Modeling and Analysis of Stochastic Mobile-Edge Computing Wireless Networks

Yixiao Gu, Yao Yao, *Senior Member, IEEE*, Cheng Li, Bin Xia, *Senior Member, IEEE*, Dingjie Xu, and Chaoxian Zhang

***Abstract*—To realize the vision of the Internet of Things (IoT), mobile-edge computing (MEC) has recently emerged as a promising paradigm to meet the computation demand from mobile users (MUs). In this article, we study the network performance in large-scale stochastic MEC wireless networks, where the tasks can be computed locally by the local computation capabilities (LCCs) or be offloaded to MEC servers for edge computing. To this end, a MEC network is modeled featuring random node distribution, dynamic task requests, orthogonal frequency-division multiple access, task retransmission, and parallel computing in MEC servers. Given the model, a 2-D discrete-time Markov chain is first adopted to characterize the task execution process, including local computing and task offloading. Based on the coupling between communication and computing, the average outage probability of the task transmission and the average MEC computation load are derived by integrating the stochastic geometry and queuing theory. Furthermore, by jointly analyzing the local computation latency, transmission latency, and edge computation latency, we derive the average end-to-end latency of the task execution. Our results show that the LCCs in MUs can improve the network performance, including communication and computation performance, in stochastic MEC networks. In addition, useful guidelines for MEC network provisioning and planning are provided to avoid either the local computing or the task offloading being the latency performance bottleneck.**

***Index Terms*—Computation offloading, dynamic traffic, Markov chain (MC), mobile-edge computing (MEC), stochastic geometry.**

I. Introduction

THE INCREASING development of the Internet of Things (IoT) introduces numerous computation-intensive and latency-sensitive tasks, which cannot be easily tackled by mobile users (MUs) owing to their limited processing capabilities. To this end, the network architecture known as mobile-edge computing (MEC), which brings vast resources of central servers to the network edges near MUs, were proposed in [2]. By offloading the tasks to MEC access points (APs), the quality of computation experience of MUs can be greatly improved. To study the large-scale MEC networks with stochastic network topology, the random point process was adopted to model the MEC networks [3]. However, the local computation capabilities (LCCs) have always been neglected in stochastic MEC networks despite the rapid development of integrated chips providing considerable computation resources [4]. In this article, we investigate the benefits incurred by LCCs on the MEC system performance and the coupling between the communication and computation in stochastic MEC networks.

A. Related Work

1) Mobile Computation Offloading: By offloading tasks to the MEC servers, the high computation pressure of MUs can be alleviated and the low-latency experience can be achieved [5]. In the area of MEC, one research thrust focuses on the joint radio-and-computational resource managements to realize the energy-efficient and low-latency mobile computation offloading. Specifically, in [6], the centralized optimal resource allocation policy was proposed for the multiuser MEC system. The distributed resource allocation schemes were considered in [7] by proposing the game-theoretic approach. By utilizing the collaboration spaces formed from dividing edge servers into cooperative networks, the computation offloading strategy based on the IoT devices density was designed in [8] to improve the quality of service experience. To enable ubiquitous edge computing, heterogeneous MEC networks comprising central cloud servers and edge servers were proposed in [9]. Based on this architecture, the high accuracy and fine-grained mechanism by jointly considering practical factors, such as load balance, etc., was proposed in [10] to obtain the optimal scheduling strategy. Recently, aiming at the problem of possible trustless environment and accessible records of network behaviors[11], the blockchain-based smart contract operations were designed in [12] to provide security protection in parked vehicle-assisted MEC systems. However, the computation offloading policy designed in previous works mostly focused on MUs under the fixed size of workloads

Manuscript received November 29, 2020; revised February 24, 2021; accepted March 17, 2021. Date of publication March 24, 2021; date of current version September 6, 2021. This work was supported in part by the National Natural Science Foundation of China under Grant 61771307, in part by the Ministry of Education of China (MoE)—China Mobile Communication Corporation (CMCC) Science Joint Foundation under Grant MCM20180102; and in part by Shanghai Municipal Science and Technology Major Project under Grant 2021SHZDZX0102. *(Corresponding author: Bin Xia.)*

Yixiao Gu, Yao Yao, Bin Xia, and Dingjie Xu are with the Shanghai Institute for Advanced Communication and Data Science, Shanghai Key Laboratory of Digital Media Processing and Transmission, Department of Electronic Engineering, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: guyixiao0214@sjtu.edu.cn; sandyyao@sjtu.edu.cn; bxia@sjtu.edu.cn; xudingjie1993@sjtu.edu.cn).

Cheng Li is with the Wireless Research Department, Huawei Technologies Company, Ltd., Shanghai 201206, China (e-mail: lichengg@sjtu.edu.cn).

Chaoxian Zhang is with the School of Information Science and Engineering, Xiamen University Tan Kah Kee College, Xiamen 363105, China (e-mail: zhangcx@xujc.com).

Digital Object Identifier 10.1109/JIOT.2021.3068382

and only short-term profits, e.g., the profits of executing a single task, are considered. In reality, the task requests of MUs are generally dynamic rather than static. For such systems, the long-term computation offloading performance is more important and it cannot be effectively characterized by the one-shot analysis and optimization in the above works. Designing and analyzing the computation offloading policy in MEC networks under dynamic traffic motivate the following works.

2) MEC With Dynamic Traffic: The recent works have studied the MEC with dynamic traffic [13]–[17], where the arrived but not yet executed tasks enter the queues in task buffers. Huang *et al.* [13] presented a dynamic offloading algorithm to achieve energy saving while satisfying given application execution time requirement. Assuming that concurrent local and edge executions are feasible, the latency-optimal task scheduling policies were designed in [14] based on the theory of the Markov decision process. Via task scheduling and load balancing, the efficient task offloading and resource management scheme was proposed in [15] to enhance MEC server utilization in the multiedge networks. Recently, efficient deep reinforcement learning algorithms were utilized to study the large-scale MEC networks with dynamic traffic. Specifically, an online resource scheduling framework is proposed in [16] to obtain the optimal strategy of offloading decision, transmission power, and resource allocation. By jointly optimizing the positions of ground vehicles and unmanned aerial vehicle, user association, and resource allocation in real time, the offloading algorithm was designed in [17] to minimize the energy consumption of all the users in large-scale hybrid MEC networks. However, the MEC schemes in previous works are generally investigated in networks with deterministic network topology. With the development of the femtocell technique, the positions of MUs are generally randomly distributed [18]. Moreover, the computation requirements from MUs are different across different service areas (such as downtowns, rural areas, etc.), yielding the irregular deployment of MEC access points (APs). In stochastic networks, due to the distance-dependent signal power decay and the shared nature of the wireless medium, the network geometry has a significant impact on the performance of wireless networks. Studying the impacts of network node geographic features on the MEC system performance drives the active works discussed in the sequel.

3) Stochastic MEC Networks: In the past decades, stochastic geometry and Poisson point processes (PPPs) [18] have been adopted in the study of wireless networks to capture the stochastic nature of practical wireless networks, such as cellular networks [19], cognitive radio networks [20], and cache-enabled networks [21]. However, the PPPs cannot accurately model the user-centric and content-centric deployments, where the nonuniformity and correlation may exist between the locations of network nodes [22]. Accordingly, the Third-Generation Partnership Project has considered clustered models [23]. The models based on Poisson cluster processes (PCPs) [24] have recently been studied for *ad hoc* networks [25], heterogeneous networks [26], and device-to-device networks [27]. Most existing works in this area focused on the effect of network geometry. Based on the interference characterization, they further provide tractable analytical results on the limits of communication performance in wireless networks.

In contrast, due to the more complex network architecture compared with other radio access networks (RANs) [3], the design and analysis of stochastic MEC networks need to jointly account for the following two aspects, i.e., communication and computing. Moreover, the challenges arising from the coupling of communication and computing in stochastic MEC are urgent to be addressed. In order to investigate the network performance in stochastic MEC networks, Ko *et al.* [3] analyzed the communication latency and computation latency under the constraints of radio access connectivity and computer server stability. The MEC-enabled heterogeneous networks were considered in [28], where the successful edge computing probability considering both the computing and communication performance was derived to explore the effects of the association bias on offloading efficiency. Al-Shuwaili and Lawey [29] formulated the offloading latency for the MEC-based scheme with decoupled uplink/downlink association, providing lower offloading latency compared with the conventional offloading with coupled uplink/downlink association. To maximize the cloud service provider's profit while guaranteeing outage probability requirements, the optimal cloudlet deployment was investigated in [30].

One limitation of the above works is the effects of LCCs at MUs on the MEC network performance in stochastic MEC networks are not fully investigated. On the one hand, though the impacts of the LCC have been widely investigated in small-scale MEC systems. The topological randomness in the network geometry is generally ignored as the number of network nodes is limited. In stochastic networks, the network geometry has a significant impact on the performance of wireless networks [18]. From the practical perspective, how much performance improvement actually can be reaped via utilizing LCC in stochastic MEC networks is urgent to be answered theoretically. On the other hand, the coupling between the communication and computing arising from LCC needs to be addressed. For example, since the task executing by offloading to MEC servers will introduce interference and that by local computing will not, the interference characterization in stochastic MEC networks depends both on the spatial location of the nodes and LCC in MUs. That is, the communication performance is fundamentally limited by LCCs. These coupling relationships introduced by LCCs call for developing a sophisticated analytical approach to studying the MEC network performance and deployment.

B. Contributions

In this article, we study the modeling and analysis of the multicell stochastic MEC wireless networks, where MUs can process tasks with idle local computing resources. The proposed model is not only sufficiently practical but also allows the tractable approach of analyzing network performance. We focus on the stochastic MEC wireless networks under dynamic traffic, with modeling the locations of the MEC servers and MUs as the PCPs. From the communication point of view, the orthogonal frequency-division multiple access (OFDMA) [31] is adopted to enable multiple access of multiple MUs, and the task retransmission is considered to guarantee reliable task offloading. From the computation point of view, each MU has one local computation buffer and one transmission buffer to store tasks that can be computed locally with idle local resources or be offloaded to the MEC servers, respectively. For each MEC server at AP, it can compute multiple tasks simultaneously by parallel computing. The main contributions of this article are listed as follows.

1) *Revealing the Benefits of LCC:* In order to reveal the benefits of utilizing LCC, the 2-D discrete-time Markov chain (MC) is first adopted to characterize the task execution process. Then, by combining the queuing theory and stochastic geometry approaches, the average outage probability for task transmission is derived based on the coupling relationships between the communication and computation. Finally, the average offloading throughput defined as the average task arrival rate at the MEC server is investigated to quantify the effects of LCC on the computation performance.
2) *Analysis of the End-to-End (E2E) Latency:* The E2E latency, which is defined as the period of time from when the task arrives at MU to when the MU obtains the computation result, is analyzed to evaluate the latency performance. We first derive the probability that an arbitrary task is computed locally or offloaded to the MEC server. Then, the conditional E2E latency, i.e., the E2E latency for a given initial buffer state, is derived by jointly analyzing the local computation latency, task transmission latency, and edge computation latency. Based on the conditional E2E latency and the stationary distribution of MC, we obtain the average E2E latency.
3) *Network Provisioning and Planning:* The simulation results verify the analytical accuracy and provide further insights for network provisioning and planning. On the one hand, for the MEC networks with random node distribution, the idle local resources in MUs are worthy to be utilized since it can improve the communication performance and computation performance. On the other hand, we show the conditions under which either the LCC in MUs or the edge computation resources can be a latency performance bottleneck.

C. Outline

The remainder of this article is organized as follows. In Section II, we elaborate on the system model. The network performance and latency performance are analyzed in Sections III and IV, respectively. In Section VI, we provide the simulation results, followed by our conclusions in Section VII.

II. System Model

Consider a discrete-time multicell wireless MEC network consisting of wireless APs with co-located MEC servers and MUs as illustrated in Fig. 1. Each MU has two limited size buffers: one local computation buffer and one transmission buffer, so that the arrived but not yet executed tasks will be queued in buffers. Each task can either be executed locally at the MU with idle computation resources or be offloaded to the MEC server when the local computation buffer is full. In the following, the network model, computation model, and task retransmission model will be discussed in detail.

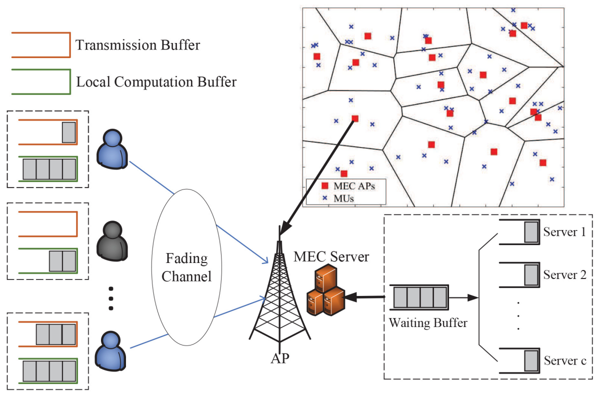

Fig. 1. Multicell MEC-enabled network consisting MUs with LCC.

A. Network Model

The APs (with co-located MEC servers) are spatially located on a 2-D plane by PPPs Φ_c with intensity λ_c. Each AP is active in the closed-access mode and its covered MUs are located around it based on a conditionally independent offspring cluster process. The union of all the offspring points constitutes PCPs. The MUs around each parent AP $x \in \Phi_c$ are assumed to be independent and identically distributed (i.i.d.) according to a zero-mean Gaussian distribution with variance σ^2. Therefore, the density function of the MU location $y \in R^2$ relative to AP is [26]

$$f_Y(y) = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{\|y\|^2}{2\sigma^2}\right). \tag{1}$$

The number of MUs per cell is assumed to be Poisson distributed with mean m. Without loss of generality, we focus on a *typical MU*, which is a randomly chosen MU in a randomly chosen cluster, termed *representative cluster*. Due to the stationarity of this process, the cluster center (i.e., the AP) of the representative cluster and the typical MU is assumed to be located at the origin and y_0, respectively. The uplink channel with fixed bandwidth of B Hz is shared by all MUs who need to transmit tasks, where OFDMA is adopted to enable the multiple access.[1] Therefore, no intracell interference exists and only intercell interference needs to be considered.

Remark 1 (PPP Versus PCP): The PPP-based model (i.e., the MUs and APs are spatially distributed by two independent PPP) and the PCP-based model (i.e., the network spatial model in our work) are two common point process for modeling and designing the stochastic wireless networks. The PPP-based model is suitable for the networks where the location of MEC APs and users is independent, while the PCP-based model can abstract the networks where the MUs are clustered around

[1]In this article, we focus on the basic radio access model, i.e., OFDMA, to get first-order insights into the design and analysis of the stochastic MEC networks with LCC. The bandwidth resource allocation is beyond the scope of this paper can be promising topics for the future.

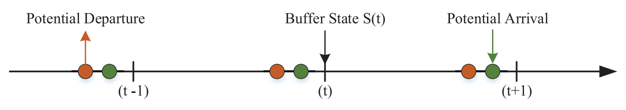

Fig. 2. Various time epochs at which arrivals and departures take place.

MEC APs. In our work, MEC APs are assumed to be deployed in proximity of users to push computing resources to the network edge. This user-centric deployment of MEC servers results in the coupling between the locations of users and APs. Therefore, the PCP-based model is adopted to characterize the random node distribution.

B. Computation Model

The time axis is divided into time intervals with fixed length τ, and is indexed by $\mathbf{T} = \{0, 1, 2, \ldots, \}$. The task arrival process of different MUs across different slots is modeled as i.i.d. Bernoulli process with mean value $\lambda \in [0, 1]$, and the size of each task is l bits. For each MU, the local computation buffer B_1 and transmission buffer B_2 can provide a finite buffer space of size $b_1 \times l$ bits and $b_2 \times l$ bits, respectively. The various time epochs at which events take place are described in Fig. 2. All task arrivals and departures (local computation or transmission) occur at the end of the respective time slots. If at the same slot an arrival and a departure occur, we follow the departure before arrival rule [32]. The state of the MU buffers is observed at slots $t \in \mathbf{T}$ just after possible departures and arrivals have happened, and it is described by a discrete-time stochastic process $\mathcal{S} = \{S(t) = (s_1(t), s_2(t)) : t \in \mathbf{T}\}$, where $s_1(t)$ and $s_2(t)$ denote the number of tasks in local computation buffer and transmission buffer, respectively.

The arrival tasks are computed by LCC when the MU has idling local resources, i.e., the local computation buffer is not full. When the local computation buffer is full, the arrival tasks will enter the transmission buffer and wait to be offloaded to MEC servers. In addition, when MU has idle local resources, and meanwhile, there exist tasks waiting in the transmission buffer, the task at the head of the transmission buffer will be moved to the local computation buffer. This scheduling event is assumed to occur after the possible arrival. Moreover, since both two buffers are finite in size, the arrivals tasks will be dropped if both two buffers are full. In the following parts, the local computing model and edge computing mode are described in detail.

1) Local Computing Model: The first-come–first-served (FCFS) rule is adopted to serve the tasks in the local computation buffer. In each slot, the task at the head of the local computation buffer is processed until it is computed and then leaves the buffer. The local computation of a task can start or end at slot boundaries only, so the local computation time for each task always consists of an integer number of full slots. The local computation rate of MU μ_1 is determined as $\mu_1 = f_1/(vl)$, where v is the number of CPU cycles required to compute 1 b of a computation task and f_1 is the computing capacity of each MU measured the number of CPU cycles per second. Considering the elastic features in the computational complexity [33], the execution requirements (measured by the number of CPU cycles) of the tasks are i.i.d. random variables. Therefore, the distribution of the local computation time for one task T_l is assumed to be i.i.d. random variables and geometrically distributed with parameter $\mu_1(0 < \mu_c < 1)$, i.e., with common probability mass function (PMF)

$$\mathbf{P}(T_l = n) = \mu_1(1 - \mu_1)^{n-1}, \quad n \geq 1. \tag{2}$$

Accordingly, the task departure process for the tasks in local computation buffer follows an i.i.d. Bernoulli processes with mean value μ_1.

2) Edge Computing Model: To offload a computation task to the MEC server, the MU attempts to deliver the task at the head of the transmission buffer to AP in each slot. Accordingly, the task departure process follows an i.i.d. Bernoulli processes with mean value μ_2 provided that the wireless link is reliable [34], and μ_2 is determined by

$$\mu_2 = 1 - P_{\text{out}} \tag{3}$$

where P_{out} is the average outage probability.

Upon arriving at APs, the tasks are assumed to be delivered to the MEC servers without any delay and queue at the MEC servers buffer for computation. The MEC server has an infinite size buffer to store tasks and c computation servers to provide parallel computing of multiple tasks. The task arrival and computation process are assumed to be mutually independent, and tasks are served according to the FCFS queueing discipline. The arrival tasks wait in the buffer for some time and are then computed via one of the c computation servers, after which the computation results will be transmitted to corresponding MUs. The computation time for one task g_c in each computation server is assumed to be i.i.d. geometrically distributed random variables parameter $\mu_c(0 < \mu_c \leq 1)$, i.e., with common PMF

$$\mathbf{P}(g_c = n) = \mu_c(1 - \mu_c)^{n-1}, \quad n \geq 1 \tag{4}$$

and the probability generation function (PGF) is

$$G_c(z) = \sum_{n=1}^{\infty} \mathbf{P}(g_c = n) z^n = \frac{\mu_c z}{1 - (1 - \mu_c) z}. \tag{5}$$

Finally, note that the computation results typically have small sizes compared with offloaded tasks [3]. We assume that transmission can be finished within one slot and the MUs can always successfully receive the results.

3) Task Retransmission Model: Considering the network interference and the channel fading [35], computation tasks may not be successfully received for once transmission. Therefore, the task retransmission strategy based on the automatic repeat request (ARQ) is considered to ensure reliable task transmission [36]. When the outage occurs, the task keeps staying at the head of the buffer and the corresponding packet will be retransmitted in the following time slots until it is successfully received. The error-free feedback of the acknowledgment and the negative acknowledgment (ACK/NACK) is assumed, and both the ACK/NACK transmission time and the corresponding round trip delay are neglected.

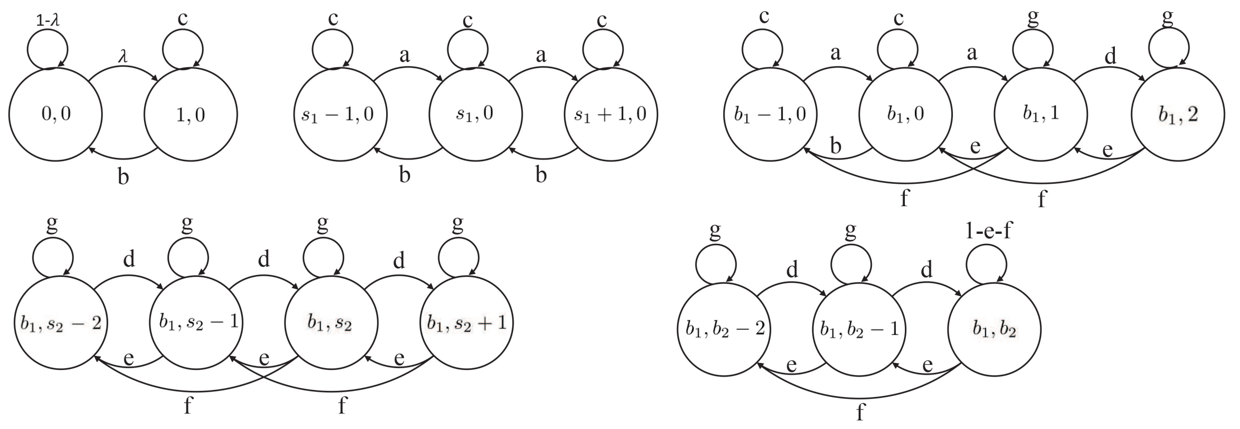

Fig. 3. State transition diagram of the 2-D discrete-time MC model for the MU buffers.

III. Network Performance Analysis

In this section, we analyze the network performance, including communication performance (i.e., average outage probability) and computation performance (i.e., average MEC computation load), based on the coupling between the communication and computing. The main issues addressed in this section are summarized as follows.

1) The buffer state transition is first analyzed, based on which we derive the buffer state stationary distribution.
2) The average outage probability is derived by combining the stochastic geometry approach and buffer analysis results.
3) Based on the above results, the average MEC computation load is derived by analyzing the task execution process.

A. MU Buffer Analysis

The task processing process for each MU is modeled by a 2-D discrete-time time-homogeneous MC $\mathcal{S} = \{s(t) = (s_1(t), s_2(t)) : t \in \mathbf{T}\}$, and the MC is described by $P_{(i,j)|(u,k)} = \Pr\{s(t+1) = (i,j)|s(t) = (u,k)\}$, where $P_{(i,j)|(u,k)}$ denotes the probability of transition from state $s(t) = (u,k)$ to state $s(t+1) = (i,j)$ in a single step, i.e., the one-step transition probability. According to the system model described preceding section, the buffer state $s(t+1)$ depends on the buffer state $s(t)$, task arrival, and task processing (including local computation and transmission). As a result, there are three cases described as follows.

1) *Case 1:* Buffer state $s(t) = (0,0)$. The buffer state $s(t+1)$ depends on the task arrival. The set of $s(t+1)$ is $\{(0,0), (1,0)\}$.
2) *Case 2:* Buffer state $s(t) = (s_1, 0)$, where $1 \le s_1 \le b_1$. The buffer state $s(t+1)$ depends on the task arrival and local computing result. When $1 \le s_1 < b_1$, the set of $s(t+1)$ is $\{(s_1-1,0), (s_1,0), (s_1+1,0)\}$. When $s(t) = (b_1, 0)$, the set of $s(t+1)$ is $\{(b_1-1,0), (b_1,0), (b_1,1)\}$.
3) *Case 3:* Buffer state $s(t) = (b_1, s_2)$, where $1 \le s_2 \le b_2$. The buffer state $s(t+1)$ depends on the task arrival and task processing results (including local computation and transmission). Note that if one task departs local computation buffer and no new task arrives, the task at the head of transmission buffer will be moved to the local computation buffer. Therefore, when $s(t) = (b_1, 1)$, the set of $s(t+1)$ is $\{(b_1-1,0), (b_1,0), (b_1,1), (b_1,2)\}$. When $s(t) = (b_1, s_2)$, where $2 \le s_2 < b_2$, the set of $s(t+1)$ is $\{(b_1, s_2-2), (b_1, s_2-1), (b_1, s_2), (b_1, s_2+1)\}$. When $s(t) = (b_1, b_2)$, the set of $s(t+1)$ is $\{(b_1, b_2-2), (b_1, b_2-1), (b_1, b_2)\}$.

TABLE I
Transition Probabilities of the MC Shown in Fig. 3

Notations	Value
a	$\lambda(1-\mu_1)$
b	$(1-\lambda)\mu_1$
c	$\lambda\mu_1 + (1-\lambda)(1-\mu_1)$
d	$\lambda(1-\mu_1)(1-\mu_2)$
e	$(1-\lambda)[\mu_1(1-\mu_2) + \mu_2(1-\mu_1)] + \lambda\mu_1\mu_2$
f	$(1-\lambda)\mu_1\mu_2$
g	$(1-\lambda)(1-\mu_1)(1-\mu_2) + \lambda[\mu_1(1-\mu_2) + \mu_2(1-\mu_1)]$

The state transition diagram of MU is shown in Fig. 3 and the transition probabilities are listed in Table I.

According to the one-step transition probability, it can be observed that: 1) the MC process $\mathcal{S}$ is irreducible; 2) the period of each state is 1; and 3) each state is recurrent, so each state is aperiodic. Let $\pi_{(s_1,s_2)}$ be the steady probability of state $\mathbf{s} = (s_1, s_2)$ in the MC model. Then, we can conclude that its stationary distribution $\Pi = [\pi_{(0,0)}, \pi_{(1,0)}, \ldots, \pi_{(b_1,b_2)}]$ uniquely exists. Note that the state space in this 2-D MC is made up of finite vectors s_1, s_2. Therefore, it can be converted to scalars and then one obtains a single-dimensional MC [37]. Define the MC process $\mathcal{X} = \{x(t) = s_1(t) + s_2(t)) : t \in \mathbf{T}\}$ and its finite state space is $\{0, 1, \ldots, b_1 + b_2\}$. The process $\mathcal{X}$ is described by $P_{m,n} = \Pr\{x(t+1) = m|x(t) = n\}$, where $P_{m,n}$ denotes the probability of transition from state $x(t) = n$ to the state $x(t+1) = m$ in a single step. Since state $x(t) = s_1(t) + s_2(t)$ in MC $\mathcal{X}$ and state $s(t) = (s_1(t), s_2(t))$ in MC $\mathcal{S}$ are one-to-one correspondence, one obtains $P_{i+j,u+k} = P_{(i,j),(u,k)}$ and the corresponding transition probability matrix $\mathbf{P}_x$. Then, the stationary distribution for the MC process $\mathcal{X}$ defined as $\Pi_x = [\pi_0^x, \pi_1^x, \ldots, \pi_{b_1+b_2}^x]$ can be obtained by solving the following linear system of equations:

$$\Pi_x \cdot \mathbf{P}_x = \Pi_x, \quad \Pi_x \cdot \mathbf{1} = 1 \tag{6}$$

where $\mathbf{1}$ denotes the column vector of ones. Then, one obtains the stationary distribution $\Pi = [\pi_{(0,0)}, \pi_{(1,0)}, \ldots, \pi_{(b_1,b_2)}]$, where $\pi_{s_1,s_2} = \pi^x_{s_1+s_2}$.

B. Outage Probability Analysis

The MU that has tasks in the transmission buffer needs to transmit the task to AP. Recall that the number of MUs that have arrival tasks around each AP is Poisson distributed with mean m. Then, the number of MUs that have tasks to transmit around each AP is Poisson distributed with mean $\widetilde{m} = m\sum_{s_2=1}^{b_2}\pi_{(b_1,s_2)}$, and the set of MUs with the task to transmit around AP x is denoted by B^x. Assume that the transmit power of each MU is fixed as P_u and the number of MU with the task to transmit around the typical AP is m_0. Then, the received power at typical AP from the typical MU is $P = P_u h_0 r^{-\alpha}$, where $r = ||y_0||$ and $h_0 \sim \exp(1)$ is i.i.d. exponential random variable, which models Rayleigh fading and $\alpha > 2$ is the path-loss exponent.

For the typical AP, the interference comes from MUs who have the task to transmit around other APs. Recall that the AP location is $x \in R^2$, the MU location relative to the AP is y, and the number of MUs who have the task to transmit around each AP $x \in \Phi_c$ is denoted by m_x. Then, the power of interference at the typical AP is $I = \sum_{x\in\Phi_c\backslash x_0}\sum_{y\in B^x} P_u h_{y_x}||x+y||^{-\alpha}$. Here, we consider the interference-limited wireless networks, where the background noise can be negligible compared to the total interference. Recall that the serving distance from the typical MU to its corresponding AP is r, and then the SIR at the typical AP for receiving the task from the typical MU is

$$\begin{aligned}\mathrm{SIR}(r) &= \frac{P_u h_0 r^{-\alpha}}{\sum_{x\in\Phi_c\backslash x_0}\sum_{y\in B^x} P_u h_{y_x}||x+y||^{-\alpha}}\\ &= \frac{h_0 r^{-\alpha}}{\sum_{x\in\Phi_c\backslash x_0}\sum_{y\in B^x} h_{y_x}||x+y||^{-\alpha}}. \end{aligned}\quad (7)$$

In order to analyze the average outage probability, one needs to obtain the average available bandwidth for the typical MU. We thus have the following lemma.

Lemma 1: The average bandwidth for the typical MU is

$$\mathbb{E}_B = B\cdot\mathbb{E}_m\left\{\frac{1}{m}\right\} \quad (8)$$

where $\mathbb{E}_m\{(1/m)\} = e^{-\widetilde{m}}\sum_{n=1}^{\infty}(\widetilde{m}^n/n\cdot n!)$.

Proof: See Appendix A. ■

The condition for the MEC server to receive the task from the typical MU is that the SIR should exceed a fixed decoding threshold β. The decoding threshold should satisfy $\mathbb{E}_B\cdot\tau\log_2(1+\beta) = l$, where τ is the length of a slot (in second) and l is the number of bits for one task. The outage probability is then formally defined as the probability that the SIR is lower than the decoding threshold β and can be mathematically expressed as $P_{\text{out}} = \mathbb{E}_r[\mathbb{P}\{\mathrm{SIR}(r) < \beta \mid r\}]$.

To derive the outage probability, we first characterize the distribution from the typical AP to the MUs around other APs. Recall that the locations of APs and MUs are modeled by PCPs. Then, the density function of the serving distance r, i.e., the distance between the typical MU to its corresponding AP, is given by $f_R(r) = (1/2\pi\sigma^2)\exp(-[r^2/2\sigma^2])$. In addition, according to [38], the conditional probability density function of the distances from the intercluster MU to the AP of the representative cluster $u = ||x+y||$, condition on the serving distance r, is $f_U(u|r) = (u/\sigma^2)\exp(-[u^2+r^2/2\sigma^2])I_0(ur/\sigma^2)$, where $I_0(.)$ is the modified Bessel function of the first kind with order zero and σ is the scale parameter. To simplify certain order statistics arguments in the sequel, we assume that the maximum number of MUs per cluster is M [27].

Then, based on the definition of the outage probability, one obtains the following theorem.

Theorem 1: The average outage probability for the typical MU is

$$P_{\text{out}} = 1 - \int_0^{\infty}\mathscr{L}_I\big(\beta r^{\alpha}\big)f_R(r)dr \quad (9)$$

and $\mathscr{L}_I(s) =$

$$\exp\left(-2\pi\lambda_c\int_0^{+\infty}\left(1-\sum_{k=0}^{\widetilde{m}}(\rho(v))^k\frac{\widetilde{m}^k e^{-\widetilde{m}}}{k!}\right)v dv\right) \quad (10)$$

where $\rho(v) = \int_0^{+\infty}(1/[1+su^{-\alpha}])f_U(u|r)du$.

Proof: See Appendix B. ■

Solving the equations obtained by combining (3), (6), and (9), the detailed numerical expressions of the average outage probability can be obtained.

C. MEC Computation Load Analysis

In this part, the MEC computation load is analyzed by studying the task arrival process at the MEC server. Note that for each MU, the average probability that it has task to transmit is $\sum_{s_2=1}^{b_2}\pi_{(b_1,s_2)}$. Therefore, the probability that the MU can successfully transmit one task to the MEC server in each slot is

$$p_c = \sum_{s_2=1}^{b_2}\pi_{(b_1,s_2)}(1-P_{\text{out}}). \quad (11)$$

Recall that in each cell, there are M MUs with the task to be offloaded. Therefore, the number of tasks arriving in the MEC server at each slot follows the i.i.d binomial distribution with parameters M and p_c, with common PGF $A(z) = (p_c z + 1 - p_c)^M$. Then, the average computation load of MEC servers, which can be measured by the expected task arrival rate (in the number of tasks per slot) at the typical AP, can be derived as

$$\Lambda = A(1)' = Mp_c = M\sum_{s_2=1}^{b_2}\pi_{(b_1,s_2)}(1-P_{\text{out}}). \quad (12)$$

It can be observed that in order to keep steady for the MEC system, the average number of tasks arriving during each slot must be smaller than the average number of packets that can be computed by the MEC server [39]. Then, the steady condition is given by $A'(1) < c\mu_c$; in the analysis that follows, we assume this condition is always satisfied.

IV. Latency Analysis

In this section, we derive the average E2E latency of task execution. The E2E latency is defined as the number of slots from when the task arrives at MU to when the MU obtains the computation results. Consider an arbitrary task denoted by **A**; on the one hand, its E2E latency is related to the buffer state before its arrival. Specifically, when the local computation buffer is not full, it will enter the local computation buffer. Otherwise, it will enter the transmission buffer. On the other hand, there exists the temporal correlations of task execution under dynamic traffic. Accordingly, the E2E latency is related to the task processing result in each slot from when the task **A** arrives to when the result of new task **A** is obtained. Therefore, in order to derive the average E2E latency, all possible buffer states need to be traversed and the executing processes for all tasks in MU's buffer should be analyzed. The outline to derive the average E2E latency is summarized as follows.

1) Based on the buffer analysis results in Section III-A, the task processing mode, i.e., the probability that task **A** is computed by LCC or the MEC server, is first analyzed.
2) By analyzing the queue dynamic in the MEC server, the average edge computation latency is obtained based on the MEC load analysis results in Section III-C.
3) Based on the above results, the average E2E latency of task **A** after entering the buffer is first analyzed. Then, we derive the average E2E latency for a given $s(t-1)$. Finally, one obtains the average E2E latency.

A. Task Processing Mode Analysis

In this part, we analyze the conditional probability that when the buffer state is $s(t) = (s_1, s_2)$ after task **A** entering buffer is processed by local computing (denoted by $P_{c|(s_1,s_2)}$) or being transmitted to the MEC server (denoted by $P_{t|(s_1,s_2)}$). Note that when $s(t)(s_1, 0)$ where $1 \leq s_1 \leq b_1$, **A** enters the local computation buffer and it will be processed by local computing. When **A** enters transmission buffer [i.e., $s(t) = (b_1, s_2)$], it can be processed by edge computing or be scheduled to the local computation buffer and then be local computed. This scheduling event happens when in one slot, no new task arrives and the task at the head of the buffer B_1 departs. We denote this scheduling probability by p_s and we have $p_s = \mu_1(1-\lambda)$. It can be observed that the conditional probability of task processing mode depends on the number of tasks in transmission buffer s_2. For conciseness, we simplify $P_{c|(s_1,s_2)}$ and $P_{t|(s_1,s_2)}$ to $P_{c|s_2}$ and $P_{t|,s_2}$, respectively.

When buffer state is $s(t) = (s_1, 0)$ after task **A** arrives, it dictates that **A** enters local computation buffer and it will be computed by local computing, i.e.,

$$P_{c|0} = 1, \quad P_{t|0} = 0. \tag{13}$$

Consider buffer state $s(t) = (b_1, 1)$ after the arrival of task **A**. We thus have the following proposition.

Proposition 1: The conditional probability of task **A** being processed by local computing is

$$\mathbb{P}_{c|1} = \frac{(1-\mu_2)p_s}{1-(1-\mu_2)(1-p_s)} \tag{14}$$

and the conditional probability of task **A** being processed by edge computing is

$$\mathbb{P}_{t|1} = \frac{\mu_2}{1-(1-\mu_2)(1-p_s)}. \tag{15}$$

Proof: See Appendix C. ■

From (14) and (15), we have $P_{c|1} + P_{t|1} = 1$, which is a check for the correctness of our analysis. When transmission buffer B_2 has tasks, the number of tasks departing buffer B_2 in each slot can be 0, 1, or 2 depends on the task transmission and task scheduling results. We denote the probability that the number of departing tasks is i by p_i^d, and then one obtains $p_0^d = (1-\mu_2)(1-p_s)$, $p_1^d = (1-\mu_2)p_s + \mu_2(1-p_s)$ and $p_2^d = \mu_2 p_s$.[2] Consider buffer state $s(t) = (b_1, s_2)$ after the arrival of task **A** where $s_2 > 1$. In the following slot, the buffer state $s(t+1)$ can be (s_1, s_2), (s_1, s_2-1), and (s_1, s_2-2). Thus, we have

$$P_{c|s_2} = p_0^d P_{c|s_2} + p_1^d P_{c|s_2-1} + p_2^d P_{c|s_2-2} \tag{16}$$

$$P_{t|s_2} = p_0^d P_{t|s_2} + p_1^d P_{t|s_2-1} + p_2^d P_{t|s_2-2}. \tag{17}$$

Then, $P_{c|s_2}$ and $P_{t|s_2}$ are, respectively, given by

$$P_{c|s_2} = \frac{p_1^d P_{c|s_2-1} + p_2^d P_{c|s_2-2}}{1-p_0^d} \tag{18}$$

$$P_{t|s_2} = \frac{p_1^d P_{t|s_2-1} + p_2^d P_{t|s_2-2}}{1-p_0^d}. \tag{19}$$

From (13)–(15), (18), and (19), one obtains the conditional probability $P_{c|s_2}$ and $P_{t|s_2}$.

B. Average Edge Computation Latency Analysis

In this part, we first analyze PGF of the number of tasks in the MEC server. Let us denote the number of tasks arriving during slot k by a_k, the number of tasks at the beginning of slot k by q_k, and the number of tasks departing during slot k by d_k. Then, according to the edge computation model, we have $q_{k+1} = q_k + a_k - d_k$, where $d_k = \sum_{j=1}^{(q_k,c)^-} d_{k,j}$ with $(q_k, c)^- \triangleq \min(q_k, c)$ and $d_{k,j}$ corresponds to the number of tasks departing at slot k via the jth computing server. Since the edge computation times are geometric distributed, $d_{k,j}$ is a Bernoulli distributed random variable with parameter μ_c, i.e.,

$$d_{k,j} = \begin{cases} 1, & \text{with probability } \mu_c \\ 0, & \text{with probability } 1-\mu_c. \end{cases} \tag{20}$$

From [40], let $H(z) = \mu_c + (1-\mu_c)z$ and $A(z) = (p_c z + 1 - p_c)^M$, and define $z_j, j = 1, 2, \ldots, c-1$ as the $c-1$ roots of $H(z)^c A(z) - z^c = 0$ inside the unit disk, i.e., $|z| < 1$. Then, the average number of tasks in the MEC server is given by

$$\mathbb{E}[q] = \frac{\mu_c(2-\mu_c)}{2[c\mu_c - A'(1)]} \sum_{i=1}^{c-1} \left(c^2 - i^2\right) q(i) + \frac{A^* + 2cA'(1) - (2-\mu_c)\left[c^2\mu_c - A'(1)\right]}{2[c\mu_c - A'(1)]} \tag{21}$$

where $A^* = A''(1) - 2A'(1)^2$, $q(i), i = 0, 1, \ldots, c-1$ are solutions of the following equation:

$$\begin{cases} \mu_c \sum_{i=0}^{c-1} (c-i) q(i) = c\mu_c - A'(1) \\ \sum_{i=0}^{c-1} \left[H(z_j)^{i-c} - z_j^{i-c}\right] q(i) = 0, & j = 1, 2, \ldots, c-1. \end{cases} \tag{22}$$

[2]Note that when MU has one task in the transmission buffer, i.e., buffer state $s(t) = (b_1, 1)$, p_2^d denotes that the MU has no task arriving and the MU can process two tasks during slot t, so we have $s(t+1) = (b_1 - 1, 0)$.

Then, the average edge computation latency is given by

$$\mathbb{E}[T_e] = \frac{\mathbb{E}[q]}{A'(1)} = \frac{\mathbb{E}[q]}{M\sum_{s_2=1}^{b_2}\pi_{(b_1,s_2)}(1-P_{\text{out}})}. \tag{23}$$

Note that $A(z)$ is joint determined by the network parameters and the parameters of LCC. Therefore, in addition to the parameters of the MEC server, the edge computation performance is also depends on local computation performance and communication performance.

C. E2E Latency Analysis

Consider task **A** arriving during slot t. In this part, we first analyze the E2E latency that task **A** experienced [denoted by $\bar{T}_a(s_1, s_2)$] when buffer state is $s(t) = (s_1, s_2)$ after its arrival. Then, the E2E latency that task **A** experienced [denoted by $\bar{T}(s_1, s_2)$] when buffer state is $s(t-1) = (s_1, s_2)$ before its arrival is derived. Finally, we obtain the average E2E latency.

In order to derive $\bar{T}_a(s_1, s_2)$, we first define cases 1 and 2 as the case that task **A** enters local computation buffer B_1 and transmission buffer B_2, respectively. For case 1, consider buffer state $s(t) = (s_1, 0)$ after the arrival of **A**. Due to the FCFS rule, the E2E of **A** is the time it takes the MU to complete the local computation of s_1 tasks. Recall that the local computation time for each task is i.i.d. random variable and geometrically distributed with parameter μ_1. Hence, the average local computation time for each task is $(1/\mu_1)$ [41], and then we have

$$\overline{T_a}(s_1, 0) = \frac{s_1}{\mu_1}. \tag{24}$$

Then, we focus on case 2 and consider buffer state $s(t) = (b_1, s_2)$ after the arrival of task **A**. If task **A** is offloaded to the MEC server, its average E2E latency includes three parts: 1) the average latency in buffer B_2; 2) the average edge computation latency; and 3) one slot for result transmission. Define $P_t(i|s_2)$ as the conditional probability that the number of slot experienced by task **A** in buffer B_2 is i. Denote $T_t(s_2, z)$ as the PGF of $P_t(i|s_2)$ and then we obtain the $T_t(s_2, z)$ as follows.

Proposition 2: The PGF $T_t(s_2, z)$ of $P_t(i|s_2)$ is

$$T_t(1, z) = \frac{\mu_2 z}{1-(1-\mu_2)(1-p_s)z} \tag{25a}$$

$$T_t(2, z) = \frac{p_1^d \mu_2 z^2}{\left(1-p_0^d z\right)\left[1-(1-\mu_2)(1-p_s)z\right]}, \tag{25b}$$

$$T_t(s_2, z) = \frac{p_1^d z T_t(s_2-1, z) + p_2^d z T_t(s_2-2, z)}{1-p_0^d z} \tag{25c}$$

where $3 \le s_2 \le b_2$.

Proof: See Appendix D. ■

Based on Proposition 2, the following lemma is obtained.

Lemma 2: Consider buffer state $s(t) = (b_1, s_2)$ after the arrival of task **A** and task **A** being processed by task offloading. The average E2E latency is

$$\overline{T_a^t}(s_2) = \mathbb{E}[T_t(s_2)] + \mathbb{E}[T_e] + 1 \tag{26}$$

where $\mathbb{E}[T_t(s_2)] = ([dT_t(s_2, z)]/dz)|_{z=1}$.

Proof: For the task processed by edge computing, its average E2E latency includes average latency in buffer B_2, average edge computing latency, and one slot for result transmission. According to the property of PGF, the average latency can be obtained by taking the first-order derivative of $T_t(s_2, z)$, i.e., $\mathbb{E}[T_t(s_2)] = ([dT_t(s_2, z)]/dz)|_{z=1}$. Therefore, the average E2E latency is $\mathbb{E}[T_t(s_2)] + \mathbb{E}[T_e] + 1$. The proof is completed. ■

If task **A** is processed by local computing, it will be scheduled to the local computation buffer B_1. Accordingly, the E2E latency includes two parts: 1) the in buffer B_2 and 2) the local computation latency. When task **A** is scheduled to buffer B_1, the number of tasks in buffer B_1 is b_1 and the average local computation latency is (b_1/μ_1). For the average latency in buffer B_2, we denote the probability that the number of slot experienced by task **A** in buffer B_2 is i by $P_c(i|s_2)$, and the corresponding conditional PGF is $T_c(s_2, z)$. Using the similar approached to derive $T_t(s_2, z)$, we have the following proposition.

Proposition 3: The PGF $T_c(s_2, z)$ of $P_c(i|s_2)$ is

$$T_c(1, z) = \frac{(1-\mu_2)p_s z}{1-(1-\mu_2)(1-p_s)z} \tag{27a}$$

$$T_c(2, z) = \frac{p_1^d(1-\mu_2)p_s z^2}{\left(1-p_0^d z\right)\left(1-\left[(1-\mu_2)(1-p_s)z\right]\right)} \tag{27b}$$

$$T_c(s_2, z) = \frac{p_1^d z T_c(s_2-1, z) + p_2^d z T_c(s_2-2, z)}{1-p_0^d z} \tag{27c}$$

where $3 \le s_2 \le b_2$.

Based on Proposition 3, we obtain the following result with the proof omitted for brevity.

Lemma 3: Consider buffer state $s(t) = (b_1, s_2)$ after the arrival of task **A** and task **A** being processed by local computing. The average E2E latency is

$$\overline{T_a^c}(s_2) = \mathbb{E}[T_c(s_2)] + \frac{b_1}{\mu_1} \tag{28}$$

where $\mathbb{E}[T_c(s_2)] = ([dT_t(s_2, z)]/dz)|_{z=1}$.

Then, based on Lemmas 2 and 3 and the task processing mode analysis in Section IV-A, the E2E latency that task **A** experienced when buffer state is $s(t) = (b_1, s_2)$ after its arrival

$$\overline{T_a}(b_1, s_2) = P_{t|s_2} \cdot \overline{T_a^t}(s_2) + P_{c|s_2} \cdot \overline{T_a^c}(s_2). \tag{29}$$

Now, we focus on the conditional average E2E latency $T_{\text{ave}}(s_1, s_2)$ of task **A**. If local computation buffer B_1 is not full, i.e., $s(t-1) = (s_1, 0)$ and $s_1 < b_1$, task A will enter buffer B_1. When $s_1 = 0$, the buffer state $s(t) = (1, 0)$ after task A arrives and we have

$$\overline{T}(0, 0) = \overline{T_a}(1, 0) = \frac{1}{\mu_1}. \tag{30}$$

Then, for $1 \le s_1 < b_1$, if one task departs buffer B_1 during slot t, buffer state $s(t) = (s_1, 0)$, otherwise, $s(t) = (s_1 + 1, 0)$. Then, we have

$$\overline{T}(s_1, 0) = \mu_1\overline{T_a}(s_1, 0) + (1-\mu_1)\overline{T_a}(s_1+1, 0). \tag{31}$$

Then, consider the case that $s(t-1) = (b_1, s_1)$ and $0 \le s_2 \le b_2$ before the arrival of task **A**. For the buffer state $s(t-1) = (b_1, 0)$, if one task departs buffer B_1 during slot t, task A enters buffer B_1 and we have $s(t) = (b_1, 0)$, otherwise, task A enters buffer B_2 and we have $s(t) = (b_1, 1)$. Thus, we have

$$\overline{T}(b_1, 0) = \mu_1\overline{T_a}(b_1, 0) + (1-\mu_1)\overline{T_a}(b_1, 1). \tag{32}$$

For $0 < s_2 < b_2$, if the MU completes the local computation of one task during slot t, task A enters buffer B_1 and buffer B_1 has b_1 tasks. Otherwise, if the MU successfully transmits one task during slot t, the buffer state $s(t) = (b_1, s_2)$, and if the transmission during slot t fails, the buffer state $s(t) = (b_1, s_2 + 1)$. Thus, we have

$$\overline{T}(b_1, s_2) = \mu_1 \overline{T_a}(b_1, 0) + (1 - \mu_1)\mu_2 \overline{T_a}(b_1, s_2) + (1 - \mu_1)(1 - \mu_2)\overline{T_a}(b_1, s_2 + 1). \quad (33)$$

Finally, for $s(t-1) = (b_1, b_2)$, if the MU completes the local computation of one task during slot t, task A enters buffer B_1 and buffer B_1 has b_1 tasks. Otherwise, if the MU successfully transmits one task during slot t, the buffer state $s(t) = (b_1, s_2)$, and if the transmission during slot t fails, task **A** will be dropped. Thus, we have

$$\overline{T}(b_1, b_2) = \mu_1 \overline{T_a}(b_1, 0) + (1 - \mu_1)\mu_2 \overline{T_a}(b_1, b_2). \quad (34)$$

Based on the stationary distribution Π and conditional average E2E latency $T_{\text{ave}}(b_1, b_2)$, the average E2E latency is given by

$$T_{\text{ave}} = \sum_{s_1=0}^{b_1} \sum_{s_2=0}^{b_2} \pi_{(s_1,s_2)} \overline{T}(s_1, s_2). \quad (35)$$

V. Simulation and Discussion

In this section, we simulate the stochastic MEC networks to validate the theoretical analysis and obtain some useful insights.

A. Simulation Setting

The simulations were implemented in MATLAB. We obtain the result with Monte Carlo methods in a square area of 10×10 km^2. The BS and MUs are scattered in the area based on PPP with density $\lambda_b = (150/1000^2)$ nodes/m^2 and the PCP with parameter $\sigma = 50$, respectively, and the average number of MUs in each cell $M = 10$. The duration of time slot is $\tau = 0.1$ s, the transmission power of each user is $P_u = 23$ dBm, the path-loss exponent is $\beta = 4$, the total bandwidth is $B = 15$ MHz, and the date size of per task is $l = 0.5 \times 10^6$ b. For each MU, the local computation buffer B_1 and transmission buffer B_2 can store up to 2 and 5 tasks, respectively. These parameters do not change unless otherwise stated.

We present the comparisons between Monte Carlo simulations (10^4 realizations) and analytical results. For each realization, the MEC APs and MUs are distributed in the plane based on PPPs and PCPs, respectively. According to the task execution model, the task processing process of each user is simulated. During each realization, we record the key variables, such as the number of MUs with the task to offload, the number of transmission failures, the number of tasks being computed by LCC or the edge servers, the experienced latency for each task, etc. Then, based on these variables, the average outage probability, average MEC computation load, and the task E2E latency are further obtained to evaluate the MEC system performance.

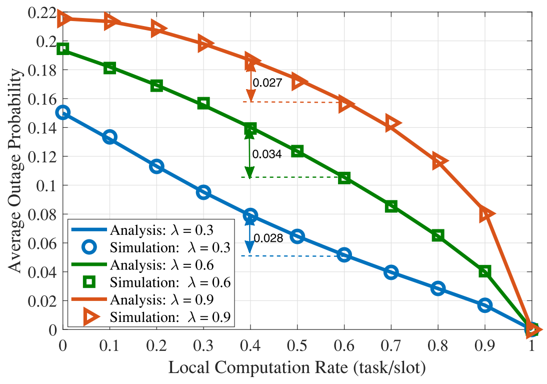

Fig. 4. Effect of local computation rate on average outage probability.

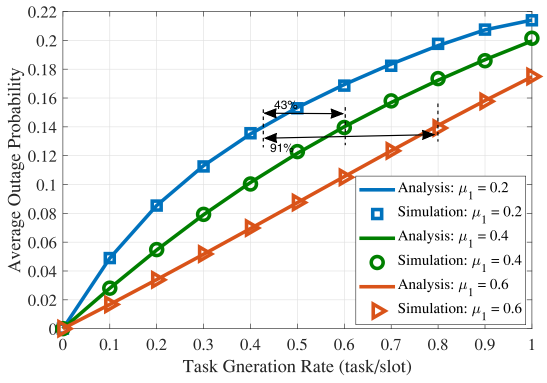

Fig. 5. Effect of task generation rate on average outage probability.

B. Numerical Results

1) Average Outage Probability: The impact of local computation rate μ_1 on the average outage probability is presented in Fig. 4. The benefits incurred by the usage of LCC in communication performance can be observed since the average outage probability decreases with the local computation rate μ_1. It is because MUs with idle computation resources (the local computation buffer is not full) can compute tasks locally. This reduces the number of MUs with tasks to transmit in each cell and hence reduce the intercell interference. For instance, when μ_1 is increased from 0.4 to 0.6, the outage probability can be reduced by 0.027 for $\lambda = 0.3$, 0.034 for $\lambda = 0.6$, and 0.027 for $\lambda = 0.3$. In addition, the outage probability is equal to 0 when $\mu_1 = 1$ since it implies that all tasks can be computed by LCC and no task transmission requests are generated.

The outage probability is illustrated in Fig. 5 as the function of the task generation rate λ. It can be seen that the outage probability increases with the task generation rate λ since MUs need more assistance from MEC servers when they have more task requests for a given local computation rate. As a result, the proportion of MUs transmitting tasks simultaneously increases, which brings more network interference and thus increases the outage probability. When $\lambda = 0$, no MU needs to transmit the task and the outage probability is equals 0. Moreover, the MUs with a larger local computation rate can support more task requests under the same communication performance. For instance, for outage probability $P_{\text{out}} = 0.14$, compared to $\mu_1 = 0.2$, up to 43% (or 91%)

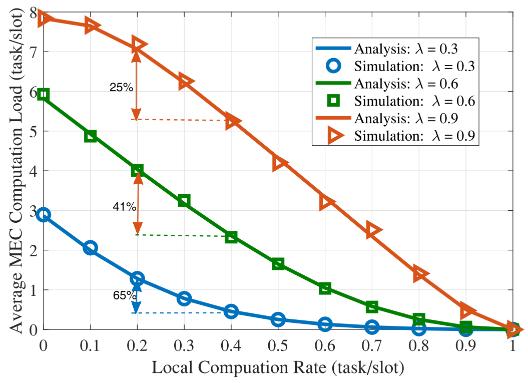

Fig. 6. Effect of local computation rate on average MEC computation load.

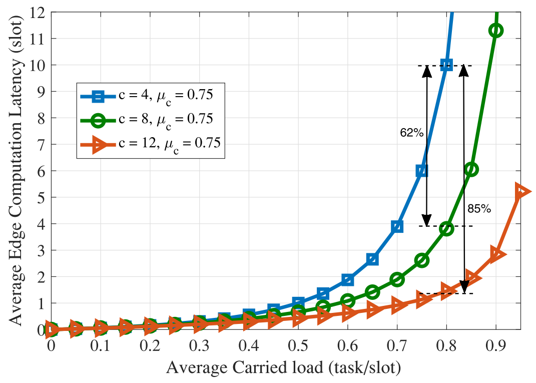

Fig. 8. Effect of average carried load on average edge computation latency.

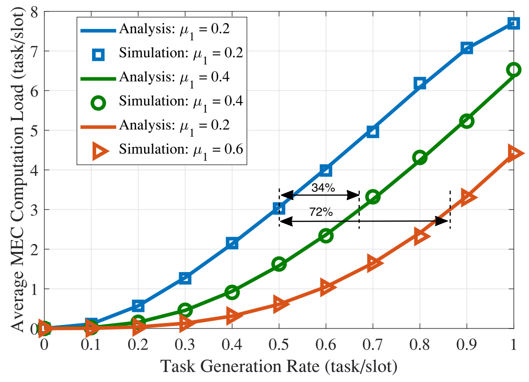

Fig. 7. Effect of task generation rate on average MEC computation load.

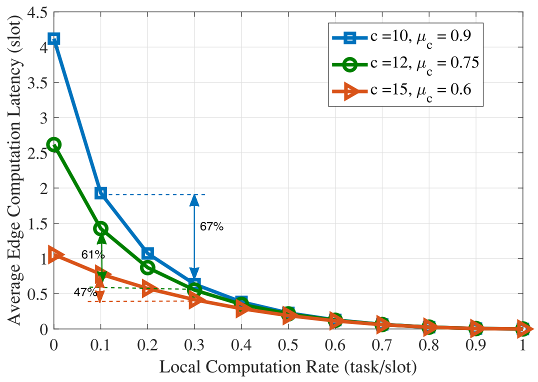

Fig. 9. Effect of local computation rate on average edge computation latency.

more requests can be supported at for MUs with $\mu_1 = 0.4$ (or $\mu_1 = 0.6$).

2) *Average MEC Computation Load:* The effects of local computation rate on the average MEC computation load are shown in Fig. 6. It can be observed that the average MEC computation load decreases with the local computation rate, and this shows that the LCC can effectively alleviate the computation pressure on the MEC server. For instance, when local computation rate is increased from 0.2 to 0.4, the MEC computation load can be reduced by 25% for $\lambda = 0.3$, 41% for $\lambda = 0.6$, and 65% for $\lambda = 0.9$. In addition, when MUs do not have LCC ,i.e., local computation rate $\mu_1 = 0$, the average MEC computation load is less than the total task requests in each cell $M\lambda$ (3 for $\lambda = 0.3$, 6 for $\lambda = 0.6$, and 9 for $\lambda = 0.9$) due to the existence of outage. Finally, as the local computation rate increases, MEC load will gradually decrease to 0 since MUs can compute all tasks locally.

The average MEC computation load is shown in Fig. 7 as a function of the task generation rate. From the figure, we can see that the average MEC computation increases with the task generation rate. The performance improvement for MEC systems brought by LCC in MUs can be shown by noting that for the MEC system with fixed MEC computation load, more task requests can be satisfied for MUs with a larger local computation rate. For instance, when the MEC computation load is equal to 3, compared with the MEC system with $\mu_1 = 0.2$, up to 34% (or 72%) more requests can be supported for MUs with $\mu_1 = 0.4$ (or $\mu_1 = 0.6$).

3) *Edge Computation Latency Performance:* In Fig. 8, the average edge computation latency is plotted versus the average carried load $\rho = (\Lambda/c\mu_c)$, which is defined the average carried load at each computation server in the MEC server. The number of computation servers $c = 4, 8, 12$ and all curves have the same $\mu_c = 0.75$. For a given number of computation servers, the average edge computation latency increases with the average carried load. Moreover, the slope for each curve increases with the average carried load, which implies that the latency increases significantly for a large load. In addition, we observe that for a given ρ at each computation server, when the buffer in the MEC server is shared for more computation servers, the MEC system can achieve buffer edge latency performance. It concludes that the shared resource policy has an advantage over the nonshared one.

The impact of local computation rate μ_1 on the average edge computation latency is illustrated in Fig. 9. We set the parameters for MEC server to be $\{(c, \mu_c)\} = \{(10, 0.9), (12, 0.75), (15, 0.6)\}$. Several observations are made as follows. First, the average edge computation latency decreases with local computation rate μ_1 since the LCC can reduce the MEC computation load as shown in Fig. 6. Second, the edge computation latency reduction brought by increasing LCC is more obvious when the local computation rate μ_1 is small. It is because when μ_1 is small, the MEC computation load is large. Meanwhile, for the MEC system with a large MEC computation load, the edge computation latency decreases significantly with the decrease of MEC computation

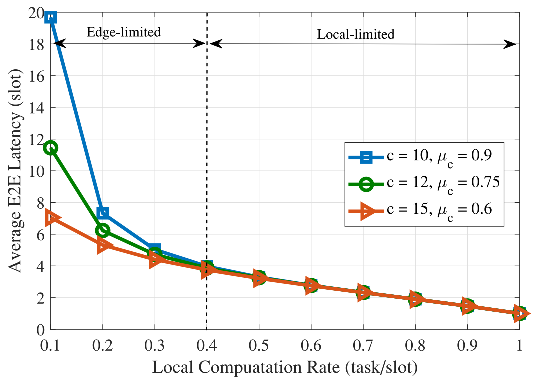

Fig. 10. Effect of local computation rate on average E2E latency.

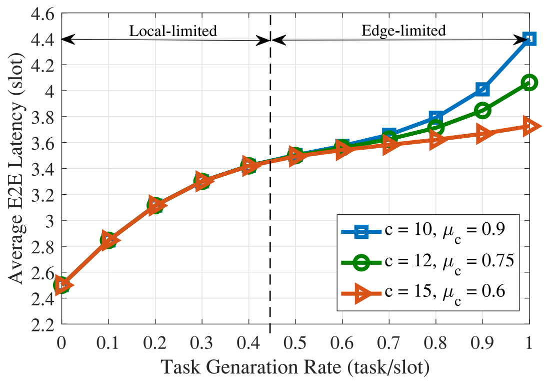

Fig. 11. Effect of task generation rate on average E2E latency.

load as shown in Fig. 8. Finally, for the MEC server with fixed computation resources ($c \cdot \mu_c$ = constant), when the computation resources are shared for more computation servers, the edge latency performance is better.

4) E2E Latency Performance: The curves of the average E2E latency versus the local computation rate μ_1 are plotted in Fig. 10. One can observe the average E2E latency decreases with the local computation rate. Note that improving LCC can decrease the E2E latency by reducing the local computation latency and the edge computation latency. These properties lead to the partitioning of the range of local computation rate into two network-operation regimes, edge-limited regime, and local-limited regime. For the MEC system under edge-limited regime, there exist significant edge computation latency performance differences in MEC systems with different server deployment solutions, and hence, the E2E latency performance is mainly limited by the edge performance. For the MEC system under local-limited regime, the edge computation latency in MEC systems with different server deployment solutions are close, and hence, the E2E latency performance is mainly limited by the local computation performance.

The impacts of the task generation rate on the average E2E latency are presented in Fig. 11. With the increasing of task generation rate, the average E2E latency increases due to the increase of local computation latency and edge computation latency. Moreover, the range of task generation rate is divided into different network-operation regimes, i.e., the edge-limited regime and local-limited regime. For the MEC system under low traffic load, the E2E latency is majority limited by local computation latency, while E2E latency performance is mainly limited by the edge performance under the heavy traffic load.

C. Discussion

In our paper, we studied the benefits incurred by LCC on the MEC system performance and the coupling between the communication and computing in stochastic MEC networks. From the communication point of view, LCC can be used to reduce the number of active MUs (i.e., the MU having task to offload) in each slot. Accordingly, the available bandwidth for each MU is increased and the interference is decreased, yielding the improvement in transmission reliability. From the computation point of view, with the utilization of LCC, some tasks are computed locally and thus, the computation load pressure of the MEC servers is relieved. Considering that the computing resources at the MEC servers are limited, the edge computing latency can be reduced. In addition, the E2E latency is reduced as computing efficiency at the MEC server and communication efficiency at the MU side are both improved. In addition, the useful guidelines are provided to avoid either the local computing resources or the edge computing resources being the performance bottleneck. The proposed model and analysis can be utilized in MEC networks that include one or more following practical features, namely, random node distribution, MUs with limited LCC, MEC servers with parallel computing power, dynamic traffic, and task retransmission.

Although the basic one-layers MEC network architecture and task offloading strategy are considered in this article, the modeling and analysis of the stochastic MEC networks can be can be extended to other MEC systems featuring random node distribution, dynamic traffic, and limited LCC at MUs. When considering the D2D offloading, the queue model at MUs should be adjusted since one another queue needs to be reserved to compute tasks from other MUs. Then, the queue analysis and network performance analysis approaches adopted in our work can be applied in this scenario. In addition, in order the obtain the optimal task offloading decision, one can first derive the key network performance according to our analysis methods, based on which the optimal computation offloading problem can be formulated and further solved.

In the future work, the current work can be extended in several directions. In this work, we consider the homogeneous task requests with identical priority, and the each MEC server has the same edge computing capability. Considering the heterogeneity of MEC networks in terms of not only the edge computing capability but also task priority is more challenging but is more practical. Next, studying a large-scale hierarchical MEC network comprising MUs, edge cloud and central cloud can further utilize the potential available computing resources despite the more complex network architecture. Moreover, introducing the recently proposed promising communication technique, such as coordinated multipoint (CoMP) transmission, network caching, and so on, into stochastic MEC networks will be another promising direction.

VI. Conclusion

In this work, we investigated the network performance in stochastic MEC-enabled wireless networks with dynamic traffic, where MUs have LCC to support task computation with idle local resources. To study the benefits incurred by LCC on MEC system performance, a stochastic MEC network was modeled featuring random node distribution, parallel computing in MEC server, task retransmission, and dynamic task request. Based on the joint analysis of communication and computing, we derived the average outage probability, average MEC computation load, and average E2E latency by applying the approaches of queuing theory and stochastic geometry. We showed that LCC in MUs can significantly improve the network performance in stochastic MEC networks, including communication performance and computation performance. Furthermore, useful guidelines were provided for MEC-network provisioning and planning to avoid LCC or the edge computing resources being a latency performance bottleneck.

Appendix A
Proof of Lemma 1

Recall that the number of MUs with task to transmit around each AP is Poisson distributed with mean $\widetilde{m}$ and the OFDMA is adopted for multiaccess. Accordingly, the average bandwidth for each MU to transmit task is given by $\mathbb{E}_B = B \cdot E\{(1/m)\}$. We denote mean of $(1/m)$ by $f(\widetilde{m})$ as $f(\widetilde{m}) = E\{(1/m)\} = \sum_{m=1}^{\infty}(1/m)\cdot[(e^{-\widetilde{m}}\widetilde{m}^m)/m!]$ and $f(\widetilde{m})$ is given by

$$\begin{aligned} f'(\widetilde{m}) &= \sum_{m=1}^{\infty} -\frac{e^{-\widetilde{m}}\widetilde{m}^m}{m\cdot m!} + \sum_{m=1}^{\infty}\frac{e^{-\widetilde{m}}\widetilde{m}^{m-1}}{m!} \\ &= f(\widetilde{m}) + \sum_{m=1}^{\infty}\frac{e^{-\widetilde{m}}\widetilde{m}^{m-1}}{m!}. \end{aligned} \tag{36}$$

According to the Taylor series for the exponential function, i.e., $e^x = \sum_{m=0}^{\infty}(x^m/m!)$, we can obtain

$$\sum_{m=1}^{\infty}\frac{e^{-\widetilde{m}}\widetilde{m}^{m-1}}{m!} = \frac{e^{-\widetilde{m}}}{\widetilde{m}}\cdot\left(\sum_{m=0}^{\infty}\frac{\widetilde{m}^m}{m!} - 1\right) = \frac{1-e^{-\widetilde{m}}}{\widetilde{m}}. \tag{37}$$

Therefore, $f(\widetilde{m})$ satisfies the first order linear nonhomogeneous differential equation

$$f'(\widetilde{m}) + f(\widetilde{m}) = \frac{1-e^{-\widetilde{m}}}{\widetilde{m}}. \tag{38}$$

Then, we have $f(\widetilde{m}) = e^{-\widetilde{m}}(\int[(1-e^{-\widetilde{m}})/\widetilde{m}]\,d\widetilde{m} + C_1) = e^{-\widetilde{m}}(\sum_{n=1}^{\infty}[(\widetilde{m}^n)/(n\cdot n!)] + C_2)$, where C_1 and C_2 are arbitrary real number. Considering that $f(0) = 0$, then we can obtain $f(\widetilde{m}) = e^{-\widetilde{m}}\cdot(\sum_{n=1}^{\infty}[(\widetilde{m}^n)/(n\cdot n!)])$. Therefore, the average bandwidth is $\mathbb{E}_B = B\cdot e^{-\widetilde{m}}\cdot(\sum_{n=1}^{\infty}[(\widetilde{m}^n)/(n\cdot n!)])$, and the proof is completed.

Appendix B
Proof of Theorem 1

According to the definition of the average outage probability, we have

$$\begin{aligned} &P_{\text{out}} \\ &= 1 - \mathbb{E}_r\left[\mathbb{P}\left\{\frac{h_0 r^{-\alpha}}{\sum_{x\in\Phi_c\setminus x_0}\sum_{y\in B^x} h_{y_x}||x+y||^{-\alpha}} > \beta|r\right\}\right] \\ &\overset{(a)}{=} 1 - \mathbb{E}_r\left[\mathbb{E}\left[\exp\left(-\beta r^{\alpha}\sum_{x\in\Phi_c\setminus x_0}\sum_{y\in B^x} h_{y_x}||x+y||^{-\alpha}\right)|r\right]\right] \\ &= 1 - \int_0^{\infty}\mathscr{L}_I(\beta r^{\alpha}) f_R(r)dr \end{aligned} \tag{39}$$

where (a) follows from the Rayleigh fading assumption. The Laplace transform of the aggregate interference from the intercluster interferers at the typical MEC AP is $\mathscr{L}_I(s)$

$$\begin{aligned} &= \mathbb{E}\left[\exp\left(-\beta r^{\alpha}\sum_{x\in\Phi_c\setminus x_0}\sum_{y\in B^x} h_{y_x}||x+y||^{-\alpha}\right)\right] \\ &= \mathbb{E}_{\Phi_c}\left[\prod_{x\in\Phi_c\setminus x_0}\mathbb{E}_{B^x}\left[\prod_{y\in B^x}\mathbb{E}_{h_{yx}}\left[\exp\left(-s h_{yx}||x+y||^{-\alpha}\right)\right]\right]\right] \\ &\overset{(a)}{=} \mathbb{E}_{\Phi_c}\left[\prod_{x\in\Phi_c\setminus x_0}\mathbb{E}_{B^x}\left[\prod_{y\in B^x}\frac{1}{1+s||x+y||^{-\alpha}}\right]\right] \\ &\overset{(b)}{=} \mathbb{E}_{\Phi_c}\left[\prod_{x\in\Phi_c\setminus x_0}\sum_{k=0}^{M}\left(\int_{\mathbb{R}^2}\frac{1}{1+s||x+y||^{-\alpha}} f_Y(y)dy\right)^k \right.\\ &\qquad\left. \times\ \mathbb{P}(K=k|K<M)\right] \\ &\overset{(c)}{=} \exp\left(-\lambda_c\int_{\mathbb{R}^2}\left(\int_{\mathbb{R}^2}\left(1-\frac{1}{1+s||x+y||^{-\alpha}} f_Y(y)dy\right)^k \right.\right.\\ &\qquad\left.\left.\times\ \frac{\widetilde{m}^k e^{-\widetilde{m}}}{k!\eta}\right)dx\right) \\ &\overset{(d)}{=} \exp\left(-2\pi\lambda_c\int_0^{+\infty}\left(1-\sum_{k=0}^{\widetilde{m}}(\rho(v))^k\frac{\widetilde{m}^k e^{-\widetilde{m}}}{k!}\right)v dv\right) \end{aligned}$$

where (a) follows from expectation over $h_{yx}\sim\exp(1)$, (b) is due to the expectation over number of interfering MUs per cluster, (c) is obtained by the probability generating functional of PPP [18], and (d) follows by converting from the Cartesian to polar coordinates where $\rho(v) = \int_0^{+\infty}(1/[1+\beta r^{\alpha}u^{-\alpha}])f_U(u|r)du$. Here, the proof is finished.

Appendix C
Proof of Proposition 1

Consider task **A** arriving during slot t and the subsequent buffer state $s(t) = (b_1, 0)$. If task **A** is processed by local computing, it indicates that **A** is scheduled to the local computation buffer in a certain slot. Assume that task **A** is scheduled in slot $t+i$, where $i > 0$. That is, the MU fails to transmit task **A** i times and no scheduling event occurs until the $t+i$ slot. The corresponding probability is $(1-\mu_2)^i(1-p_s)^{i-1}p_s$. Therefore, the conditional probability of task **A** being processed by local computing is

$$\begin{aligned} P_{c|1} &= \sum_{i=1}^{\infty}(1-\mu_2)^i(1-p_s)^{i-1}p_s \\ &= (1-\mu_2)p_s\sum_{i=1}^{\infty}\left[(1-\mu_2)(1-p_s)\right]^{i-1} \\ &= \frac{(1-\mu_2)p_s}{1-(1-\mu_2)(1-p_s)}. \end{aligned}$$

Likewise, as to the case that task **A** is processed by edge computing. Assume that task **A** is successfully transmitted to the MEC server at slot $t+i$, where $i>0$. It indicates that in the first $i-1$ slot, MU fails to transmit task **A** and no scheduling event occurs. Then, in the $t+i$ slot, task **A** is offloaded successfully. Accordingly, the conditional probability of task **A** being processed by edge computing is

$$\begin{aligned} P_{t|1} &= \sum_{i=1}^{\infty}(1-\mu_2)^{i-1}(1-p_s)^{i-1}\mu_2 \\ &= \mu_2\sum_{i=1}^{\infty}\big[(1-\mu_2)(1-p_s)\big]^{i-1} \\ &= \frac{\mu_2}{1-(1-\mu_2)(1-p_s)} \end{aligned}$$

and the proof is finished.

Appendix D
Proof of Proposition 2

We first analyze the conditional probability $P_t(i|s_2)$. The probability $P_t(i|1)$ indicates that task **A** is not successfully offloaded until the ith slot, and one obtains

$$P_t(i|1) = \big[(1-\mu_2)(1-p_s)\big]^{i-1}\cdot\mu_2. \tag{40}$$

Then, based on task departure process in buffer B_2, $P_t(i|2)$ is given by

$$P_t(i|2) = p_0^dP_t(i-1|2) + p_1^dP_t(i-1|1). \tag{41}$$

Likewise, as to $P_t(i|s_2)$, where $3\le s_2\le b_2$. One obtains

$$\begin{aligned} P_t(i|s_2) = p_0^dP_t(i-1|s_2) &+ p_1^dP_t(i-1|s_2-1) \\ &+ p_2^dP_t(i-1|s_2-2).\ \ 3\le s_2\le b_2. \end{aligned}\tag{42}$$

Then, we begin to derive the PGF $T_t(1,z)$. From (40) and the definition of PGF, we have $T_t(1,z)$

$$\begin{aligned} &= \sum_{i=1}^{\infty}\big[(1-\mu_2)(1-p_s)\big]^{i-1}\mu_2z^i \\ &= \mu_2z\sum_{i=0}^{\infty}\big[(1-\mu_2)(1-p_s)\big]^{i}z^i \\ &= \frac{\mu_2z}{1-(1-\mu_2)(1-p_s)z}. \end{aligned}\tag{43}$$

Then, from (41), we have $T_t(2,z)$

$$\begin{aligned} &= \sum_{i=2}^{\infty}p_0^dP_t(i-1|2)z^i + \sum_{i=2}^{\infty}p_1^dP_t(i-1|1)z^i \\ &\overset{(a)}{=} p_0^dz\sum_{i=3}^{\infty}P_t(i-1|2)z^{i-1} + p_1^dz\sum_{i=2}^{\infty}P_t(i-1|1)z^{i-1} \\ &\overset{(b)}{=} p_0^dz\sum_{j=2}^{\infty}P_t(j|2)z^j + p_1^dz\sum_{j=1}^{\infty}P_t(j|1)z^j \\ &= p_0^dzT_t(2,z) + p_1^dzT_t(1,z) \end{aligned}\tag{44}$$

where (a) is due to $P_t(1|2)=0$ and (b) comes from $j=i-1$. Then, we have

$$T_t(2,z) == \frac{p_1^d\mu_2z^2}{\big(1-p_0^dz\big)\big[1-(1-\mu_2)(1-p_s)z\big]}. \tag{45}$$

Likewise, as to $T_t(s_2,z)$ where $3\le s_2\le b_2$, we have

$$T_t(s_2,z) = \frac{p_1^dzT_t(s_2-1,z)+p_2^dzT_t(s_2-2,z)}{1-p_0^dz}. \tag{46}$$

The proof is completed.

References

[1] Y. Gu, C. Li, B. Xia, D. Xu, and Z. Chen, "Modeling and performance analysis of stochastic mobile edge computing wireless networks," in *Proc. IEEE VTC-Spring*, Apr. 2019, pp. 1–5.

[2] M. Patel *et al.*, "Mobile-edge computing introductory technical white paper," Mobile-Edge Comput. Ind. Initiative, ETSI, Sophia Antipolis, France, White Paper, 2014.

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

[4] K. Habak, M. Ammar, K. A. Harras, and E. Zegura, "Femto clouds: Leveraging mobile devices to provide cloud service at the edge," in *Proc. IEEE CLOUD*, Jun. 2015, pp. 9–16.

[5] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "Mobile edge computing: Survey and research outlook," 2017. [Online]. Available: arXiv:1701.01090.

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

[7] X. Chen, "Decentralized computation offloading game for mobile cloud computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 4, pp. 974–983, Apr. 2015.

[8] C. Zhang, H. Zhao, and S. Deng, "A density-based offloading strategy for IoT devices in edge computing systems," *IEEE Access*, vol. 6, pp. 73520–73530, 2018.

[9] L. Lei, Z. Zhong, K. Zheng, J. Chen, and H. Meng, "Challenges on wireless heterogeneous networks for mobile cloud computing," *IEEE Wireless Commun.*, vol. 20, no. 3, pp. 34–44, Jun. 2013.

[10] R. Urgaonkar, S. Wang, T. He, M. Zafer, K. Chan, and K. K. Leung, "Dynamic service migration and workload scheduling in edge-clouds," *Perform. Eval.*, vol. 91, pp. 205–228, Sep. 2015.

[11] P. Zhang, M. Zhou, and G. Fortino, "Security and trust issues in fog computing: A survey," *Future Gener. Comput. Systs.*, vol. 88, pp. 16–27, Nov. 2018.

[12] X. Huang, D. Ye, R. Yu, and L. Shu, "Securing parked vehicle assisted fog computing with blockchain and optimal smart contract design," *IEEE/CAA J. Automatica Sinica*, vol. 7, no. 2, pp. 426–441, Mar. 2020.

[13] D. Huang, P. Wang, and D. Niyato, "A dynamic offloading algorithm for mobile computing," *IEEE Trans. Wireless Commun.*, vol. 11, no. 6, pp. 1991–1995, Jun. 2012.

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

[15] T. Alfakih, M. M. Hassan, A. Gumaei, C. Savaglio, and G. Fortino, "Task offloading and resource allocation for mobile edge computing by deep reinforcement learning based on sarsa," *IEEE Access*, vol. 8, pp. 54074–54084, 2020.

[16] F. Jiang, K. Wang, L. Dong, C. Pan, and K. Yang, "Stacked autoencoder-based deep reinforcement learning for online resource scheduling in large-scale MEC networks," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 9278–9290, Oct. 2020.

[17] F. Jiang, K. Wang, L. Dong, C. Pan, W. Xu, and K. Yang, "Deep-learning-based joint resource scheduling algorithms for hybrid mec networks," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 6252–6265, Jul. 2020.

[18] H. ElSawy, E. Hossain, and M. Haenggi, "Stochastic geometry for modeling, analysis, and design of multi-tier and cognitive cellular wireless networks: A survey," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 3, pp. 996–1019, 3rd Quart., 2013.

[19] J. G. Andrews, F. Baccelli, and R. K. Ganti, "A tractable approach to coverage and rate in cellular networks," *IEEE Trans. Wireless Commun.*, vol. 59, no. 11, pp. 3122–3134, Nov. 2011.

[20] C. Lee and M. Haenggi, "Interference and outage in Poisson cognitive networks," *IEEE Trans. Wireless Commun.*, vol. 11, no. 4, pp. 1392–1401, Apr. 2012.

[21] C. Yang, Y. Yao, Z. Chen, and B. Xia, "Analysis on cache-enabled wireless heterogeneous networks," *IEEE Trans. Wireless Commun.*, vol. 15, no. 1, pp. 131–145, Jan. 2016.

[22] C. Saha, M. Afshang, and H. S. Dhillon, "3GPP-inspired HetNet model using Poisson cluster process: Sum-product functionals and downlink coverage," *IEEE Trans. Commun.*, vol. 66, no. 5, pp. 2219–2234, May 2018.
[23] E. Access, *Further Advancements for e-Utra Physical Layer Aspects*, 3GPP Standard TS 36.814, 2010.
[24] M. Haenggi, *Stochastic Geometry for Wireless Networks*. Cambridge, U.K.: Cambridge Univ. Press, 2012.
[25] R. K. Ganti and M. Haenggi, "Interference and outage in clustered wireless ad hoc networks," *IEEE Trans. Inf Theory*, vol. 55, no. 9, pp. 4067–4086, Sep. 2009.
[26] V. Suryaprakash, J. Moller, and G. Fettweis, "On the modeling and analysis of heterogeneous radio access networks using a Poisson cluster process," *IEEE Trans. Wireless Commun.*, vol. 14, no. 2, pp. 1035–1047, Feb. 2015.
[27] M. Afshang, H. S. Dhillon, and P. H. J. Chong, "Fundamentals of cluster-centric content placement in cache-enabled device-to-device networks," *IEEE Trans. Commun.*, vol. 64, no. 6, pp. 2511–2526, Jun. 2016.
[28] C. Park and J. Lee, "Mobile edge computing-enabled heterogeneous networks," 2018. [Online]. Available: arXiv:1804.07756.
[29] A. Al-Shuwaili and A. Lawey, "Achieving low-latency mobile edge computing by uplink and downlink decoupled access in HetNets," 2018. [Online]. Available: arXiv:1809.04717.
[30] H. Lee and J. Lee, "Task offloading in heterogeneous mobile cloud computing: Modeling, analysis, and cloudlet deployment," *IEEE Access*, vol. 6, pp. 14908–14925, 2018.
[31] D. Lopez-Perez, A. Valcarce, G. de la Roche, and J. Zhang, "OFDMA femtocells: A roadmap on interference avoidance," *IEEE Commun. Mag.*, vol. 47, no. 9, pp. 41–48, Sep. 2009.
[32] A. Gravey and G. Hebuterne, "Simultaneity in discrete-time single server queues with bernoulli inputs," *Perform. Eval.*, vol. 14, no. 2, pp. 123–131, 1992.
[33] K. Li, "A game theoretic approach to computation offloading strategy optimization for non-cooperative users in mobile edge computing," *IEEE Trans. Sustain. Comput.*, early access, Sep. 5, 2018, doi: 10.1109/TSUSC.2018.2868655.
[34] H. Takagi, *Queueing Analysis: A Foundation of Performance Evaluation*, vol. 1. Amsterdam, The Netherlands: North-Holland, 1991.
[35] D. Q. Qiao and M. C. Gursoy, "Buffer-aided relay systems under delay constraints: Potentials and challenges," *IEEE Commun. Mag.*, vol. 55, no. 9, pp. 168–174, Sep. 2017.
[36] J. Li and Y. Q. Zhao, "Resequencing analysis of stop-and-wait arq for parallel multichannel communications," *IEEE/ACM Trans. Netw.*, vol. 17, no. 3, pp. 817–830, Jun. 2009.
[37] M. Zukerman, "Introduction to queueing theory and stochastic teletraffic models," 2013. [Online]. Available: arXiv:1307.2968.
[38] M. Afshang, H. S. Dhillon, and P. H. Joo Chong, "Modeling and performance analysis of clustered device-to-device networks," *IEEE Trans. Wireless Commun.*, vol. 15, no. 7, pp. 4957–4972, Jul. 2016.
[39] H. Bruneel, "A general model for the behaviour of infinite buffers with periodic service opportunities," *Eur. J. Oper. Res.*, vol. 16, no. 1, pp. 98–106, 1984.
[40] P. Gao, S. Wittevrongel, and H. Bruneel, "Discrete-time multiserver queues with geometric service times," *Comput. Oper. Res.*, vol. 31, no. 1, pp. 81–99, 2004.
[41] A. S. Alfa, *Queueing Theory for Telecommunications: Discrete Time Modelling of a Single Node System*. New York, NY, USA: Springer, 2010.

Yixiao Gu received the B.Eng. degree in communication engineering from Huazhong University of Science and Technology, Wuhan, China, in 2017. He is currently pursuing the Ph.D. degree with the Institute of Wireless Communications Technology, Department of Electronic Engineering, Shanghai Jiao Tong University, Shanghai, China.

His current research interests include computing communication, caching networks, and radio network architecture.

Yao Yao (Senior Member, IEEE) received the B.Eng. degree in computer science and the M.Eng. degree in circuits and systems from the University of Science and Technology of China, Hefei, China, in 1997 and 2000, respectively, and the Ph.D. degree in electrical engineering from the University of Hong Kong, Hong Kong, in 2004.

From 2005 to 2020, she was a Senior Research Scientist with Huawei Technologies Company Ltd., Shanghai, China, on 3G wideband CDMA Systems. Since 2020, she has been a Senior Engineer with the Department of Electronic Engineering, Shanghai Jiao Tong University, Shanghai. Her research interests include synchronization, radio resource management, and mobile network evolution.

Dr. Yao was a recipient of the 8th IEEE ComSoc Asia–Pacific Outstanding Paper Award.

Cheng Li received the B.Eng. degree in communication engineering from Northwestern Polytechnical University, Xi'an, China, in 2015, and the Ph.D. degree in information and communication engineering from Shanghai Jiao Tong University, Shanghai, China, in 2020.

Since 2020, he has been a Senior Research with Huawei Technologies Company Ltd., Shanghai. From 2017 to 2018, he was a Visiting Scholar with the Department of Electrical and Computer Engineering, Rice University, Houston, TX, USA. His research interests include full-duplex networks, joint communication and radar systems, and performance analysis and optimization.

Bin Xia (Senior Member, IEEE) received the B.Eng. degree in electrical engineering and the M.Eng. degree in information and communication engineering from the University of Science and Technology of China, Hefei, China, in 1997 and 2000, respectively, and the Ph.D. degree in electrical engineering from the University of Hong Kong, Hong Kong, in 2004.

From 1995 to 2000, he was with the Personal Communication and Spread Spectrum Laboratory, University of Science and Technology of China, as a Research Engineer involved in the development of CDMA communication systems based on IS-95 and UMTS standards. From 1999 to 2001, he worked as a Department Manager with UTStarcom Inc., Hong Kong, on WCDMA Systems. From 2004 to 2005, he was working as a System Engineer with Alcatel Shanghai Bell Company Ltd., Shanghai, China, on UMTS and WiMAX Systems. From 2005 to 2012, he was working as a Senior Research Scientist, a Project Manager, and a Director with Huawei Technologies Company Ltd., Shanghai, on Beyond 3G Research and 4G LTE R&D. Since 2012, he has been a Professor with the Department of Electronic Engineering, Shanghai Jiao Tong University, Shanghai. His research interests are in the areas of coded modulation, MIMO, OFDM, crosslayer design, and radio network architecture.

Dr. Xia was a recipient of the IEEE Asia–Pacific Outstanding Paper Award in 2019. He served as the Publicity Chair of the IEEE WCNC in 2013, the Workshop Co-Chair of the IEEE ICC in 2015, and the Executive Co-Chair of the IEEE ICC in 2019.

Dingjie Xu received the B.Eng. degree in communication engineering and the M.Eng. degree in electronic and information engineering from Shanghai Jiao Tong University, Shanghai, China, in 2015 and 2018, respectively.

His research interests include stochastic geometry, relay transmission, and caching networks.

Chaoxian Zhang received the B.Eng. degree in electronic information engineering and the M.Eng. degree in signals and information processing from Xiamen University, Xiamen, China, in 2005 and 2008, respectively.

From 2007 to 2011, he worked as a Wireless Communication Algorithm Engineer with Huawei Technologies Company Ltd., Shanghai, China, on 4G LTE system design and performance evaluation. He is currently a Lecturer with the School of Information Science and Technology, Xiamen University Tan Kah Kee College, Xiamen. His research interests include next-generation wireless communication system design, performance evaluation, and distributed parallel simulation methodologies.