞ ($N \geq 4$)
ϕ_1	0	0 ($N \leq 1$) and ∞ ($N \geq 2$)
ϕ_2	0	0 ($3N < 5M$) and ∞ ($3N > 5M$)

TABLE V
OUTAGE BOTTLENECKS IN TWO CASES

Scenario	Case 1: exponential	Case 2: power-law
Type 1	Transmission	Transmission ($N \leq 1$) and computation ($N \geq 2$)
Type 2	Transmission	Transmission ($N \leq 3$ and $3N < 5M$) and computation (in others)
Type 3	Transmission	Transmission ($N \leq 3$ and $3N < 5M$) and computation (in others)
Type 4	Transmission	Transmission ($N \leq 1$) and computation ($N \geq 2$)

A. Overall outage probabilities in different scenarios

The overall outage probabilities in four considered scenarios are given in Figs. 4 and 5, versus different power consumptions of computation and transmission, respectively. With the increase of the power consumptions of computation and transmission, the overall outage probabilities decrease in all considered scenarios. In addition, for the scenarios of centralized computation, the scenario of Type 2 always has a better outage performance than the scenario of Type 1. Similarly, for the scenarios of distributed computation, the overall outage probability in the scenario of Type 3 always below that in the scenario of Type 4. Hence it shows that the considered system can achieve a better outage performance with the help of the computation information.

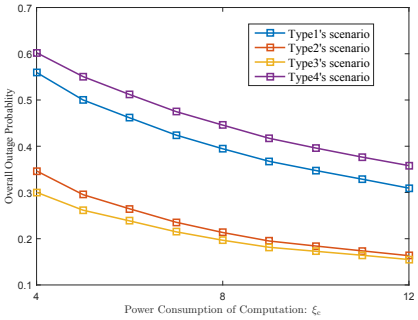


Fig. 4. Overall outage probabilities with $\xi_t = 15$.

B. One simple system with $M = 1$

We consider a simple scenario that $M = 1$ firstly. Since there is only one computation resource in the system, all scenarios will degrade into the scenario of Type 1.

Fig. 6 indicates the power consumption ratio $\frac{\xi_c}{\xi_t}$ defined in Eq. (39) with the assurance that $p_k^t(\xi_t) = p_k^c(\xi_c)$, $k \in \{1, 2, 3, 4\}$, with the increase of ξ_c. By increasing the number of channels N from 1 to 4, the power consumption ratio of case 1 always tends to 0, while that of case 2 tends to ∞ when $N \geq 2$. It shows that the *outage bottleneck* for case 1 when $1 \leq N \leq 4$ and for case 2 when $N = 1$ is transmission capability, while for case 2 with $N \geq 2$ is computation capability, where theoretical results and the simulation results match perfectly well.

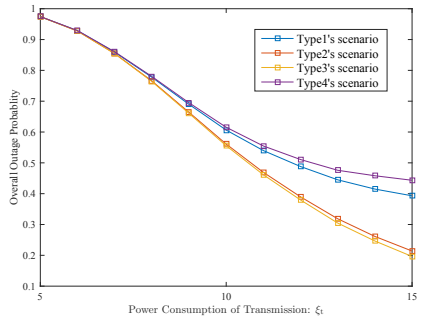


Fig. 5. Overall outage probabilities with $\xi_c = 12$.

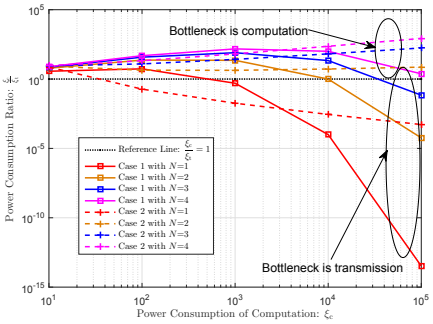


Fig. 6. The power consumption ratio with $M = 1$, i.e., Type 1's scenario.

Next we consider a system with multiple wireless channels and computation resources.

C. Centralized Computation with and without Computation Information

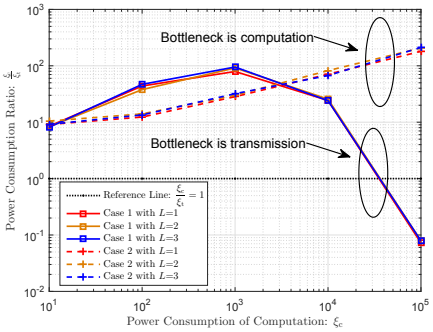


Fig. 7. The power consumption ratio in Type 1's scenario.

Consider the scenarios of centralized computation and the number of channels N is 3, where the scenario of Type 1 has

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCOMM.2019.2920348, IEEE Transactions on Communications

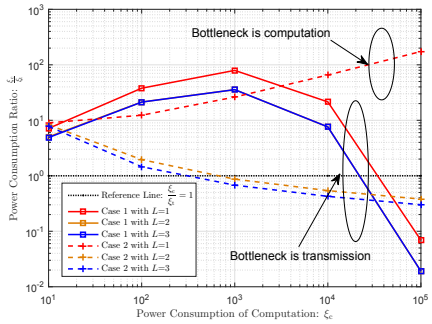


Fig. 8. The power consumption ratio in Type 2's scenario.

been presented in the previous [2] and that of Type 2 has been presented in Fig. 7. It shows that the power consumption ratio $\frac{\xi_c}{\xi_t}$ defined in Eq. (40) with the increase of ξ_c in the scenarios of Type 1 and Type 2. As shown in Fig. 7, by increasing the number of computation resources M from 1 to 3, the power consumption ratios in the scenario of Type 1 always tends to 0 for case 1, and ∞ for case 2. Moreover, Fig. 8 shows that, in the scenario of Type 2, the power consumption ratio $\frac{\xi_c}{\xi_t}$ always tends to 0 for case 1, while that of case 2 tends to 0 when $M \geq 2$ and to ∞ when $M = 1$ with the increase of ξ_c. Thus, with the known of computation capability information, the *outage bottleneck* in the scenario of Type 2 can be transformed from computation to transmission by increasing the number of computation resources. The numerical results in Figs. 7 and 8 match well with the theoretical results in Theorem 8 and Theorem 9.

D. Distributed Computation with and without Computation Information

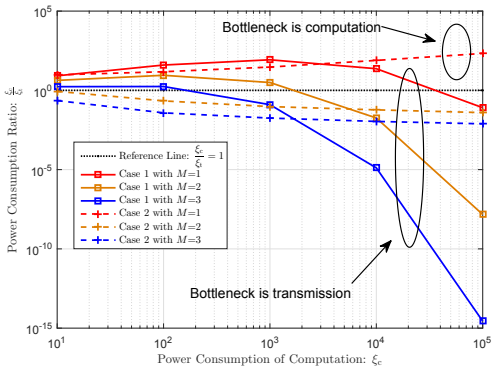


Fig. 9. The power consumption ratio in Type 3's scenario.

Consider the two scenarios of distributed computation and the number of channels N is 3. Figs. 9 and 10 show the power consumption ratio $\frac{\xi_c}{\xi_t}$ defined in Eq. (40) with the increase of ξ_c in the scenarios of Type 3 and Type 4. As shown in Fig. 9, by increasing the number of computation resources M from 1 to 3, the power consumption ratio of case 1 always tends to 0, while that of case 2 with $M \leq 1$ tends to ∞ with the increase of ξ_c in the scenario of Type 3. In addition, Fig. 10 shows

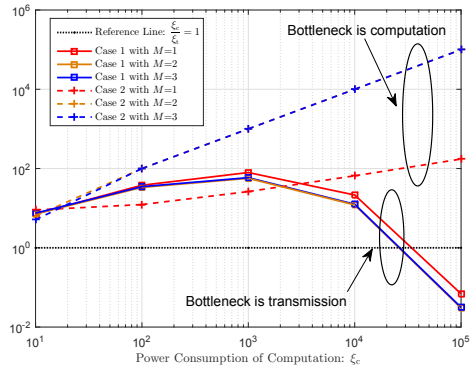


Fig. 10. The power consumption ratio in Type 4's scenario.

that the power consumption ratios of case 1 and case 2 in the scenario of Type 4 always tend to 0 and ∞, respectively. With the known of computation capability information, the *outage bottleneck* in the scenario of Type 3 can be transformed from computation to transmission by increasing the number of computation resources. The numerical results in Figs. 9 and 10 also match with the theoretical results in Theorem 8 and Theorem 9.

VII. Conclusions

In this paper, we have developed a unified framework to analyze the outage probability and *outage bottleneck* for mobile computation offloading systems. Offloading scenarios with different task division and channel allocation methods are considered. In each considered scenario, the overall outage probability can be divided into two parts corresponding to transmission and computation. The *outage bottleneck* completely determined by the power consumption ratio with high power consumption. By analyzing the computation complexities of tasks and bounding the outage probability, we have discovered the conditions for *outage bottleneck*. The theoretical results can be used to evaluate the outage performance and bottleneck for existing and future mobile computation offloading systems and improve outage performance in an efficient way. Important future topics include the extension to other system models, e.g., with local computing, the implementation of the distributed control, scheduling that takes into account the order of tasks and more influencing factors, as well as, practical bottleneck evaluation based on real data.

Appendix A
Proof of Theorem 1

Proof. According to Eqs. (15) and (19), we have

$$\begin{cases} p_{\mathrm{br}}^{\mathrm{out}}(\xi_{\mathrm{t}}) = \left(1 - e^{-2^{R_1}\frac{N_0}{\xi_{\mathrm{t}}}}\right)^N; \\ p_{\mathrm{Type1}}^{\mathrm{down}}(\xi_{\mathrm{t}}) = \left(1 - e^{-2^{R_2}\frac{N_0}{\xi_{\mathrm{t}}}}\right)^N, \end{cases} \tag{49}$$

When ξ_{c} and ξ_{t} are sufficiently large, Eq. (49) can be rewritten as

$$p_{\mathrm{br}}^{\mathrm{out}}(\xi_{\mathrm{t}}) \approx \left(\frac{2^{R_1}N_0}{\xi_{\mathrm{t}}}\right)^N \quad \text{and} \quad p_{\mathrm{Type1}}^{\mathrm{down}}(\xi_{\mathrm{t}}) \approx \left(\frac{2^{R_2}N_0}{\xi_{\mathrm{t}}}\right)^N. \tag{50}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCOMM.2019.2920348, IEEE Transactions on Communications

Moreover, we have

$$p_{\text{Type1}}^{\text{comp}}(m,\xi_{\text{c}}) \in [0,1], \forall m = 1,\ldots,M. \tag{51}$$

Thus substituting Eqs. (50) and (51) into Eq. (21), we prove the theorem in Eq. (52) as follows:

□

APPENDIX B
PROOF OF THEOREM 3

Proof. According to Eqs. (8) and (26), when ξ_{t} is sufficiently large, we have

$$p_{\text{Type3}}^{\text{down},i}(x_i,\xi_{\text{t}}) \approx 2^{x_i R_2} N_0 {\xi_{\text{t}}}^{-1}. \tag{53}$$

Then, by taking Eq. (53) into Eq. (27), we have

$$\begin{aligned} & p_{\text{type3}}^{\text{down}}(m,\xi_{\text{t}}) \\ \approx & \left(\frac{N_0}{\xi_{\text{t}}}\right)^N \sum_{c_i \in \mathcal{C}_{\text{ava}}} 2^{N x_i R_2} + \left(\frac{N_0}{\xi_{\text{t}}}\right)^m a_0(m) N \prod_{c_i \in \mathcal{C}_{\text{ava}}} 2^{x_i R_2} \\ = & \left(\frac{N_0}{\xi_{\text{t}}}\right)^N \sum_{c_i \in \mathcal{C}_{\text{ava}}} \left(2^{N R_2}\right)^{x_i} + \left(\frac{N_0}{\xi_{\text{t}}}\right)^m a_0(m) N 2^{R_2}, \end{aligned} \tag{54}$$

where $\sum_{i=1}^{m} x_i = 1$. Therefore, minimizing $p_{\text{type3}}^{\text{down}}(m,\xi_{\text{t}})$ is equivalent to minimizing

$$\sum_{c_i \in \mathcal{C}_{\text{ava}}} \left(2^{N R_2}\right)^{x_i} = \sum_{c_i \in \mathcal{C}_{\text{ava}}} \alpha^{x_i}, \tag{55}$$

where $\alpha = 2^{N R_2} > 1$. Then, by considering a case that $x_1 = \beta_1$ and $x_2 = \beta_2$ with $\beta_1 < \beta_2$, we have

$$(\alpha^{\beta_1} + \alpha^{\beta_2}) - 2\alpha^{\frac{\beta_1+\beta_2}{2}} > 0, \tag{56}$$

according to *Basic Inequality* that $a + b > 2\sqrt{ab}$ with $a, b > 0$. For the purpose of minimizing the value in Eq. (55), $(x_1, x_2) = (\frac{\beta_1+\beta_2}{2}, \frac{\beta_1+\beta_2}{2})$ is better than (β_1, β_2) when $m = 2$. According to the above proof by contradiction, the optimal $\mathbf{x}$ minimizing Eq. (55) is $x_i = \frac{1}{m}$ for $c_i \in \mathcal{C}_{\text{ava}}$; otherwise $x_i = 0$. However, according to Eq. (14), to avoid outage in the computation phase, the condition $x_i \leq \frac{\max\{C(\xi_{\text{c}}) - q_i, 0\}}{w}$ is required for all $c_i \in \mathcal{C}_{\text{ava}}$, which might not always hold.

Furthermore, we should allocate the subtask to the computation resource with $q_{i\max} = \max_{c_i \in \mathcal{C}_{\text{ava}}} q_i$ firstly. The fraction of task $x_{i\max}$ in $c_{i\max}$ is the lower bound among that of all resources in $\mathcal{C}_{\text{ava}}$. From Eqs. (5) and (14), we have the upper bound of $x_{i\max}$ is $\frac{C(\xi_{\text{c}}) - q_{i\max}}{w}$, which is the fraction that $c_{i\max}$ can support while guaranteeing no outage during the computation phase. Similar to the contradiction in Eq. (56), we could prove that $(x_1, x_2) = (\beta - \varepsilon, \varepsilon)$ is better than $(\beta, 0)$, where ε is a small constant. Thus, we set $x_{i\max} = \min\{\frac{1}{m}, \frac{C(\xi_{\text{c}}) - q_{i\max}}{w}\}$ to minimize $p_{\text{type3}}^{\text{down}}(m,\xi_{\text{t}})$ while avoid outage in the computation phase if at all possible. After this operation, the unallocated fraction of the whole task is updated to $1 - x_{i\max}$ and the set of computation resources waiting to allocate the subtasks is $\mathcal{C}_{\text{ava}} \backslash c_{i\max}$. Then we repeat the above operation until all resources have already be assigned to their subtasks, i.e., $\mathcal{C}_{\text{ava}} = \emptyset$. The above operations are summarized in Algorithm 1.

□

APPENDIX C
PROOF OF THEOREM 4

Proof. If $m < N$, according to Lemma 6 in [30], there is one subtask that cannot be downloaded successfully only when the subtask cannot be downloaded through any channel in $\mathcal{S}$. That is, this subtask is an isolated vertex in the RBG sample.

This event can occur in the subtask in any computation resources $\mathcal{C}_{\text{ava}}$. If $m = N$, according to Lemma 6 in [30], there is one subtask that cannot be downloaded successfully only when the subtask cannot be downloaded through any channel in $\mathcal{S}$ or all subtasks cannot be downloaded through one channel in $\mathcal{S}$. That is, this subtask or one channel is an isolated vertex in the RBG sample. This event can occur in the subtask in any computation resources $\mathcal{C}_{\text{ava}}$ and any channel in $\mathcal{S}$. Thus, similar to the proof of Theorem 1 in [30], we obtain the proof of the theorem. □

APPENDIX D
PROOF OF COROLLARY 1

Proof. According to Eq. (53), for given ξ_{t}, we can find that $p_{\text{Type3}}^{\text{down},1}(x_i,\xi_{\text{t}})$ is monotonously increasing in the fraction of the whole task x_i. Therefore, $p_{\text{Type3}}^{\text{down},1}(x_i,\xi_{\text{t}})$ can be bounded by

$$p_{\text{Type3}}^{\text{down},i}(x_{\min},\xi_{\text{t}}) \leq p_{\text{Type3}}^{\text{down},i}(x_i,\xi_{\text{t}}) \leq p_{\text{Type3}}^{\text{down},i}(x_{\max},\xi_{\text{t}}). \tag{57}$$

Then, by taking Eq. (53) in Eq. (27), we have

$$\begin{aligned} & p_{\text{Type3}}^{\text{down}}(m,\xi_{\text{t}}) \\ \approx & \sum_{i=1}^{m} \left(2^{x_i R_2} \frac{N_0}{\xi_{\text{t}}}\right)^N + a_0(m) N \prod_{i=1}^{m} \left(2^{x_i R_2} \frac{N_0}{\xi_{\text{t}}}\right). \end{aligned} \tag{58}$$

Hence, by taking Eq. (58) into Eq. (57), we have

$$p_{\text{Type3}}^{\text{down}}(m,\xi_{\text{t}}) \geq m 2^{x_{\min} R_2 N} {N_0}^N \xi_{\text{t}}^{-N}, \tag{59}$$

and

$$p_{\text{Type3}}^{\text{down}}(m,\xi_{\text{t}}) \leq (m+N) 2^{x_{\max} R_2 N} {N_0}^N \xi_{\text{t}}^{-N}. \tag{60}$$

□

APPENDIX E
PROOF OF THEOREM 7

Proof. According to Eqs. (18), (23) and (32), we have

$$g_k({f_k}^{-1}(\xi_{\text{c}})) = \frac{1}{\Pr\{u_k > k_{\text{c}} x^{\frac{1}{3}}\}^{\frac{1}{N}}}. \tag{61}$$

For any $k \in \{1, 2, 4\}$, by taking Eq. (61) into Eq. (40) and using L'Hôpital's rule, we have

$$\begin{aligned} \Xi_k = \lim_{x\to 0} h_k(x) &= \lim_{x\to\infty} \frac{x}{g_k({f_k}^{-1}(x))} \\ &= \lim_{x\to\infty} \left\{\frac{\Pr\{u_k > k_c x^{\frac{1}{3}}\}}{x^{-N}}\right\}^{\frac{1}{N}} \\ &= \lim_{x\to\infty} \frac{1}{{k_c}^3} \left\{\frac{\Pr\{u_k > x\}}{x^{-3N}}\right\}^{\frac{1}{N}} \\ &= \lim_{x\to\infty} \frac{1}{{k_c}^3} \left\{\frac{f_{u_k}(x)}{3N x^{-(3N+1)}}\right\}^{\frac{1}{N}} \\ &= W \lim_{x\to\infty} l_k(x). \end{aligned} \tag{62}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCOMM.2019.2920348, IEEE Transactions on Communications

$$
\begin{aligned}
&p_{\mathrm{Type1}}(\xi_{\mathrm{t}},\xi_{\mathrm{c}})\\
&\sim \left(\frac{2^{R_1}N_0}{\xi_{\mathrm{t}}}\right)^{NL}+\sum_{m=1}^{M}\left\{\binom{M}{m}\left\{1-\left(\frac{2^{R_1}N_0}{\xi_{\mathrm{t}}}\right)^{N}\right\}^{m}\left(\frac{2^{R_1}N_0}{\xi_{\mathrm{t}}}\right)^{N(M-m)}\left\{p_{\mathrm{Type1}}^{\mathrm{comp}}(m,\xi_{\mathrm{t}})+\left(\frac{2^{R_2}N_0}{\xi_{\mathrm{t}}}\right)^{N}\right\}\right\}\\
&\sim \sum_{m=1}^{M}\left\{\binom{M}{m}\left(\frac{2^{R_1}N_0}{\xi_{\mathrm{t}}}\right)^{N(M-m)}\left\{p_{\mathrm{Type1}}^{\mathrm{comp}}(m,\xi_{\mathrm{t}})+\left(\frac{2^{R_2}N_0}{\xi_{\mathrm{t}}}\right)^{N}\right\}\right\} \qquad (52)\\
&\sim p_{\mathrm{Type1}}^{\mathrm{comp}}(M,\xi_{\mathrm{t}})+\left(\frac{2^{R_2}N_0}{\xi_{\mathrm{t}}}\right)^{N}+M\left(\frac{2^{R_1}N_0}{\xi_{\mathrm{t}}}\right)^{N}p_{\mathrm{Type1}}^{\mathrm{comp}}(M-1,\xi_{\mathrm{t}})\\
&\in\left[p_{\mathrm{Type1}}^{\mathrm{comp}}(M,\xi_{\mathrm{t}})+\left({N_0}^{N}2^{NR_2}\right)\xi_{\mathrm{t}}^{-N},p_{\mathrm{Type1}}^{\mathrm{comp}}(M,\xi_{\mathrm{t}})+{N_0}^{N}\left(2^{NR_1}M+2^{NR_2}\right)\xi_{\mathrm{t}}^{-N}\right].
\end{aligned}
$$

□

Appendix F
Proof of Theorem 8

Proof. Since both w and q_i are non-negative random variables and independent of each other, we have $u_1 = w + q_i$ is larger than w and q_i, and

$$
\begin{cases}
\Pr\{u_1 > x\} \geq \Pr\{w > x\};\\
\Pr\{u_1 > x\} \geq \Pr\{q_i > x\};\\
\Pr\{u_1 > x\} \leq \Pr\left\{w > \frac{x}{2}\right\} + \Pr\left\{q_i > \frac{x}{2}\right\}.
\end{cases} \qquad (63)
$$

By substituting Eq. (63) into Ξ_1 in Eq. (41), we have

$$
\begin{cases}
\Xi_1 \geq \lim\limits_{x\to\infty}\left\{\frac{\Pr\left\{w\geq k_c x^{\frac{1}{3}}\right\}}{x^{-N}}\right\}^{\frac{1}{N}};\\
\Xi_1 \geq \lim\limits_{x\to\infty}\left\{\frac{\Pr\left\{q_i\geq k_c x^{\frac{1}{3}}\right\}}{x^{-N}}\right\}^{\frac{1}{N}};\\
\Xi_1 \leq \lim\limits_{x\to\infty}\left\{\frac{\Pr\left\{w\geq \frac{k_c x^{\frac{1}{3}}}{2^{\frac{1}{3}}}\right\}+\Pr\left\{q\geq \frac{k_c x^{\frac{1}{3}}}{2^{\frac{1}{3}}}\right\}}{x^{-N}}\right\}^{\frac{1}{N}}.
\end{cases} \qquad (64)
$$

Thus, by using L'Hôpital's rule, the inequalities in Eq. (64) can be rewritten as

$$
\begin{cases}
\Xi_1 \geq \lim\limits_{x\to\infty}\frac{1}{{k_c}^3}\left\{\frac{f_w(x)}{3Nx^{-(3N+1)}}\right\}^{\frac{1}{N}} = k_c^{-3}(3N)^{-\frac{1}{N}}{\varphi_1}^{\frac{1}{N}};\\
\Xi_1 \geq \lim\limits_{x\to\infty}\frac{1}{{k_c}^3}\left\{\frac{f_q(x)}{3Nx^{-(3N+1)}}\right\}^{\frac{M}{N}} = k_c^{-3}(3N)^{-\frac{1}{N}}{\phi_1}^{\frac{1}{N}};\\
\Xi_1 \leq 8k_c^{-3}\left((3N)^{-1}\varphi_1+(3N)^{-1}\phi_1\right)^{\frac{1}{N}}.
\end{cases} \qquad (65)
$$

Therefore, Ξ_1 can be bounded as follows

$$
k_c^{-3}\theta^{\frac{1}{N}} \leq \Xi_1 \leq 8k_c^{-3}(2\theta)^{\frac{1}{N}}, \qquad (66)
$$

where $\theta = \frac{1}{3N}\max\{\varphi_1,\phi_1\}$, which shows that the upper and lower bounds of Ξ_1 can be determined by φ_1 and ϕ_1. Moreover, Ξ_4 can be bounded by similar method.

□

Appendix G
Proof of Theorem 11

Proof. It is easy to find that

$$
\begin{aligned}
\max\{C(\xi_{\mathrm{c}})-q,0\} &\leq \sum_{c_i\in\mathcal{C}_{\mathrm{ava}}}\max\{C(\xi_{\mathrm{c}})-q_i,0\}\\
&\leq m\max\{C(\xi_{\mathrm{c}})-q,0\}
\end{aligned} \qquad (67)
$$

where $|C_{\mathrm{ava}}| = m$ and $q = \min\limits_{c_i\in\mathcal{C}_{\mathrm{ava}}} q_i$. Then, by taking Eq. (67) into Eq. (25), we have

$$
\begin{aligned}
&p_{\mathrm{Type3}}^{\mathrm{comp}}(m,\xi_{\mathrm{c}}) = \Pr\left\{w > \sum_{c_i\in\mathcal{C}_{\mathrm{ava}}}\max(C(\xi_{\mathrm{c}})-q_i,0)\right\}\\
&\in\left[\Pr\left\{\frac{w}{m}+q > C(\xi_{\mathrm{c}})\right\},\Pr\{w+q > C(\xi_{\mathrm{c}})\}\right].
\end{aligned} \qquad (68)
$$

Since w and q_i are non-negative random variables and independent of each other, we have

$$
\begin{cases}
p_{\mathrm{Type3}}^{\mathrm{comp}}(M,\xi_{\mathrm{c}}) \geq \Pr\left\{\frac{w}{M} > C(\xi_{\mathrm{c}})\right\};\\
p_{\mathrm{Type3}}^{\mathrm{comp}}(M,\xi_{\mathrm{c}}) \geq \Pr\{q > C(\xi_{\mathrm{c}})\};\\
p_{\mathrm{Type3}}^{\mathrm{comp}}(M,\xi_{\mathrm{c}}) \leq \Pr\left\{w > \frac{C(\xi_{\mathrm{c}})}{2}\right\}+\Pr\left\{q > \frac{C(\xi_{\mathrm{c}})}{2}\right\}.
\end{cases} \qquad (69)
$$

By substituting Eq. (69) into Ξ_3 in Eq. (40), we have

$$
\begin{cases}
\Xi_3 \geq \lim\limits_{x\to\infty}\left\{\frac{\Pr\left\{\frac{w}{M}\geq k_c x^{\frac{1}{3}}\right\}}{x^{-N}}\right\}^{\frac{1}{N}};\\
\Xi_3 \geq \lim\limits_{x\to\infty}\left\{\frac{\Pr\left\{q\geq k_c x^{\frac{1}{3}}\right\}}{x^{-N}}\right\}^{\frac{1}{N}} = \lim\limits_{x\to\infty}\left\{\frac{\Pr\left\{q_i\geq k_c x^{\frac{1}{3}}\right\}}{x^{-\frac{N}{M}}}\right\}^{\frac{M}{N}};\\
\Xi_3 \leq \lim\limits_{x\to\infty}\left\{\frac{\Pr\left\{w\geq \frac{k_c x^{\frac{1}{3}}}{2}\right\}+\Pr\left\{q\geq \frac{k_c x^{\frac{1}{3}}}{2}\right\}}{x^{-N}}\right\}^{\frac{1}{N}}.
\end{cases} \qquad (70)
$$

Thus, by using L'Hôpital's rule, the inequalities in Eq. (70) can be rewritten as

$$
\begin{cases}
\Xi_3 \geq \lim\limits_{x\to\infty}\frac{1}{{k_c}^3M^3}\left\{\frac{f_w(x)}{3Nx^{-(3N+1)}}\right\}^{\frac{1}{N}} = k_c^{-3}M^{-3}{k_5}{\varphi_2}^{\frac{1}{N}};\\
\Xi_3 \geq \lim\limits_{x\to\infty}\frac{1}{{k_c}^3}\left\{\frac{f_q(x)}{\frac{3N}{M}x^{-(\frac{3N}{M}+1)}}\right\}^{\frac{M}{N}} = k_c^{-3}{k_6}{\phi_2}^{\frac{M}{N}};\\
\Xi_3 \leq 8k_c^{-3}\left({k_5}^{N}\varphi_2+{k_6}^{N}{\phi_2}^{M}\right)^{\frac{1}{N}},
\end{cases} \qquad (71)
$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCOMM.2019.2920348, IEEE Transactions on Communications

where $k_5 = (3N)^{-\frac{1}{N}}$ and $k_6 = \left(\frac{3N}{M}\right)^{-\frac{M}{N}}$. Hence, Ξ_3 can be bounded as follows

$$k_{\mathrm{c}}^{-3}\theta^{\frac{1}{N}} \leq \Xi_3 \leq 8k_{\mathrm{c}}^{-3}(2\theta)^{\frac{1}{N}}, \tag{72}$$

where $\theta = \max\{{k_1}^N\varphi_1, {k_2}^N\phi_1\}$, which shows that the upper and lower bounds of Ξ_3 can be determined by φ_1 and ϕ_1. □

REFERENCES

[1] D. Han, B. Bai, and W. Chen, "Outage bottleneck for reliable mobile computation offloading: Transmission or computation?" in *Proc. IEEE GlobalSIP*, 2016.

[2] D. Han, W. Chen, B. Bai, and Y. Fang, "On outage of wireless cloud computing: Offloading optimization and bottleneck analysis," in *Proc. IEEE GLOBECOM*, 2017.

[3] R. Kemp, N. Palmer, T. Kielmann, and H. Bal, "Cuckoo: a computation offloading framework for smartphones," in *International Conference on Mobile Computing, Applications, and Services*. Springer, 2010, pp. 59–79.

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

[5] K. Kumar, J. Liu, Y.-H. Lu, and B. Bhargava, "A survey of computation offloading for mobile systems," *Mobile Networks and Applications*, vol. 18, no. 1, pp. 129–140, 2013.

[6] C. Wang and Z. Li, "A computation offloading scheme on handheld devices," *Journal of Parallel and Distributed Computing*, vol. 64, no. 6, pp. 740–746, 2004.

[7] S. Kosta, A. Aucinas, P. Hui, R. Mortier, and X. Zhang, "Thinkair: Dynamic resource allocation and parallel execution in the cloud for mobile code offloading," in *Infocom, 2012 Proceedings IEEE*. IEEE, 2012, pp. 945–953.

[8] M. R. Palacin, "Recent advances in rechargeable battery materials: a chemists perspective," *Chemical Society Reviews*, vol. 38, no. 9, pp. 2565–2575, 2009.

[9] J. Liu, H. Ding, Y. Cai, H. Yue, Y. Fang, and S. Chen, "An energy-efficient strategy for secondary users in cooperative cognitive radio networks for green communications," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 12, pp. 3195–3207, 2016.

[10] Y.-D. Lin, E. T.-H. Chu, Y.-C. Lai, and T.-J. Huang, "Time-and-energy-aware computation offloading in handheld devices to coprocessors and clouds," *IEEE Systems Journal*, vol. 9, no. 2, pp. 393–405, 2015.

[11] Y. Mao, J. Zhang, and K. B. Letaief, "Dynamic computation offloading for mobile-edge computing with energy harvesting devices," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 12, pp. 3590–3605, 2016.

[12] H. T. Dinh, C. Lee, D. Niyato, and P. Wang, "A survey of mobile cloud computing: architecture, applications, and approaches," *Wireless communications and mobile computing*, vol. 13, no. 18, pp. 1587–1611, 2013.

[13] Z. Jiang and S. Mao, "Energy delay tradeoff in cloud offloading for multi-core mobile devices," *IEEE Access*, vol. 3, pp. 2306–2316, 2015.

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

[15] C. You, K. Huang, and H. Chae, "Energy efficient mobile cloud computing powered by wireless energy transfer," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 5, pp. 1757–1771, 2016.

[16] S. Guo, B. Xiao, Y. Yang, and Y. Yang, "Energy-efficient dynamic offloading and resource scheduling in mobile cloud computing," in *Proc. INFOCOM*. IEEE, 2016, pp. 1–9.

[17] M. Salmani and T. N. Davidson, "Multiple access computational offloading with computation constraints," in *Signal Processing Advances in Wireless Communications (SPAWC), 2017 IEEE 18th International Workshop on*. IEEE, 2017, pp. 1–5.

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

[19] S. Guo, J. Liu, Y. Yang, B. Xiao, and Z. Li, "Energy-efficient dynamic computation offloading and cooperative task scheduling in mobile cloud computing," *IEEE Transactions on Mobile Computing*, no. 2, pp. 319–333, 2019.

[20] N. T. Ti and L. B. Le, "Joint computation offloading and resource allocation in cloud based wireless hetnets," 2018.

[21] 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 *INFOCOM 2017-IEEE Conference on Computer Communications, IEEE*. IEEE, 2017, pp. 1–9.

[22] K. Wang, K. Yang, and C. S. Magurawalage, "Joint energy minimization and resource allocation in c-ran with mobile cloud," *IEEE Transactions on Cloud Computing*, vol. 6, no. 3, pp. 760–770, 2018.

[23] X. Ma, Y. Cui, L. Wang, and I. Stojmenovic, "Energy optimizations for mobile terminals via computation offloading," in *Parallel Distributed and Grid Computing (PDGC), 2012 2nd IEEE International Conference on*. IEEE, 2012, pp. 236–241.

[24] T. D. Burd and R. W. Brodersen, "Processor design for portable systems," *Journal of VLSI signal processing systems for signal, image and video technology*, vol. 13, no. 2-3, pp. 203–221, 1996.

[25] S. Kaxiras and M. Martonosi, "Computer architecture techniques for power-efficiency," *Synthesis Lectures on Computer Architecture*, vol. 3, no. 1, pp. 1–207, 2008.

[26] E. Biglieri, J. Proakis, and S. Shamai, "Fading channels: Information-theoretic and communications aspects," *IEEE Transactions on Information Theory*, vol. 44, no. 6, pp. 2619–2692, 1998.

[27] L. Zheng and D. N. C. Tse, "Diversity and multiplexing: A fundamental tradeoff in multiple-antenna channels," *IEEE Transactions on information theory*, vol. 49, no. 5, pp. 1073–1096, 2003.

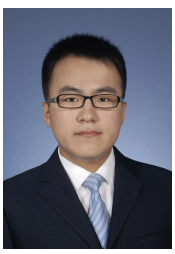
[28] G. Calice, A. Mtibaa, R. Beraldi, and H. Alnuweiri, "Mobile-to-mobile opportunistic task splitting and offloading," in *2015 IEEE 11th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*. IEEE, 2015, pp. 565–572.

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

[30] B. Bai, W. Chen, Z. Cao, and K. B. Letaief, "Max-matching diversity in ofdma systems," *IEEE Transactions on Communications*, vol. 58, no. 4, 2010.

[31] J. R. Lorch and A. J. Smith, "Improving dynamic voltage scaling algorithms with pace," in *ACM SIGMETRICS Performance Evaluation Review*, vol. 29, no. 1. ACM, 2001, pp. 50–61.

[32] S. Foss, D. Korshunov, S. Zachary *et al.*, *An introduction to heavy-tailed and subexponential distributions*. Springer, 2011, vol. 6.



Di Han (S16) received the B.S. degree in information engineering from Shanghai Jiao Tong University, Shanghai, China, in 2015, where he is currently pursuing the Ph.D. degree with the Department of Electronic Engineering in Tsinghua University, Beijing, China. His research interests are in wireless communications, with an emphasis on mobile computation offloading technologies. He has received the Samsung Scholarship in 2013 and National Scholarship in 2014.

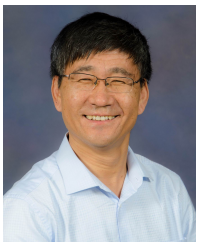
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCOMM.2019.2920348, IEEE Transactions on Communications

Wei Chen (S'05-M'07-SM'13) received his BS and Ph.D. degrees (both with the highest honors) from Tsinghua University in 2002 and 2007. From 2005 until 2007, he was also a visiting PhD student at the Hong Kong University of Science & Technology. Since 2007, he has been on the faculty at Tsinghua University, where he is a tenured full Professor, the director of degree office of Tsinghua University, and a university council member. During 2014 to 2016, he served as a deputy head of Department of Electronic Engineering. He visited Princeton University, Telecom ParisTech, and University of Southampton in 2016, 2014, and 2010 respectively. His research interests are in the areas of communication theory, stochastic optimization, and statistical learning.

Dr. Chen is a Cheung Kong young scholar and a member of national program for special support for eminent professionals, also known as 10,000-talent program. He has also been supported by the national 973 youth project, the NSFC excellent young investigator project, the new century talent program of Ministry of Education, and the Beijing nova program. He received the IEEE Marconi Prize Paper Award and the IEEE Comsoc Asia Pacific Board Best Young Researcher Award in 2009 and 2011, respectively. He serves as an editor of the IEEE TRANSACTIONS COMMUNICATIONS and has served as a TPC co-chair of IEEE VTC-Spring, 2011, as well as, symposium co-chairs for IEEE ICC and Globecom. Dr. Chen is a receipt of the *National May 1st Labor Medal* and the *China Youth May 4th Medal*.

Bo Bai received the B.S. degree with the highest honor in School of Communication Engineering from Xidian University, Xian China, 2004, and the Ph.D. degree in Department of Electronic Engineering from Tsinghua University, Beijing China, 2010. He received the Honor of Outstanding Graduates of Shaanxi Province and the Honor of Young Academic Talent of Electronic Engineering in Tsinghua University. He was a Research Assistant from April 2009 to September 2010 and a Research Associate from October 2010 to April 2012 with the Department of Electronic and Computer Engineering, Hong Kong University of Science and Technology. From July 2012 to January 2017, he was an Assistant Professor with the Department of Electronic Engineering, Tsinghua University. He has obtained the support from Backbone Talents Supporting Project of Tsinghua University. Currently, he is a Senior Researcher at Theory Lab, 2012 Labs, Huawei Technologies Co., Ltd., Hong Kong. He is leading a team to develop fundamental principles, algorithms, and systems for graph informatics, B5G/6G, and network information theory.

He is an IEEE Senior Member, USENIX Member, and has authored about 100 papers in major IEEE/ACM journals and conferences, 2 book chapters, and 1 textbook. He is one of the founded vice chairs of IEEE TCCN SIG on Social Behavior Driven Cognitive Radio Networks. He served as a committee member in IEEE ComSoc WTC and IEEE ComSoc SPCE. He served as a TPC co-chair of IEEE Infocom 2018 - 1st AoI Workshop and IEEE Infocom 2019 - 2nd AoI Workshop, a TPC co-chair of IEEE ICCC 2018, and an Industrial Forum & Exhibition co-chair of IEEE HotICN 2018. He also served as a TPC member for several IEEE conferences such as ICC, Globecom, WCNC, VTC, and ICCC. He served as a reviewer for several major IEEE/ACM journals and conferences. He was the recipient of the Student Travel Grant at IEEE Globecom 2009. He was invited as a Young Scientist Speaker at IEEE TTM 2011. He was a recipient of the Best Paper Award in IEEE ICC 2016.



Yuguang Fang (F'08) received an MS degree from Qufu Normal University, Shandong, China in 1987, a PhD degree from Case Western Reserve University in 1994, and a PhD degree from Boston University in 1997. He joined the Department of Electrical and Computer Engineering at University of Florida in 2000 and has been a full professor since 2005. He holds a University of Florida Research Foundation (UFRF) Professorship (2017-2020, 2006-2009), University of Florida Term Professorship (2017-2019), a Changjiang Scholar Chair Professorship (Xidian University, Xian, China, 2008-2011; Dalian Maritime University, Dalian, China, 2015-2018), Overseas Adviser, School of Information Science and Technology, Southwest Jiao Tong University, Chengdu, China (2014-present), and Overseas Academic Master (Dalian University of Technology, Dalian, China, 2016-2018).

Dr. Fang received the US National Science Foundation Career Award in 2001, the Office of Naval Research Young Investigator Award in 2002, the 2018 IEEE Vehicular Technology Outstanding Service Award, the 2015 IEEE Communications Society CISTC Technical Recognition Award, the 2014 IEEE Communications Society WTC Recognition Award, and the Best Paper Award from IEEE ICNP (2006). He has also received a 2010-2011 UF Doctoral Dissertation Advisor/Mentoring Award, a 2011 Florida Blue Key/UF Homecoming Distinguished Faculty Award, and the 2009 UF College of Engineering Faculty Mentoring Award. He was the Editor-in-Chief of IEEE Transactions on Vehicular Technology (2013-2017), the Editor-in-Chief of IEEE Wireless Communications (2009-2012), and serves/served on several editorial boards of journals including Proceedings of the IEEE (2018-present), ACM Computing Surveys (2017-present), IEEE Transactions on Mobile Computing (2003-2008, 2011-2016), IEEE Transactions on Communications (2000-2011), and IEEE Transactions on Wireless Communications (2002-2009). He has been actively participating in conference organizations such as serving as the Technical Program Co-Chair for IEEE INFOCOM2014 and the Technical Program Vice-Chair for IEEE INFOCOM'2005. He is a fellow of the IEEE and a fellow of the American Association for the Advancement of Science (AAAS).