Performance Analysis of Mobile Cloud Computing with Bursty Demand: A Tandem Queue Model

Bo Sun,* *Member, IEEE*, Yuxuan Jiang,* *Member, IEEE*, Yuan Wu, *Senior Member, IEEE*, Qiang Ye, *Member, IEEE*, and Danny H.K. Tsang, *Fellow, IEEE*

***Abstract*—Resource-constrained end devices can offload computation to backend clouds. The stochastic wireless channel that an end device is connected to can introduce bursty computation demand to the cloud. Specifically, under good channel conditions, a device can transmit more data to the cloud, which consequently yields higher instantaneous computation demand. Conversely, poor channel conditions can result in lower instantaneous demand. The performance indicator for such a mobile cloud computing system is the average of the response time, which is the time span from the arrival of the computation demand at the backend cloud instance to the completion of its execution. The question we target in this paper is how resources should be provisioned for the backend cloud instance to address this bursty computation demand and guarantee a desired quality-of-service (QoS), namely, a user-specified average response time. To answer this question, we model the mobile cloud computing system as two tandem queues. We analyze this queueing network using the fluid flow analysis framework, and derive the analytical relationship between the required resource capacity at the backend cloud instance and the desired QoS, given the workload generation process at the end device and the wireless channel conditions. Having obtained the required resource capacity for a desired QoS, we then determine whether it is economical to provision this resource capacity by subscribing to the traditional static instance or the recently introduced burstable instance offered by public cloud providers. Finally, trace-driven simulations validate our theoretical results.**

***Index Terms*—Mobile cloud computing, offloading, fluid flow analysis**

*B. Sun and Y. Jiang contributed equally to this work.

Manuscript received December 12, 2021; revised March 27, 2022 and April 29, 2022; accepted May 1, 2022. This work was supported in part by the National Sciences and Engineering Research Council of Canada (NSERC) under Grant No. RGPIN-2017-05853; in part by FDCT-MOST Joint Project under Grant 0066/2019/AMJ; and in part by Science and Technology Development Fund of Macao SAR under Grant 0162/2019/A3.

A preliminary version of this work was presented as a poster in the IEEE International Conference on Distributed Computing Systems (ICDCS), Singapore, November 29–December 1, 2020 [1].

B. Sun is with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong SAR, China (e-mail: bsunaa@connect.ust.hk).

Y. Jiang was with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong SAR, China. He is now with the Faculty of Computer Science, Dalhousie University, Halifax, NS B3H 4R2, Canada (e-mail: jiang@dal.ca).

Y. Wu is with The State Key Lab of Internet of Things for Smart City, and also with the Department of Computer and Information Science, The University of Macau, Taipa, Macao SAR, China (e-mail: yuanwu@um.edu.mo).

Q. Ye is with the Faculty of Computer Science, Dalhousie University, Halifax, NS B3H 4R2, Canada (e-mail: qye@cs.dal.ca).

D.H.K. Tsang is with the Internet of Things Trust, The Hong Kong University of Science and Technology (Guangzhou), Guangzhou, Guangdong 511400, China, and also with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong SAR, China (e-mail: eetsang@ust.hk).

I. Introduction

FOR many computation-intensive Internet of Things (IoT) applications, the workload generation rate at the resource-constrained end device is usually much higher than the local computation capability. Mobile cloud computing has thus been proposed to offload the extra workload that cannot be tackled locally to the backend cloud for execution. The performance of mobile cloud computing depends on how fast the workload can be processed (i.e., the latency from the workload generated at the end device to the completion of its execution in the cloud) [2]–[13]. This latency consists of two parts–the transmission time from the end device to the backend cloud and the response time from the arrival of the workload at the cloud instance to the completion of its execution.[1] The transmission time, however, depends on the physical communication channel between the end device and the cloud, which is usually uncontrollable. Therefore, the key question that we target in this paper is *how to provision resources to the backend cloud instance to guarantee a desired average response time*.

Making a resource provisioning decision for the backend cloud is a non-trivial task. Typically, end devices are connected to the network via wireless links. As shown in Figure 1, the stochastic evolution of the wireless channels results in time-varying throughput and consequently brings *bursty computation demand* to the backend cloud.[2] Specifically, when the wireless channel is in a good condition with high throughput, an end device is able to transmit more data to the backend cloud instance for execution, leading to higher instantaneous computation demand. In contrast, a lower instantaneous computation demand is received by the cloud instance under a poor channel condition. Our resource provisioning decision should be able to meet this bursty computation demand that arrives at the cloud through the time-varying wireless channel.

The key in making the resource provisioning decision is to analytically model the relationship between the provisioned resources at the cloud and the resulting average response time given the parameters of the system environment, which involves the wireless channel parameters and the workload

[1]The time to transmit the execution result from the cloud back to the edge device is usually negligible, and thus not included as a performance metric. This is because the execution result typically has a much smaller size than the offloaded raw data [4], [9], [14]–[16].

[2]Since the workload offloaded from the end device to the cloud instance can also be viewed as the computation demand to be fulfilled by the cloud, we use the words "workload" and "demand" interchangeably in this paper.

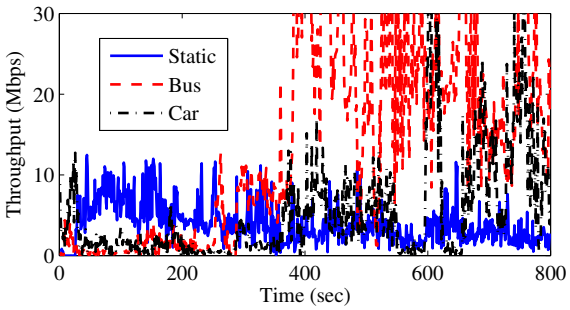


Fig. 1: LTE throughput under different mobility patterns [17]. It can be observed that the throughput experiences vast variations for moving end devices, such as for a moving bus or moving car. When the end device is static, the throughput for this device still varies from 1 Mbps to 12 Mbps over time.

generation process at the end device.[3] The mobile cloud computing system can be abstracted as two tandem queues, where the first queue models the offloading process from the end device to the cloud instance, and the second queue models the workload execution procedure in the cloud. The major technical challenge in analyzing this tandem queue network lies in quantifying the burstiness of the computation demand received by the cloud instance after it passes through a time-varying wireless channel. Note that this tandem queue network cannot be analyzed using the conventional Jackson network theory [18] with off-the-shelf analytical results. This is because the *service rate* of the first queue, which captures the throughput of the wireless channel, is modulated by the time-varying channel conditions (which can be modeled as a Markov chain). Therefore, we must explicitly characterize the output process of the first queue, which is equivalently the input process of the second queue, before delving into a detailed analysis of the second queue. However, most existing works on Markov-modulated queues (e.g., [19]–[21]) can only derive the output process for queues with a Markov-modulated *arrival rate* and a constant service rate. To the best of our knowledge, while a few papers (e.g., [22]) have studied the queues with a Markov-modulated *service rate*, they only focus on deriving the stationary distribution instead of the output process (see Section II-A for details).

In this paper, we explicitly characterize the output process of a queue with a Markov-modulated *service rate* to bridge the research gap. Based on this result, we further conduct a comprehensive analysis of the tandem queue model for mobile cloud computing and derive an analytical performance model between the required resource capacity at the backend cloud instance and the desired average response time.

Our performance model can help to not only calculate the required resource capacity for the backend cloud instance, but also make appropriate cloud resource provisioning decisions with the least monetary cost. Once the required resource capacity is known, we can calculate its mean utilization with our performance model. The resource capacity and its mean utilization act as two critical indicators in selecting the most appropriate instance type and configuration from the instance offerings of public clouds. In general, there are two types of instances in state-of-the-art offerings of public cloud providers (such as Amazon EC2 and Microsoft Azure): traditional static instances and the recently introduced burstable instances [23]–[25]. When a user subscribes to a traditional static instance, (s)he is offered a dedicated resource capacity (e.g., two vCPUs) for his/her instance, which can be freely used up at all times. The unit price of a static instance depends only on the provisioned resource capacity. In contrast, if a user subscribes to a burstable instance, (s)he is also offered a certain resource capacity (e.g., two vCPUs), but will be charged according to both this resource capacity and its actual utilization. If the average actual utilization is lower than or equal to a certain baseline, which is pre-announced by the cloud provider, the user will pay a fixed price, which is much lower than the price of a static instance with the same resource capacity. However, if the user's average utilization exceeds the baseline, (s)he will need to pay an additional charge proportional to the surplus. If this surplus is very large, the ultimate combined charge for the burstable instance can be much higher than that for a static instance with the same resource capacity. Fortunately, our performance model is able to calculate the mean utilization of the capacity. By further formulating the pricing strategies from the current instance offerings of public cloud providers and combining them with our performance model, we then determine whether static instances or burstable instances are more economical to be deployed at the backend cloud and guarantee a desired quality-of-service (QoS) requirements, namely a desired average response time.

To sum up, our contributions in this paper are as follows.

- We propose a tandem queue model for a practical mobile cloud computing system. The first queue models the offloading process from the end device through the wireless channel, and the second queue models the computation at the backend cloud instance. The key technical challenge in analyzing this tandem queue model lies in tackling the first queue's Markov-modulated service rate, which models the wireless channel in the computation offloading. To this end, in Section III, we prove that the output process of the first queue is a semi-Markov process (SMP) and derive the statistical characteristics of this SMP. To the best of our knowledge, we are the first to conduct a theoretical analysis of the output process of a queue with a Markov-modulated service rate. To gain more insights into our analytical results on the SMP, we then use the Gilbert-Elliott channel, a widely adopted channel model, as an example to numerically showcase the output process of the first queue.
- Having obtained the output process of the first queue in the tandem queue model, we are ready to derive the analytical relationship between the resource capacity at the backend cloud instance and the resulting average response time (i.e., to derive the analytical performance model) from the second queue. In Section IV, we first propose a simple yet accurate approximation for the first queue's output process as a continuous-time Markov

[3]In this paper, we treat one end device as a user and assign him/her a backend cloud instance. We thus use the words "end device" and "user" interchangeably.

chain (CTMC). The analytical performance model is then derived using the fluid flow analysis framework.

- Given a desired average response time, our performance model is able to derive the corresponding resource capacity required at the backend cloud instance, and further calculate the actual utilization of this capacity. According to these two resource indicators, i.e., the required resource capacity and its mean utilization, we can determine the instance type and configuration to choose from current instance offerings of public clouds in order to provision the required resource capacity as a backend cloud instance with the minimum monetary cost. In Section V, we model the charges of static and burstable instances as functions of the two resource indicators, based on which we then decide on the instance type.
- In Section VI, we validate our theoretical results by trace-driven simulations. Numerical results show that our performance model can accurately approximate the average response time. In terms of the choice of instance type, burstable instances are preferred to save users' monetary costs when (*i*) the wireless channel changes very rapidly, with the demand being generated in a rather bursty manner, and (*ii*) the QoS requirement (i.e., the desired average response time) is more stringent.

The remainder of this paper is organized as follows. We first survey the literature in Section II-A. The core idea of fluid flow analysis framework, the method that we will use to analyze our tandem queue model, is briefly introduced in Section II-B before we delve into our technical analysis. We present our tandem queue model and derive the output process of the first queue in Section III. We then derive the relationship between the resource capacity at the backend cloud instance and the average response time in Section IV, which completes our analytical performance model. The decision on instance type is made in Section V. We validate our analytical results by trace-driven simulations in Section VI. Finally, Section VII concludes the paper.

II. Background and Related Work

In this section, we first survey the literature, and then introduce the fluid flow analysis framework for queueing network analysis.

A. Related Work

Computation offloading [26], [27] is an important and widely studied topic in IoT and future wireless networks [28]. However, a limited number of existing works have addressed how to evaluate the response time of the offloaded computation demand. Most of the existing works (e.g., [2]–[8]) focus on offloading decisions about when and how much of the workload is to be offloaded from end devices. They simplify the response time as a positive constant [2]–[6] or simply zero [7]. Other works use oversimplified and unrealistic assumptions and models to address the average response time. Following the Lyapunov optimization framework, for example, Mao *et al.* [9] take the response time into account, but without providing an analytical expression nor guaranteeing a required response time. Other works [10]–[12] derive the analytical response time by assuming that workload can be offloaded from the end device through a perfect wireless channel with constant throughput. However, this assumption is inconsistent with the situations in real-world systems, as we have stated in Section I and shown with Figure 1. Therefore, to understand the performance of practical mobile cloud computing systems for better planning and real-time decision-making, this paper aims to analytically model the average response time by considering a practical finite-state CTMC wireless channel model.

Technically, a key contribution of our work is to derive the output process of a Markov-modulated fluid model (MMFM). In the literature, there exists a large body of works (e.g., [19], [20], [29], [30]) on the performance analysis of MMFM. However, most focus on the MMFM with a Markov-modulated *arrival rate* and a constant service rate. Our paper is different from this line of works in that we focus on an MMFM with a Markov-modulated *service rate*, where this service rate changes according to an external CTMC. Although the stationary distribution of this queueing model can be derived from the classical fluid flow analysis (see Section II-B for details), how to derive its output process still remains largely unexplored in the literature. This is because the output process of the MMFM is known to be non-Markovian, making its application to fluid queueing networks significantly challenging.

To the best of our knowledge, the most relevant works to ours are [19] and [20], which essentially study the output process of the MMFM with a Markov-modulated *arrival rate* and a constant service rate. These works approximate the output process as a CTMC by lumping all non-Markovian states with the same fluid drift into one state and approximating the sojourn time by an exponential variable. This approximation is only valid when the stationary probability of this lumped state is small, which is the case in [19] and [20]. Different from these works, however, our MMFM has a constant arrival rate and a Markov-modulated *service rate*. Therefore, the non-Markovian states are associated with different fluid drifts. We should also note that the sojourn times of the non-Markovian states are not negligible in our model. As a result, the existing analytical results cannot be directly applied to our problem.

Rather than the fluid model, Mahabhashyam *et al.* [22] analyze the average response time (but not the output process) of a queue with a Markov-modulated service rate using the matrix geometric method (MGM). Huang *et al.* [31] further extend the analysis using the service-beginning probability proposed in [22] and derive closed-form expressions of the average response time for the same queuing model with a finite buffer. However, deriving the output process of a queue is far more complicated than deriving the average response time. Therefore, the theoretical results in [22] and [31] cannot be simply extended to derive the corresponding output process. In summary, to the best of our knowledge, no prior work has derived the output process of a queue with a Markov-modulated *service rate*.

After a resource capacity is derived for a desired QoS from our performance model, the next step in making the resource provisioning decision is to determine how to provision this

resource capacity at the backend cloud, i.e., which type of instance to choose from the offerings of public cloud providers. Besides the traditional static instance, our work also considers burstable instances, an instance type recently introduced by public cloud providers. Burstable instances have been an emerging research topic over the past few years, and the majority of existing works on burstable instances are empirical reports on their use cases [32]–[36]. The theoretical work on burstable instances most relevant to ours is by Jiang *et al.* [25], in which the performance of burstable instances in an Infrastructure-as-a-Service (IaaS) cloud is analytically modeled. Based on the performance model, an optimal pricing scheme is then worked out for public cloud providers. This work considers the perspective of public cloud providers in setting instance configuration and pricing schemes that can maximize their revenues. Given the pricing schemes announced by the public cloud providers, our work considers the perspective of an individual cloud user. We determine the best selection of instance type and configuration according to the user's required average response time for mobile cloud computing. To the best of our knowledge, no work in the literature has done this before.

B. Preliminaries on Fluid Flow Analysis

Traditional queueing theory [37] studies a system of one or more servers at which individual customers arrive and receive services from the servers. A fluid queue model [30] approximates the traditional discrete queue model by treating the customers as a continuous (i.e., highly fine-grained) entity, which is referred to as a "fluid." As shown in Figure 2, the fluid flows in and out of a fluid reservoir (i.e., the buffer of the queue) at certain input and output rates, respectively. The state of a fluid queue is defined as the amount of buffered fluid at a particular time instance and the state dynamics depends on the net input rate (i.e., the instantaneous fluid input rate minus output rate) of the fluid queue. In this paper, we particularly focus on an MMFM, in which the net input rate depends on an external Markov process, and fluid flow analysis is a technique to evaluate the stationary distribution of the MMFM. When conducting fluid flow analysis, we usually consider a fluid queue with an infinite-size buffer, which can temporally store fluid that flows through it. (We include a discussion on how to remove the assumption of an infinite-size buffer for real-world systems at the end of this sub-section.) Let X_t denote the amount of buffered fluid at time t, $X := \{X_t\}_{t\geq 0}$ denote the fluid process, and $S := \{S_t\}_{t\geq 0}$ denote a CTMC with state space $\mathcal{S} := \{1, 2 \ldots, N\}$.

An MMFM is defined as the joint process of fluid process X and CTMC S, i.e., $\{(X_t, S_t)\}_{t\geq 0}$. The dynamics of S is independent of X, and the evolution of S_t is totally governed by its intensity matrix $Q := \{Q_{ij}\}_{i,j\in\mathcal{S}}$, where Q_{ij} is the transition rate from state i to state j when $j \in \mathcal{S}\setminus i$ and $Q_{ii} = -\sum_{j\in\mathcal{S}\setminus i} Q_{ij}$. Let $\boldsymbol{\pi} = [\pi_1, \ldots, \pi_N]$ denote the stationary distribution of S. $\boldsymbol{\pi}$ can be derived by solving a linear system $\boldsymbol{\pi}\cdot Q = \mathbf{0}$ and $\boldsymbol{\pi}\cdot\mathbf{1}_{N\times 1} = 1$, where $\mathbf{1}_{N\times 1}$ is an N-dimensional column vector with all elements equal to 1.

The dynamics of the fluid X is modulated by S. In particular, the net input rate of the fluid queue depends on the state of the CTMC. Let d_i denote the net input rate when the CTMC is in state i. When $d_i > 0$ ($d_i < 0$), we say that the fluid queue is with an upward (downward) fluid drift. The fluid dynamics is governed by the net input rate as long as the fluid change is physically feasible, i.e.,

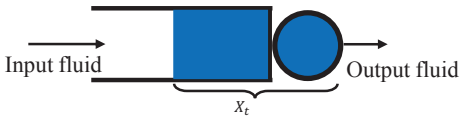


Fig. 2: The Markov-modulated fluid model (MMFM). The continuous fluid flows into the buffer. The server can be considered as a tap that allows the fluid to flow out. The amount of buffered fluid at time t is denoted by X_t. The dynamics of the fluid, i.e., dX_t/dt, depends on an external Markov process.

$$\frac{dX_t}{dt} = \begin{cases} [d_i]^+ & \text{if } S_t = i, X_t = 0, \\ d_i & \text{if } S_t = i, X_t > 0, \end{cases} \tag{1}$$

where $[d_i]^+ := \max\{d_i, 0\}$. Therefore, an MMFM can be fully characterized by the intensity matrix Q of the CTMC S and the fluid drift vector $\boldsymbol{d} := [d_1, \ldots, d_N]$. Note that the buffer is in fact a temporary storage of the fluid. To ensure the stability of the fluid queue, the amount of buffered fluid should be prevented from growing towards infinity. The stability condition [30] of the fluid queue is thus

$$\sum_{i\in\mathcal{S}} \pi_i d_i < 0, \tag{2}$$

which means the long-term average input rate should be smaller than the corresponding output rate. When condition (2) is satisfied, the bi-variant process $\{(X_t, S_t)\}_{t\geq 0}$ converges to a stationary joint distribution

$$F_i(x) = \lim_{t\to\infty} \mathsf{P}\left(X_t \leq x, S_t = i\right), \quad i \in \mathcal{S}, x \geq 0, \tag{3}$$

which is the probability that the CTMC is in state i and the fluid is no larger than x. Assuming all drifts are non-zero, i.e., $d_i \neq 0, \forall i \in \mathcal{S}$, the joint distribution satisfies the following first-order differential equations [38]:

$$\frac{d\boldsymbol{F}(x)}{dx} = \boldsymbol{F}(x) Q D^{-1}, \tag{4}$$

where $\boldsymbol{F}(x) := [F_1(x), \ldots, F_N(x)]$ and $D := \mathrm{diag}(\boldsymbol{d})$. It has been known that the solution of Equation (4) can be expressed as a sum of exponential terms, i.e.,

$$\boldsymbol{F}(x) = \sum_{k=1}^{N} a_k e^{\xi_k x} \boldsymbol{v}_k, \tag{5}$$

where ξ_k and $\boldsymbol{v}_k := [v_{k1}, \ldots, v_{kN}]$ are the k-th eigenvalue and eigenvector of QD^{-1}, respectively, and $\{a_k\}_{k=1,\ldots,N}$ are coefficients to be determined by the boundary conditions. (The boundary conditions will be discussed later in this sub-section.) The state space $\mathcal{S}$ is partitioned into two separate sets, $\mathcal{S}_D := \{i \in \mathcal{S} | d_i < 0\}$ and $\mathcal{S}_U := \{i \in \mathcal{S} | d_i > 0\}$, which consist of the states with downward and upward fluid drifts, respectively. When the stability condition (2) is satisfied, the numbers of negative and positive eigenvalues of QD^{-1}

are $|\mathcal{S}_U|$ and $|\mathcal{S}_D|-1$, respectively [39]. To ensure the joint distribution $\boldsymbol{F}(x)$ is bounded, the coefficients that correspond to positive eigenvalues must be zero, i.e., $a_k=0$ for k, where $\xi_k>0$. Also note that 0 and $\boldsymbol{\pi}$ are a pair of one eigenvalue and one eigenvector of QD^{-1}, and their corresponding coefficient is 1 since $\lim_{x\to\infty}\boldsymbol{F}(x)=\boldsymbol{\pi}$ must hold. As a result, only $|\mathcal{S}_U|$ coefficients remain to be determined, and the joint distribution can be simplified as

$$\boldsymbol{F}(x)=\boldsymbol{\pi}+\sum_{k=1}^{|\mathcal{S}_U|}a_k e^{\xi_k x}\boldsymbol{v}_k,\quad x\geq 0. \tag{6}$$

Note that $F_i(0)$, the probability that the buffer is empty when the CTMC is in state i, is zero when the fluid drift is positive. We can thus get $|\mathcal{S}_U|$ boundary conditions:

$$F_i(0)=0,\quad i\in\mathcal{S}_U. \tag{7}$$

The remaining coefficients $\{a_k\}_{k=1,\ldots,|\mathcal{S}_U|}$ can be determined by combining Equations (6) and (7). In this way, the joint stationary distribution can be fully obtained, based on which we can further derive the statistical characteristics (e.g., the average queue length) of the fluid queue.

Admittedly, it is usually impractical to have an infinite-size buffer in real-world systems. However, it is always possible to choose a sufficiently large buffer size B under the stability condition (2), where the probability that the fluid is beyond B is negligible. To see this, note that the stationary distribution (6) is the sum of exponential terms. For a large x, $\boldsymbol{F}(x)$ will be dominated by the exponential term with the largest (negative) eigenvalue, which we denote by ξ_1. As a result, the overflow probability of the fluid beyond x can be approximated as

$$\bar{F}(x)\approx 1-\sum_{n=1}^{N}\pi_n-a_1\sum_{n=1}^{N}\pi_n v_{1n}e^{\xi_1 x}=A_1 e^{\xi_1 x}, \tag{8}$$

where $A_1=a_1\sum_{n=1}^{N}\pi_n v_{1n}$. Since $\bar{F}(x)$ decreases exponentially with regard to x, it is always possible to choose a B value to ensure that $\bar{F}(B)$ is negligible. Thus, the overflow probability of a fluid queue with a buffer size B can be approximated by $\bar{F}(B)$ of the infinite queue model. In this regard, we can focus on the fluid flow analysis for the infinite-size buffer queue and the results can well approximate a practical system with a buffer size B. In other words, the fluid flow analysis also provides a way to derive the size of the buffer to guarantee a small overflow probability for real-world systems (e.g., choosing buffer size B such that $\bar{F}(B)$ is in the range of 10^{-8} to 10^{-12}).

III. Tandem Queue Model

We propose a tandem queue model for a practical mobile cloud computing system, as shown in Figure 3. The first queue, named the local data queue, models the offloading process from the mobile user to a backend cloud instance. The input workload can be either computed locally (i.e., local computing) or offloaded via a wireless channel to cloud instances for processing (i.e., cloud computing). The workload that cannot be locally computed or immediately offloaded is buffered in the local data queue. After going through a wireless channel, the offloaded demand[4] from the user enters the second queue, named the remote data queue, to be processed by a backend cloud instance. Both queues in our tandem queue model execute the workload in a first-in-first-out (FIFO) manner, and their buffer sizes are assumed to be infinitely large. The waiting times at the local data queue and the remote data queue capture the transmission time of the offloaded workload and the response time at the cloud instance, respectively.

TABLE I: A summary of the key notations.

λ	input workload rate at mobile device
c (c_ℓ)	execution capacity of cloud instance (device)
$\hat{c}$	effective execution capacity of remote queue
τ	target average response time
ρ	average capacity utilization of cloud instance
X (Y)	fluid process of local (remote) queue
S (Ω)	CTMC that models the wireless channel (offloaded demand)
R	rate process of offloaded demand
i_+ (i_0)	a state in which S is in state i and $X>0$ ($X=0$)
$\mathcal{S}_U^{(S)}$ ($\mathcal{S}_D^{(S)}$)	set of wireless channel states that lead to upward (downward) drifts in local data queue
$Q^{(S)}$ ($Q^{(\Omega)}$)	intensity matrix of CTMC S (Ω)
$\boldsymbol{r}^{(S)}$ ($\boldsymbol{r}^{(\Omega)}$)	rate vector of CTMC S (Ω)
$\boldsymbol{d}^\ell$ ($\boldsymbol{d}^e$)	fluid drift vector of local (remote) queue
$F_i^\ell(\cdot)$ ($F_i^e(\cdot)$)	stationary joint distribution of local (remote) queue and its corresponding CTMC
$\boldsymbol{\pi}^{(S)}$ ($\boldsymbol{\pi}^{(\Omega)}$)	stationary distribution of CTMC S (Ω)
ξ_k^ℓ, $\boldsymbol{v}_k^\ell$	the k-th eigenvalue and eigenvector of the matrix $[Q^{(S)}]^{-1}\mathrm{diag}(\boldsymbol{d}^\ell)$
a_k^ℓ	the k-th coefficient of MMFM for local queue
$\bar{Y}^\ell$ ($\bar{Y}^e$)	average fluid of local (remote) queue
W^ℓ (W^e)	average waiting time of local (remote) queue

In the rest of this section, we first derive the stationary distribution of the local data queue and its average waiting time in the first sub-section. In the second sub-section, we approach the offloaded demand, which is both the output process of the local data queue and the input process of the remote data queue, by proving that it is an SMP and deriving the statistical characteristics of this SMP. To gain more insight into our derived analytical results, in the third sub-section, we apply them to a Gilbert-Elliott channel, a widely adopted wireless channel model. The key notations used in this paper have been summarized in Table I.

A. Analyzing Local Data Queue

We assume that the input workload of the end device is highly fine-grained and can be modeled as a continuous fluid that flows into the local data queue at a constant rate λ. The assumption of a fine-grained workload has been widely used in the computation offloading literature [7], [8], [40], [41]. In general, this assumption is valid for many real-world applications that embrace incoming data streams at a constant rate, such as stream analytics applications for monitoring and

[4]In this paper, we define the offloaded demand as the amount of data to be processed in the remote data queue per unit time.

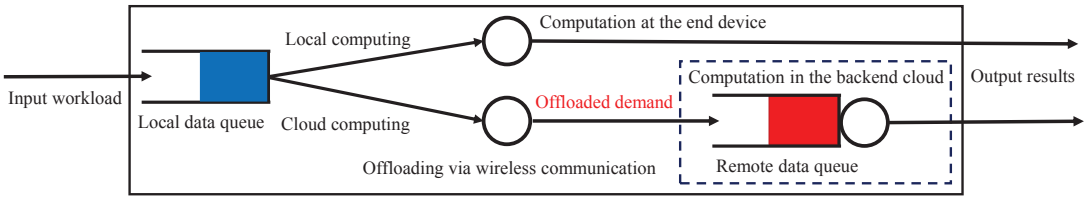


Fig. 3: A tandem queue model for a practical mobile cloud computing system with computation offloading. In this model, part of the workload at the end device is executed locally, while some is offloaded to the backend cloud for execution, through a wireless communication channel.

surveillance purposes [42], [43]. In this paper, the bursty demand of the mobile cloud computing system refers to the demand offloaded to the backend cloud, as shown in Figure 3, resulting from the variations of the wireless channel.

After the input workload arrives at the local data queue, it will be processed by either the local computing or cloud computing interface. In this paper, we consider a default scheduling policy, which greedily sends the workload to the local computing interface first and then to the cloud computing interface, as long as the local data queue is non-empty. The default policy minimizes the latency in the local data queue since it makes the best use of the local computing resources for processing the fluid workload. Through the local computing interface, the workload is executed at a constant processing rate c_ℓ. Through the cloud computing interface, on the other hand, the workload is offloaded at a rate up to the time-varying throughput of the wireless channel. Thus, both the output rate of the local data queue and the input rate of the remote data queue highly depend on the evolution of the stochastic wireless channel, leading to the stochasticity of both queues. The backend cloud instance can process the computation demand at a maximum rate of c, which is referred to as the instance's execution capacity in this paper. By analyzing the computation offloading process, as shown in Figure 3, we aim to establish the analytical relationship between the execution capacity and the resulting average response time, given the parameters of the system environment.

According to [44] and [45], from an application layer perspective, we can model the wireless channel as a CTMC $S := \{S_t\}_{t\geq 0}$, whose intensity matrix is denoted by $Q^{(S)}$. Let $\mathcal{S}^{(S)} := \{1, \ldots, N\}$ denote the state space of S. When the wireless channel is in state i, its throughput is denoted by r_i. We refer to $\boldsymbol{r}^{(S)} := [r_1, \ldots, r_N]$ as the concatenated vector of r_i. Thus, the wireless channel can be fully characterized by its intensity matrix $Q^{(S)}$ and throughput vector $\boldsymbol{r}^{(S)}$.

We then determine the stationary distribution of the fluid content in the local data queue. Let X_t denote the amount of fluid at time t. The dynamics of X_t depends on the wireless channel S given an input workload λ. When the channel is in state i, the fluid drift of the local data queue is

$$d_i^\ell = \lambda - c_\ell - r_i, \quad \forall i \in \mathcal{S}^{(S)}, \tag{9}$$

which is the input workload minus the total output rate from local computing and offloading. Therefore, the joint process $\{(X_t, S_t)\}_{t\geq 0}$ forms an MMFM with an intensity matrix $Q^{(S)}$ and a fluid drift vector $\boldsymbol{d}^\ell := [d_1^\ell, \ldots, d_N^\ell]$. Let $F_i^\ell(x)$ denote the joint stationary distribution of the fluid and wireless channel as defined in (3). By the fluid flow analysis framework introduced in Section II-B, the stationary distribution can be derived as

$$F_i^\ell(x) = \pi_i^{(S)} + \sum_{k=1}^{|\mathcal{S}_U^{(S)}|} a_k^\ell e^{\xi_k^\ell x} v_{ki}^\ell, \quad i \in \mathcal{S}^{(S)}, x \geq 0. \tag{10}$$

Given the stationary distribution, the average amount of fluid at the local data queue is

$$\bar{Y}^\ell(\lambda; \boldsymbol{r}^{(S)}, Q^{(S)}) = \sum_{i\in\mathcal{S}^{(S)}} \int_{x=0}^{\infty} x dF_i^\ell(x), \tag{11}$$

and based on Little's Law [46], the average waiting time is

$$W^\ell(\lambda; \boldsymbol{r}^{(S)}, Q^{(S)}) = \frac{\bar{Y}^\ell(\lambda; \boldsymbol{r}^{(S)}, Q^{(S)})}{\lambda}. \tag{12}$$

Remark 1 (Default Scheduling Policy)**.** *In this paper, we primarily focus on analyzing the latency of a mobile cloud computing system under a default scheduling policy. For many IoT devices that have a constant and stable (e.g., cable-connected) power supply and communicate with the backend cloud via wireless channels, such as surveillance cameras, the energy consumption of local computing is not a constraint. In such cases, our local-computing-first policy can minimize the latency of the local data queue. A related and classical problem in the computation offloading literature is to optimize the scheduling policy (or the offloading decisions) to balance the energy consumption of local computing and the latency of the local data queue [3], [4], [47]. This is a dynamic control problem, which is different from our work on performance analysis in this paper. We will consider the dynamic control problem in computation offloading as future work.*

B. Characterizing Offloaded Demand

The offloaded demand is essentially the output process of the local data queue through the cloud computing interface. Therefore, this demand depends on both the buffered fluid in the local data queue and the conditions of the wireless channel. Define i_0 as the state that represents the buffered fluid being zero and the wireless channel being in state i. Similarly, define i_+ as the state indicating that there is a positive amount of buffered fluid and the channel is in state i. Let $\Omega = \{\Omega_t\}_{t\geq 0}$ denote the underlying state evolution process of the offloaded demand. Since the local data queue can be empty only when the channel is in the states with downward drifts, we define

the state space of the underlying process Ω as the union of three separate sets as follows:

$$\mathcal{S}^{(\Omega)} := \mathcal{S}_{U_+}^{(S)} \cup \mathcal{S}_{D_0}^{(S)} \cup \mathcal{S}_{D_+}^{(S)}, \tag{13}$$

where the sets $\mathcal{S}_{U_+}^{(S)} := \{i_+ | i \in \mathcal{S}_U^{(S)}\}$ and $\mathcal{S}_{D_+}^{(S)} := \{i_+ | i \in \mathcal{S}_D^{(S)}\}$ contain the states under which the local data queue is non-empty and has upward and downward drifts, respectively. Meanwhile, the set $\mathcal{S}_{D_0}^{(S)} := \{i_0 | i \in \mathcal{S}_D^{(S)}\}$ includes the states with empty queues and downward fluid drifts.

When the local data queue is non-empty, the transmission rate of the offloaded demand is exactly the same as the time-varying throughput of the wireless channel. However, when the local data queue is empty, the offloaded demand is limited to the net input workload $\lambda - c_\ell$. To avoid the trivial scenario that the local data queue is always empty, the local computing capacity c_ℓ is assumed to be smaller than the input workload λ. Consequently, the rate process of offloaded demand $R := \{R_t\}_{t\geq 0}$ can be constructed by

$$R_t = \begin{cases} r_i & \text{if } \Omega_t = i_+, \quad i_+ \in \mathcal{S}_{D_+}^{(S)} \cup \mathcal{S}_{U_+}^{(S)}, \\ \lambda - c_\ell & \text{if } \Omega_t = i_0, \quad i_0 \in \mathcal{S}_{D_0}^{(S)}. \end{cases} \tag{14}$$

Thus, the rate vector $\boldsymbol{r}^{(\Omega)}$ of the offloaded demand can be defined as

$$\boldsymbol{r}^{(\Omega)} = \left[r_{1:|\mathcal{S}_U^{(S)}|}, (\lambda - c_\ell) \cdot \mathbf{1}_{1\times|\mathcal{S}_D^{(S)}|}, r_{|\mathcal{S}_U^{(S)}|+1:|\mathcal{S}_U^{(S)}|+|\mathcal{S}_D^{(S)}|} \right].$$

In summary, the offloaded demand can be fully determined by the underlying process Ω and the rate vector $\boldsymbol{r}^{(\Omega)}$. The following theorem proves that Ω is an SMP and derives its statistical characteristics.

Theorem 1. *Process Ω is an SMP and its transition kernel is given as follows.*
(i) For the transitions starting from state $i_0 \in \mathcal{S}_{D_0}^{(S)}$,

$$G_{i_0 n}(t) = \frac{Q_{ij}^{(S)}}{-Q_{ii}^{(S)}} \left(1 - e^{-Q_{ij}^{(S)} t}\right), \quad n = j_+ \in \mathcal{S}_{U_+}^{(S)}, \text{or } n = j_0 \in \mathcal{S}_{D_0}^{(S)} \setminus i_0.$$

(ii) For the transitions starting from state $i_+ \in \mathcal{S}_{U_+}^{(S)}$,

$$G_{i_+ j_+}(t) = \frac{Q_{ij}^{(S)}}{-Q_{ii}^{(S)}} \left(1 - e^{-Q_{ij}^{(S)} t}\right), j_+ \in \mathcal{S}_{D_+}^{(S)} \cup \mathcal{S}_{U_+}^{(S)} \setminus i_+.$$

(iii) For the transitions starting from state $i_+ \in \mathcal{S}_{D_+}^{(S)}$,

$$G_{i_+ n}(t) = \begin{cases} q_{i_+ i_0} \sum_{k=1}^{|\mathcal{S}_U^{(S)}|} p_{ik} \left(1 - e^{-\xi_k^\ell d_i^\ell t}\right), n = i_0 \in \mathcal{S}_{D_0}^{(S)}, \\ q_{i_+ j_+} (1 - e^{-Q_{ij}^{(S)} t}), n = j_+ \in \mathcal{S}_{D_+}^{(S)} \cup \mathcal{S}_{U_+}^{(S)} \setminus i_+, \end{cases}$$

where the transition probabilities are given by

$$q_{i_+ i_0} = \sum_{k=1}^{|\mathcal{S}_U^{(S)}|} \frac{p_{ik} \xi_k^\ell d_i^\ell}{\xi_k^\ell d_i^\ell - Q_{ii}^{(S)}} \text{ and } q_{i_+ j_+} = \sum_{k=1}^{|\mathcal{S}_U^{(S)}|} \frac{p_{ik} Q_{ij}^{(S)}}{\xi_k^\ell d_i^\ell - Q_{ii}^{(S)}},$$

with mixture probabilities, for $k = 1, \ldots, |\mathcal{S}_U^{(S)}|$,

$$p_{ik} := \frac{\sum_{j\in\mathcal{S}^{(S)}\setminus i} Q_{ji}^{(S)} a_k^\ell v_{kj}^\ell}{\sum_{m=1}^{|\mathcal{S}_U^{(S)}|} \sum_{j\in\mathcal{S}^{(S)}\setminus i} Q_{ji}^{(S)} a_m^\ell v_{mj}^\ell}.$$

Proof. To prove that Ω is an SMP, we need to verify two conditions according to Definition 1.

Definition 1 (Semi-Markov Process)**.** *A stochastic process $Z := \{Z_t\}_{t\geq 0}$ defined on a finite state space $\mathcal{S}^{(Z)}$ is called a semi-Markov process (SMP) if, whenever Z enters state $i \in \mathcal{S}^{(Z)}$, (i) it will visit state $j \in \mathcal{S}^{(Z)} \setminus i$ in the next transition with a fixed probability q_{ij}, and (ii) given the next state is j, the transition time T_{ij} from state i to state j has a fixed distribution.*

Definition 2 (Transition Kernel)**.** *The transition kernel $G_{ij}(t)$ of an SMP Z is the stationary probability that Z transits from state $i \in \mathcal{S}^{(Z)}$ to state $j \in \mathcal{S}^{(Z)} \setminus i$ and the transition time T_{ij} is no larger than t, i.e.,*

$$G_{ij}(t) = q_{ij} \cdot \mathsf{P}\left(T_{ij} \leq t\right). \tag{15}$$

In what follows, we will respectively derive the transition probability q_{ij} and the distribution of the transition time T_{ij} for process Ω.

We first note that all state transitions of process Ω are due to the state transitions of the wireless channel, except the transitions from states $i_+ \in \mathcal{S}_{D_+}^{(S)}$ to $i_0 \in \mathcal{S}_{D_0}^{(S)}$, which result from the draining of fluid in the local data queue from a non-empty value to zero. Thus, for the transitions starting from states $i_0 \in \mathcal{S}_{D_0}^{(S)}$ and $i_+ \in \mathcal{S}_{U_+}^{(S)}$, the transition times to any feasible states are the same as those of the corresponding wireless channel state transitions and are exponentially distributed. Thus, we have

$$q_{ij} = \frac{Q_{ij}^{(S)}}{-Q_{ii}^{(S)}}, \text{ and } \mathsf{P}(T_{ij} \leq t) = 1 - e^{-Q_{ij}^{(S)} t}. \tag{16}$$

As a result, the transition kernel of Ω in cases (*i*) and (*ii*) of Theorem 1 can be determined based on Equation (15).

To derive the distribution of the time $T_{i_+ i_0}$ for the transition from state $i_+ \in \mathcal{S}_{D_+}^{(S)}$ to state $i_0 \in \mathcal{S}_{D_0}^{(S)}$, we first show Lemma 1, which gives the conditional distribution of the local data queue observed just after the transitions of the wireless channels. Lemma 1 can be proved by following the proof of Proposition 4.1 in [20].

Lemma 1. *The stationary distribution of the fluid in the local data queue at the onset of channel state i is given by*

$$H_i(x) = \sum_{j\in\mathcal{S}^{(S)}\setminus i} \frac{Q_{ji}^{(S)}}{-Q_{ii}^{(S)} \pi_i^{(S)}} F_j^\ell(x), \forall i \in \mathcal{S}^{(S)},$$

where $F_j^\ell(x)$ is the joint stationary distribution of the MMFM $\{(X_t, S_t)\}_{t\geq 0}$ given in Equation (10)*, and $\boldsymbol{\pi}^{(S)}$ is the stationary distribution of the wireless channel.*

Since the fluid content has a stationary distribution at the onset of each channel state, the transition time for the fluid X in the local data queue to drain to zero depends only on the current channel state i and can be determined by $X/(-d_i^\ell)$,

where d_i^ℓ is the downward drift in state i. We can then derive the distribution of $T_{i_+i_0}$:

$$\begin{aligned}\mathsf{P}(T_{i_+i_0} \le t) &= \mathsf{P}\left(X/(-d_i^\ell) \le t | S=i, X>0\right)\\ &= \frac{\mathsf{P}\left(0 < X \le -d_i^\ell t | S=i\right)}{\mathsf{P}\left(X>0|S=i\right)}\\ &= \frac{H_i(-d_i^\ell t) - H_i(0)}{1-H_i(0)},\\ &= \frac{\sum_{j\in\mathcal{S}^{(S)}\setminus i} \frac{Q_{ji}^{(S)}}{-Q_{ii}^{(S)}}\left[F_j^\ell(-d_i^\ell t) - F_j^\ell(0)\right]}{\pi_i^{(S)} - \sum_{j\in\mathcal{S}^{(S)}\setminus i}\frac{Q_{ji}^{(S)}}{-Q_{ii}^{(S)}}F_j^\ell(0)}.\end{aligned}$$

Recall the structure of the joint stationary distribution $F_j^\ell(x) = \pi_j^{(S)} + \sum_{k=1}^{|\mathcal{S}_U|} a_k^\ell v_{kj}^\ell e^{\xi_k^\ell x}$ in Equation (10). Substitute it into the transition time distribution and note the fact that

$$\sum_{j\in\mathcal{S}^{(S)}\setminus i}\frac{Q_{ji}^{(S)}}{-Q_{ii}^{(S)}}\pi_j^{(S)} = \pi_i^{(S)}, \quad \forall i \in \mathcal{S}^{(S)}. \tag{17}$$

We then have

$$\begin{aligned}\mathsf{P}\left(T_{i_+i_0} \le t\right) &= \sum_{k=1}^{|\mathcal{S}_U^{(S)}|}\frac{\sum_{j\in\mathcal{S}\setminus i}\frac{Q_{ji}^{(S)}}{-Q_{ii}^{(S)}}a_k^\ell v_{kj}^\ell\left(1-e^{-\xi_k^\ell d_i^\ell t}\right)}{\sum_{m=1}^{|\mathcal{S}_U|}\sum_{j\in\mathcal{S}^{(S)}\setminus i}\frac{Q_{ji}^{(S)}}{-Q_{ii}^{(S)}}a_m^\ell v_{mj}^\ell},\\ &= \sum_{k=1}^{|\mathcal{S}_U^{(S)}|} p_{ik}\left(1-e^{-\xi_k^\ell d_i^\ell t}\right),\end{aligned} \tag{18}$$

where the mixture probabilities, for $k = 1, \ldots, |\mathcal{S}_U^{(S)}|$, are

$$p_{ik} = \frac{\sum_{j\in\mathcal{S}^{(S)}\setminus i} Q_{ji}^{(S)} a_k^\ell v_{kj}^\ell}{\sum_{m=1}^{|\mathcal{S}_U^{(S)}|}\sum_{j\in\mathcal{S}^{(S)}\setminus i} Q_{ji}^{(S)} a_m^\ell v_{mj}^\ell}. \tag{19}$$

Note that $\sum_{k=1}^{|\mathcal{S}_U^{(S)}|} p_{ik} = 1$. Therefore, $T_{i_+i_0}$ is an $|\mathcal{S}_U^{(S)}|$-order hyper-exponential random variable. Recall that the transition time $T_{i_+j_+}, j \in \mathcal{S}^{(S)} \setminus i$, is an exponential variable with the mean $1/Q_{ij}^{(S)}$. The probability that Ω transits from state i_+ to state i_0 can be determined by

$$\begin{aligned}q_{i_+i_0} &= \mathsf{P}\left(T_{i_+i_0} < \min_{j\in\mathcal{S}^{(S)}\setminus i} T_{i_+j_+}\right),\\ &= \int_0^\infty \mathsf{P}\left(T_{i_+i_0}\le t\right)\mathsf{P}\left(\min_{j\in\mathcal{S}^{(S)}\setminus i} T_{i_+j_+} = t\right)dt,\\ &= \int_0^\infty \sum_{k=1}^{|\mathcal{S}_U^{(S)}|} p_{ik}\left(1-e^{-\xi_k^\ell d_i^\ell t}\right)(-Q_{ii}^{(S)})e^{Q_{ii}^{(S)}t}dt,\\ &= \sum_{k=1}^{|\mathcal{S}_U^{(S)}|}\frac{p_{ik}\xi_k^\ell d_i^\ell}{\xi_k^\ell d_i^\ell - Q_{ii}^{(S)}},\end{aligned} \tag{20}$$

where $\min_{j\in\mathcal{S}^{(S)}\setminus i} T_{i_+j_+}$ is an exponential random variable with the mean $1/(-Q_{ii}^{(S)})$. In the same way, we can derive the transition probabilities

$$q_{i_+j_+} = \sum_{k=1}^{|\mathcal{S}_U^{(S)}|}\frac{p_{ik}Q_{ij}^{(S)}}{\xi_k^\ell d_i^\ell - Q_{ii}^{(S)}}, \quad j \in \mathcal{S}^{(S)}\setminus i. \tag{21}$$

To sum up, we have derived q_{ij} and the distribution of T_{ij} for all possible transitions for $i \in \mathcal{S}^{(\Omega)}, j \in \mathcal{S}^{(\Omega)} \setminus i$. Therefore, Ω is an SMP, and its transition kernel can be determined by Equations (15), (16), and (18)–(21). □

We can interpret the process Ω as follows. Ω jumps within a finite state space $\mathcal{S}^{(\Omega)}$, and the transition time between two consecutive jumps is exponentially distributed if the jump starts from state $i_0 \in \mathcal{S}_{D_0}^{(S)}$ (i.e., case (*i*)) or $i_+ \in \mathcal{S}_{U_+}^{(S)}$ (i.e., case (*ii*)). The transition time becomes a hyper-exponential random variable if the jump starts from state $i_+ \in \mathcal{S}_{D_+}^{(S)}$ (i.e., case (*iii*)). This is because, in the first two cases, the jumps of the process Ω are driven by the transitions of the CTMC of the wireless channel, while in case (*iii*), i.e, a non-empty local data queue with downward drift, the jump depends on either the changes of the wireless channel or the draining of the local data queue. The time that the queue takes to drain to empty is usually not an exponential variable. We can also observe that when there is only one state with upward drift in the local data queue, i.e., $|\mathcal{S}_U^{(S)}| = 1$, the hyper-exponential transition time reduces to an exponentially distributed variable, and hence the SMP Ω reduces to a CTMC. Consequently, we have the following Corollary.

Corollary 1. *When the number of states with upward drifts of the local data queue is 1 (i.e., $|\mathcal{S}_U^{(S)}| = 1$), the process Ω is a CTMC.*

C. Case Study: Gilbert-Elliott Channel Model

To gain more insight into our offloaded demand model derived in Theorem 1, in this sub-section, we apply our model to the case where the wireless channel follows the Gilbert-Elliott model [48]. This model, which has been widely adopted in the wireless communication literature [49]–[51], can capture the key characteristics of a classical wireless channel while keeping the theoretical analysis tractable. In the model, the channel evolution is governed by a two-state CTMC $S := \{S_t \in \{0,1\} | t \ge 0\}$, where states 0 and 1 represent the bad and good channel conditions, respectively. The intensity matrix of S is given by

$$Q^{(S)} = \begin{bmatrix} -\alpha & \alpha \\ \beta & -\beta \end{bmatrix}, \tag{22}$$

where α and β respectively denote the transition rates from a bad state to a good state and from a good state to a bad state. In the Gilbert-Elliott model, a wireless channel cannot offload data in a bad state but can transmit data at a constant throughput r in a good state. Therefore, the rate vector is $\boldsymbol{r}^{(S)} = [0, r]$ and the fluid drift vector of the local data queue is $\boldsymbol{d}^\ell = [d_0^\ell, d_1^\ell] = [\lambda - c_\ell, \lambda - c_\ell - r]$. In this case, the joint stationary distribution of the fluid content process X and channel state process S can be derived in a closed form based on the approach introduced in Section II-B, as follows:

$$F_0^\ell(x) = \frac{\beta}{\alpha+\beta} - \frac{\beta}{\alpha+\beta}\cdot e^{\xi^\ell x}, \tag{23}$$

$$F_1^\ell(x) = \frac{\alpha}{\alpha+\beta} + \frac{\beta}{\alpha+\beta}\cdot\frac{d_0^\ell}{d_1^\ell}\cdot e^{\xi^\ell x}, \tag{24}$$

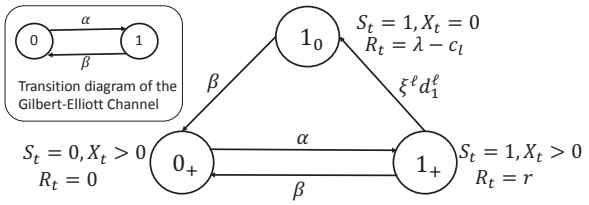


Fig. 4: Transition diagram of the offloaded process Ω under the Gilbert-Elliott channel. Except the transition from 1_+ to 1_0, all other transitions occur due to the transitions of the Gilbert-Elliott channel, and their transition times are also the same. The transition from 1_+ to 1_0 occurs when the fluid of the local data queue drains to empty before the wireless channel changes. This transition time is characterized as $\xi^\ell d_1^\ell$ according to case (*iii*) in Theorem 1.

where $\xi^\ell := -\alpha/d_0^\ell - \beta/d_1^\ell < 0$ is the negative eigenvalue.

The underlying process Ω of the offloaded demand is a three-state process, whose state space is $\mathcal{S}^{(\Omega)} := \{0_+, 1_0, 1_+\}$ and the associated rate vector is $\boldsymbol{r}^{(\Omega)} := \{0, \lambda - c_\ell, r\}$. Since the fluid drift vector $\boldsymbol{d}^\ell$ has only one positive value, Ω is a CTMC, as shown in Corollary 1. Its intensity rate matrix can be derived based on the transition kernel given in Theorem 1 as

$$Q^{(\Omega)} = \begin{bmatrix} -\alpha & 0 & \alpha \\ \beta & -\beta & 0 \\ \beta & \xi^\ell d_1^\ell & -\beta - \xi^\ell d_1^\ell \end{bmatrix}. \tag{25}$$

We depict the transition diagram of Ω in Figure 4.

The stationary distribution of Ω can then be derived as

$$\boldsymbol{\pi}^{(\Omega)} := \left[\frac{\beta}{\alpha+\beta}, \frac{\alpha}{\alpha+\beta} \cdot \frac{\xi^\ell d_1^\ell}{\xi^\ell d_1^\ell + \beta}, \frac{\alpha}{\alpha+\beta} \cdot \frac{\beta}{\xi^\ell d_1^\ell + \beta} \right].$$

Based on $\boldsymbol{\pi}^{(\Omega)}$, we can analytically obtain the first and second moments of the offloaded demand as

$$\mathsf{E}[R] = \sum\nolimits_{i \in \mathcal{S}^{(\Omega)}} r_i^{(\Omega)} \pi_i^{(\Omega)} = d_0^\ell$$

and

$$\mathsf{E}[R^2] = \sum\nolimits_{i \in \mathcal{S}^{(\Omega)}} (r_i^{(\Omega)})^2 \pi_i^{(\Omega)} = \frac{(\alpha+2\beta)(d_0^\ell)^2 - \beta d_0^\ell d_1^\ell}{\alpha+\beta}.$$

To examine the burstiness of the offloaded demand, we adopt the squared coefficient of variation (SCV) [52] of the stationary distribution of R as our measurement metric, i.e.,

$$\mathsf{SCV} = \frac{\mathsf{Var}[R]}{\mathsf{E}[R]} = \frac{\mathsf{E}[R^2]}{\mathsf{E}[R]^2} - 1 = \frac{\beta}{\alpha+\beta} \cdot \frac{r}{\lambda - c_\ell}. \tag{26}$$

It can be observed from Equation (26) that the burstiness of the offloaded demand depends on the stationary distribution of the wireless channel and the local computing capacity. In particular, the SCV of the offloaded demand grows with an increase in the stationary probability of the channel in a bad state (i.e., $\beta/(\alpha+\beta)$). On the one hand, the offloading rate is 0 for a longer time when the channel stays in a bad state with a higher probability. On the other hand, a larger proportion of the time being in a bad state results in a larger amount of the workload being buffered in the local data queue. Therefore, it takes a longer time to drain the local data queue to empty at the peak offloading rate r during the good state period. As a result, the offloading rate concentrates more on the minimum and maximum offloading rates (i.e., 0 and r) with an increase in the stationary probability of a bad state, leading to a more bursty offloaded demand. In addition, the burstiness of the offloaded demand is inversely proportional to the local computing capacity since it can be easily checked that a larger c_ℓ leads to a smaller mean and a larger variance of R, and hence a larger SCV.

IV. Analytical Performance Model

In this section, we continue our analysis of the tandem queue model and analytically derive the average response time of the remote data queue given the offloaded demand and the resource capacity, which is hereinafter also referred to as the execution capacity, of the backend cloud instance.

A cloud instance takes the offloaded demand from a user as its input and executes the demand at a maximum speed of c. Let Y_t denote the amount of buffered fluid in the remote data queue. When $Y_t > 0$, a cloud instance processes the buffered fluid at a speed of c. When $Y_t = 0$, the cloud instance immediately executes the offloaded demand. Although the offloaded demand has been characterized as an SMP in Theorem 1, it is still non-trivial to evaluate the average response time of the remote data queue, since an SMP is only Markovian at its state transition epochs, but non-Markovian for the entire process. Based on the transition kernel $G_{ij}(t)$ given in Theorem 1, for the process Ω, the sojourn time is non-Markovian only for the states $i_+ \in \mathcal{S}_{D_+}^{(S)}$. Therefore, we propose an approximation of the SMP Ω as a CTMC $\tilde{\Omega} := \{\tilde{\Omega}_t\}_{t \geq 0}$. In particular, we approximate the sojourn time in state $i_+ \in \mathcal{S}_{D_+}^{(S)}$ by an exponential variable with the same mean. The approximation is summarized in Proposition 1.

Proposition 1. *The SMP Ω can be approximated by a CTMC $\tilde{\Omega}$ with the same mean sojourn time, where $\tilde{\Omega}$ has the intensity matrix*

$$Q^{(\tilde{\Omega})} = \begin{bmatrix} Q_{UU}^{(S)} & \mathbf{0} & Q_{UD}^{(S)} \\ Q_{DU}^{(S)} & Q_{DD}^{(S)} & \mathbf{0} \\ Q_{DU}^{(S)} & \Theta & Q_{DD}^{(S)} - \Theta \end{bmatrix}, \tag{27}$$

in which $Q_{DD}^{(S)}$, $Q_{DU}^{(S)}$, $Q_{UD}^{(S)}$, and $Q_{UU}^{(S)}$ are sub-matrices that are obtained by partitioning $Q^{(S)}$, and Θ is a diagonal matrix with each diagonal element as

$$\Theta_{ii} = \frac{q_{i_+ i_0}}{\sum_{k=1}^{|\mathcal{S}_U^{(S)}|} p_{ik}/(\xi_k^\ell d_i^\ell - Q_{ii}^{(S)})}, \quad i \in \mathcal{S}_D^{(S)}. \tag{28}$$

Proof. Let $T_{i_+} := \min\{T_{i_+ i_0}, \{T_{i_+ j_+}\}_{j \in \mathcal{S}^{(S)} \setminus i}\}$ denote the sojourn time of state $i_+ \in \mathcal{S}_{D_+}^{(S)}$. The distribution of T_{i_+} is

$$\begin{aligned} \mathsf{P}\left(T_{i_+} > t\right) &= \mathsf{P}\left(\min\{T_{i_+ i_0}, \{T_{i_+ j_+}\}_{j \in \mathcal{S}^{(S)} \setminus i}\} > t\right), \\ &= \mathsf{P}\left(T_{i_+ i_0} > t\right) \prod\nolimits_{j \in \mathcal{S}^{(S)} \setminus i} \mathsf{P}\left(T_{i_+ j_+} > t\right), \\ &= \sum_{k=1}^{|\mathcal{S}_U^{(S)}|} p_{ik} e^{-\xi_k^\ell d_i^\ell t} \cdot \prod\nolimits_{j \in \mathcal{S}^{(S)} \setminus i} e^{-Q_{ij}^{(S)} t}, \\ &= \sum_{k=1}^{|\mathcal{S}_U^{(S)}|} p_{ik} e^{-(\xi_k^\ell d_i^\ell - Q_{ii}^{(S)}) t}. \end{aligned}$$

Therefore, the sojourn time T_{i_+} is also a hyper-exponential variable and its mean value is

$$\mathsf{E}[T_{i_+}] = \sum_{k=1}^{|\mathcal{S}_U^{(S)}|} \frac{p_{ik}}{\xi_k^\ell d_i^\ell - Q_{ii}^{(S)}}, \quad i \in \mathcal{S}_D^{(S)}. \tag{29}$$

Our approach is to approximate the sojourn time T_{i_+} by an exponential variable with the same mean value. Thus, the transition rate from state i_+ to state $i_0 \in \mathcal{S}_{D_0}^{(S)}$ is

$$\theta_{i_+ i_0} = \frac{q_{i_+ i_0}}{\mathsf{E}[T_{i_+}]} = \frac{q_{i_+ i_0}}{\sum_{k=1}^{|\mathcal{S}_U^{(S)}|} p_{ik}/(\xi_k^\ell d_i^\ell - Q_{ii}^{(S)})}, \tag{30}$$

and the transition rate from state i_+ to state $j_+ \in \mathcal{S}_{U_+}^{(S)} \cup \mathcal{S}_{D_+}^{(S)}$ can be determined by

$$\theta_{i_+ j_+} = \frac{q_{i_+ j_+}}{\mathsf{E}[T_{i_+}]} = Q_{ij}^{(S)}, \tag{31}$$

where $q_{i_+ i_0}$ and $q_{i_+ j_+}$ are derived in Equations (20) and (21).

Partition $Q^{(S)}$ into four sub-matrices,

$$Q_{DD}^{(S)} := \{Q_{ij}^{(S)}\}_{i,j \in \mathcal{S}_D^{(S)}}, \quad Q_{DU}^{(S)} := \{Q_{ij}^{(S)}\}_{i \in \mathcal{S}_D^{(S)}, j \in \mathcal{S}_U^{(S)}},$$
$$Q_{UU}^{(S)} := \{Q_{ij}^{(S)}\}_{i,j \in \mathcal{S}_U^{(S)}}, \quad Q_{UD}^{(S)} := \{Q_{ij}^{(S)}\}_{i \in \mathcal{S}_U^{(S)}, j \in \mathcal{S}_D^{(S)}},$$

and define the diagonal matrix Θ with the elements $\Theta_{ii} = \theta_{i_+ i_0}, i \in \mathcal{S}_D^{(S)}$. The intensity matrix of the approximate CTMC $\tilde{\Omega}$ can then be derived as Equation (27). □

Based on the approximation in Proposition 1, the offloaded demand depends on the CTMC $\tilde{\Omega}$ with an intensity matrix $Q^{(\tilde{\Omega})}$ and a rate vector $\boldsymbol{r}^{(\tilde{\Omega})} = \boldsymbol{r}^{(\Omega)}$. As a result, the fluid drift of the remote data queue can be derived as

$$d_i^e = r_i^{(\tilde{\Omega})} - c, \quad i \in \mathcal{S}^{(\Omega)}. \tag{32}$$

We observe that the joint process $\{Y_t, \tilde{\Omega}_t\}_{t \geq 0}$ forms another MMFM, whose intensity matrix is $Q^{(\tilde{\Omega})}$ and drift vector is $\boldsymbol{d}^e := \{d_i^e\}_{i \in \mathcal{S}^{(\Omega)}}$. The joint stationary distribution $F_i^e(y)$ can be derived based on the fluid flow analysis, and the average amount of fluid in the remote data queue is

$$\bar{Y}^e(c; \boldsymbol{r}^{(\tilde{\Omega})}, Q^{(\tilde{\Omega})}) = \sum_{i \in \mathcal{S}^{(\Omega)}} \int_{y=0}^{\infty} y dF_i^e(y), \tag{33}$$

which depends on the offloaded demand characterized by $(\boldsymbol{r}^{(\tilde{\Omega})}, Q^{(\tilde{\Omega})})$ as well as the execution capacity c. Since the remote data queue has no loss of fluid, its throughput can be determined by the average offloaded demand as

$$\bar{R}(\boldsymbol{r}^{(\tilde{\Omega})}, Q^{(\tilde{\Omega})}) = \sum_{i \in \mathcal{S}^{(\Omega)}} \pi_i^{(\tilde{\Omega})} r_i^{(\tilde{\Omega})}, \tag{34}$$

where $\{\pi_i^{(\tilde{\Omega})}\}_{i \in \mathcal{S}^{(\Omega)}}$ is the stationary distribution of $\tilde{\Omega}$. According to Little's law, the average response time is

$$W^e(c; \boldsymbol{r}^{(\tilde{\Omega})}, Q^{(\tilde{\Omega})}) = \frac{\bar{Y}^e(c; \boldsymbol{r}^{(\tilde{\Omega})}, Q^{(\tilde{\Omega})})}{\bar{R}(\boldsymbol{r}^{(\tilde{\Omega})}, Q^{(\tilde{\Omega})})}. \tag{35}$$

We define the *effective execution capacity* of the remote data queue as the minimum capacity that is required to guarantee a target average response time. Let $\hat{c}^\tau := \hat{c}^\tau(\boldsymbol{r}^{(\tilde{\Omega})}, Q^{(\tilde{\Omega})})$ denote the effective execution capacity for ensuring the average response time is no larger than τ, i.e.,

$$W^e(\hat{c}^\tau, \boldsymbol{r}^{(\tilde{\Omega})}, Q^{(\tilde{\Omega})}) \leq \tau. \tag{36}$$

Note that given the offloaded demand, the average response time $W^e(c, \boldsymbol{r}^{(\tilde{\Omega})}, Q^{(\tilde{\Omega})})$ is non-increasing in c. Therefore, $\hat{c}^\tau$ can be numerically derived by a bi-section algorithm efficiently. To guarantee an average response time τ, the average capacity utilization of an instance is

$$\rho^\tau(\boldsymbol{r}^{(\tilde{\Omega})}, Q^{(\tilde{\Omega})}) = \frac{\bar{R}(\boldsymbol{r}^{(\tilde{\Omega})}, Q^{(\tilde{\Omega})})}{\hat{c}^\tau(\boldsymbol{r}^{(\tilde{\Omega})}, Q^{(\tilde{\Omega})})}. \tag{37}$$

Consequently, to accommodate the offloaded demand with various degrees of burstiness, cloud instances need to be configured with different execution capacities to ensure the same average response time.

Given a QoS requirement of a user, our performance model can analytically determine the required volume of the resource capacity at the backend cloud instance to satisfy this QoS. The corresponding utilization of this capacity can also be derived from our performance model. To guarantee the resource provisioning decision at the backend cloud instance with the user's QoS requirement, our final step is to determine how to provision the required resource capacity derived from our performance model in the cloud. Current public cloud providers mainly offer two types of cloud instances, static instances and burstable instances. Based on the results derived from our performance model, in the next section, we will calculate the corresponding charges of these two instance types and then determine which should be selected to provision the required resource capacity with a lower monetary cost.

Remark 2. *To the best of our knowledge, we are the first to analytically model the relationship between the desired QoS of a user and the correspondingly required resource capacity at the backend cloud instance, taking into account practical system models, such as the CTMC wireless channel. Most of the existing works treat the average response time as a positive constant, or even zero, while other works use oversimplified and unrealistic assumptions and models to address the average response time (see Section II-A for details).*

V. Making Resource Provisioning Decision

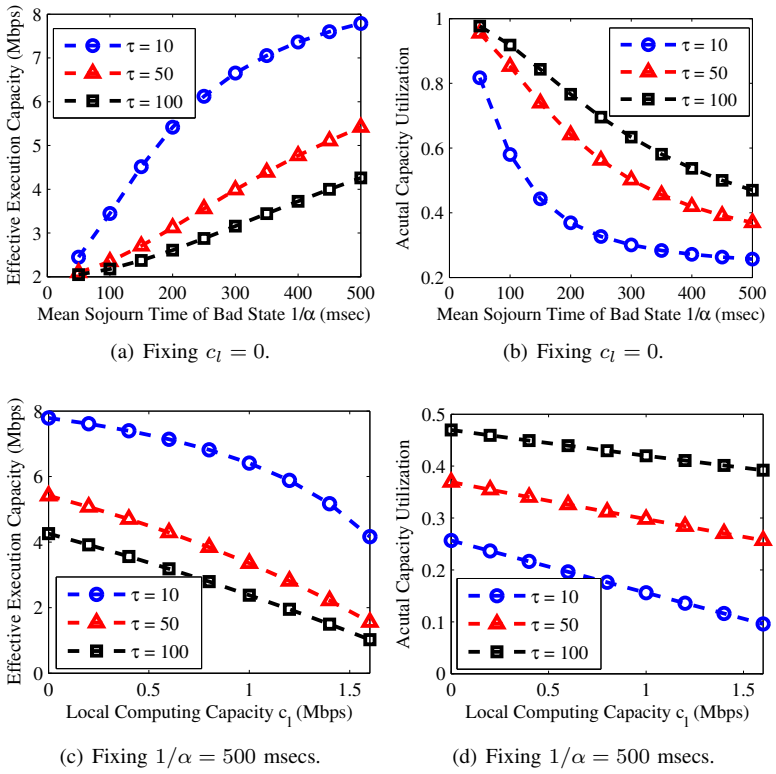
In this section, we examine whether, for an individual user, a static instance or a burstable instance should be selected to provision the required resource capacity to satisfy his/her QoS requirement with a lower monetary cost.

To execute the offloaded demand, a user can configure the backend cloud instance as a traditional static instance or a burstable instance.[5] A static instance provides a fixed execution capacity c, which is the maximum speed to execute the offloaded demand. The execution capacity can be used up at all times with a fixed price. In contrast, although a burstable instance also provides an execution capacity c, it has

[5]There are multiple implementations of burstable instances. In this paper, we focus on the latest implementation, the unlimited mode (e.g., the t3 instance in Amazon EC2 [24]).

TABLE II: Resource configurations and pricing parameters from the static instance (SI) and burstable instance (BI) offerings of Amazon EC2 (Ohio, USA), as of March 2022 [53].

SI	BI	vCPUs	Memory	c	$P_s(c)$	$\bar{P}_b(c)$	$\hat{P}_b(c)$	ρ_c	ρ_c^*
m5.large	t3.large	2	8 GiB	2.5 Mbps	0.096/hour	0.0832/hour	0.1/hour	0.3	0.428
m5.xlarge	t3.xlarge	4	16 GiB	5 Mbps	0.192/hour	0.1664/hour	0.2/hour	0.4	0.528
m5.2xlarge	t3.2xlarge	8	32 GiB	10 Mbps	0.384/hour	0.3328/hour	0.4/hour	0.4	0.528



(a) Fixing $c_l = 0$. (b) Fixing $c_l = 0$.

(c) Fixing $1/\alpha = 500$ msecs. (d) Fixing $1/\alpha = 500$ msecs.

Fig. 5: Effective execution capacity and actual capacity utilization under different wireless channels and local computing capacities. We set the mean sojourn time of a good channel state to be $1/\beta = 1$ second.

another pricing parameter, ρ_c, named the baseline. Recall from Section I that the final charge of a burstable instance depends on both the offered execution capacity c and its actual capacity utilization ρ. If ρ is smaller than or equal to ρ_c, a fixed price will be charged which is smaller than that of a static instance with the same execution capacity. However, if ρ is larger than ρ_c, an additional charge, which is proportional to the surplus capacity utilization, namely $\rho - \rho_c$, will be imposed.

We compare the prices of a static instance and a burstable instance that can accommodate the same offloaded demand and guarantee the same average response time in the remote data queue. Let $P_s(c)$ denote the price of a static instance with an execution capacity c. Recall that a burstable instance can be characterized by its execution capacity c and baseline ρ_c. Its price depends on c, ρ_c, and its actual capacity utilization ρ, and can be expressed as

$$P_b(c, \rho) = \bar{P}_b(c) + \hat{P}_b(c) \cdot [\rho - \rho_c]^+, \tag{38}$$

where $\bar{P}_b(c)$ is a capacity-dependent fixed price if the actual capacity utilization is lower than or equal to the baseline ρ_c and $\hat{P}_b(c)$ is the price of the surplus capacity utilization. By letting $P_b(c, \rho) \leq P_s(c)$, we have the following condition for a user to subscribe to a burstable instance as the backend cloud instance with a lower monetary cost:

$$\rho \leq \rho_c + \frac{P_s(c) - \bar{P}_b(c)}{\hat{P}_b(c)}. \tag{39}$$

Define the right-hand side of Equation (39) as a utilization threshold ρ_c^*, i.e.,

$$\rho_c^* = \rho_c + \frac{P_s(c) - \bar{P}_b(c)}{\hat{P}_b(c)}. \tag{40}$$

The instance with an execution capacity c and an actual capacity utilization ρ can achieve a lower monetary cost by subscribing to burstable instances when $\rho \leq \rho_c^*$.

Table II presents the resource configurations and pricing parameters of three static instances and three burstable instances from the latest instance offerings of Amazon EC2 (Ohio, USA) [24], a leading public cloud provider. The six cloud instances can be categorized into three classes, with each provisioned with a different execution capacity. For example, the static instance m5.large and the burstable instance t3.large belong to the same class with two vCPUs and 8 GiB memory that can support an execution capacity $c = 2.5$ Mbps. According

to Equation (40), we can derive the utilization threshold ρ_c^* for the three classes, as also shown in Table II.

To generalize our results, we consider a future public cloud where the execution capacity is offered in a more fine-grained manner. We consider that the utilization threshold ρ_c^*, which is essentially a pricing-related parameter, grows linearly with regard to the provisioned execution capacity c until a capped value.[6] In this regard, we conduct a linear regression from the real-world pricing data given in Table II and get the following relationship between ρ_c^* and c:

$$\rho_c^* = \begin{cases} 0.328 + 0.04c & \text{if } 0 \le c \le 5 \text{ Mbps}, \\ 0.528 & \text{if } c > 5 \text{ Mbps}. \end{cases} \tag{41}$$

Since the stochasticity of the remote data queue comes from the wireless channel, it is natural to change the instance according to the specific wireless channel statistics. In what follows, we take the Gilbert-Elliott channel model as an example to show our results on instance selection. As discussed in Section III-C, the Gilbert-Elliott channel model can be characterized by two parameters, the mean sojourn times in bad states and good states, $1/\alpha$ and $1/\beta$, respectively. We fix the input workload $\lambda = 2$ Mbps and set the local computing capacity to c_ℓ. Given the parameters $1/\alpha$, $1/\beta$, λ, and c_ℓ, we can derive the offloaded demand with the rate vector $\boldsymbol{r}^{(\Omega)}$ and the intensity matrix $Q^{(\Omega)}$. Given the execution capacity c, we can further determine the effective execution capacity $\hat{c}^\tau$ and the actual capacity utilization $\hat{\rho}^\tau$ to guarantee the QoS requirements $\tau \in \{10, 50, 100\}$ milliseconds.

In Figures 5(a) and 5(b), we fix the local computing capacity as $c_\ell = 0$, which is equivalent to fixing the throughput of the remote data queue as $\bar{R} = \lambda$. Recall that the burstiness of the offloaded demand increases with the mean sojourn time of a bad state $1/\alpha$. It can be observed that the effective execution capacity and the actual capacity utilization respectively increase and decrease when $1/\alpha$ grows. This is because additional execution capacity is needed to accommodate more bursty workload so that the same average response time can be guaranteed. Moreover, the more stringent the QoS requirement is, the greater the execution capacity that will be required. The actual capacity utilization, which is the ratio of the throughput and the effective execution capacity (i.e., $\bar{R}/\hat{c}^\tau$), is hence monotonically decreasing in $1/\alpha$. In Figures 5(c) and 5(d), we fix the mean sojourn time of a bad state to $1/\alpha = 500$ milliseconds and investigate the impact of the local computing capacity c_ℓ. Although the burstiness of the offloaded demand increases with c_ℓ, the effective execution capacity can still be a decreasing function of c_ℓ since the throughput of the remote data queue (i.e., $\bar{R} = \lambda - c_\ell$) decreases with c_ℓ and can dominate the monotonicity. Moreover, it can also be true that a larger c_ℓ can lead to a lower actual capacity utilization due to the decrease of the throughput with c_ℓ.

To gain more insight into our theoretical results on the economical selection of the instance type, for a given $c_\ell = 0$, we choose ten typical values of $1/\alpha$ and $1/\beta$ that can equally divide their feasible ranges, $1/\alpha \in [50, 500]$ milliseconds and $1/\beta \in [1, 10]$ seconds, into nine intervals, respectively [51]. By combining the typical values of the two parameters, we construct 100 Gilbert-Elliott channels for each given QoS requirement. In this way, we finally obtain 300 scenarios, which are distinguished by channels and QoS requirements. We show the SCV and the actual capacity utilization of each scenario in Figure 6(a). The circle, star, and square symbols represent the channels with QoS requirements of 10, 50, and 100 milliseconds, respectively. Based on Equations (39) and (41), we can select different types of instances for different scenarios. Black and blue distinguish the scenarios that are configured to static instances and burstable instances, respectively. We observe that the scenarios with stringent QoS requirements (circles) are more likely to have lower utilizations and save monetary costs with the choice of burstable instances. Moreover, the scenarios with the high SCVs (i.e., high degrees of burstiness) are more likely to have burstable instances be the better choice. In Figure 6(b), we fix $1/\alpha = 500$ milliseconds and construct another 300 scenarios by choosing values for $c_\ell \in [0, 1.6]$ Mbps, $1/\beta \in [1, 10]$ seconds, and $\tau \in \{10, 50, 100\}$ milliseconds. With the varying local computing capacity, the SCVs of the scenarios also vary in a much larger range and more scenarios are configured to burstable instances to save monetary costs.

[6]This is one of the commonly used cloud pricing strategies in which the charged price is linear to the resource provisioned [54].

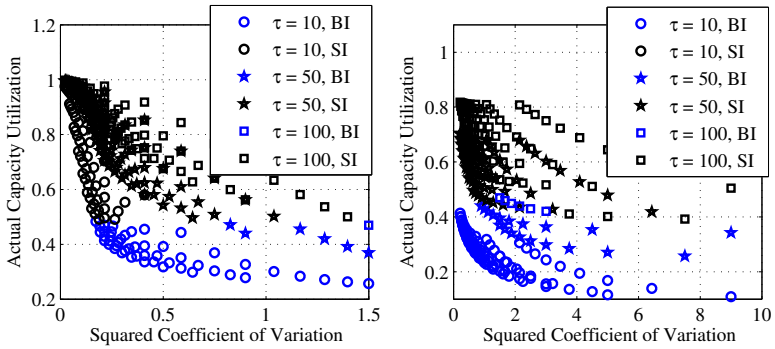
VI. Trace-Driven Simulations

In this section, we present numerical results of trace-driven simulations to further validate our theoretical results, i.e., the proposed analytical approximation in Proposition 1 and the corresponding instance selection. In the simulations, we adopt the dataset from [17] with 135 LTE traces. Each trace contains the throughput between users and base stations over 15 minutes at a granularity of one sample per second. In what follows, we will construct a channel model, namely, a CTMC, from the traces in the first sub-section. We have developed a simulator to simulate our tandem fluid queue model. Based on the trace-driven channel models, in the second sub-section, we will compare our analytical performance model in terms of the average response time with the results from the fluid queue simulator to evaluate the accuracy of our analytical performance model. In the last sub-section, we will present our instance selection with different values of the desired average response time.

A. Constructing Channel Models from Traces

Each trace in [17] has a duration of 15 minutes and records the throughput of the wireless channel in a granularity of seconds. We consider that the throughput evolves according to a CTMC, and we aim to construct its intensity matrix Q and the corresponding rate vector $\boldsymbol{r}$. To this end, we first cluster the throughput values into K groups by the K-means clustering algorithm. Each cluster represents one state of the wireless channel, and the average throughput of clusters constitute the rate vector $\boldsymbol{r}$. We then construct the intensity matrix Q as

$$Q_{ij} = \begin{cases} \frac{\text{Number of jumps from cluster } i \text{ to cluster } j}{\text{Cumulative time spent in state i}} & i \neq j, \\ -\sum_{j \neq i} Q_{ij} & i = j. \end{cases}$$



(a) Fixing $c_l = 0$. (b) Fixing $1/\alpha = 500$ msecs.

Fig. 6: Instance type selections–burstable instance (BI) or static instance (SI)–for the scenarios with different QoS requirements τ under the Gilbert-Elliott channel model.

Through an extensive grid search, we choose $K = 10$. We will show in the next sub-section that this K value, although not high, can already produce good accuracy with our analytical performance model.

B. Validating Performance Model

We first choose three trace-driven channel models for when users are in each of the following mobility patterns: static, pedestrian, on a bus, and in a car. We fix the net input workload $\lambda - c_\ell = 2$ Mbps.[7] For each channel model, we derive the approximate offloaded demand according to Equations (27)–(32) and then evaluate the average response time based on fluid flow analysis and Equations (33)–(35).

Figure 7 shows the approximate average response time obtained by our performance model in comparison with the simulated results from the fluid queue simulator under the same channel model. It can be observed that our approximate average response time is close to the simulated results for all of the 12 channel models. In Figure 7(a), we show the comparison among the three channel models derived from traces when users are static. The SCVs of the offloaded demand from these three models are 0.07, 0.38, and 0.55, respectively. In other words, the degrees of burstiness in scenarios I, II, and III are increasing. Consequently, given the same execution capacity, the average response times in the three scenarios are also increasing. Figures 7(b)–(d) show a comparison between the numerical results from our analytical approximation and the simulation when users are pedestrians, on a bus, and in a car, respectively. The average response times for mobile users are generally longer than those for static users because the move-and-stop patterns of mobile users can generally lead to larger variations in the throughput of the wireless channels and result in more bursty offloaded demand.

To further show the robustness of our analytical approximation, we evaluate the relative error of the approximation. For each trace-driven wireless channel that satisfies the stability condition (Equation (2)) of the local data queue, we derive the relative error by

$$\text{Relative error} = \frac{|\text{Analytical value} - \text{Simulated value}|}{|\text{Simulated value}|}.$$

[7]Based on the default scheduling policy described in Section III-A, the average response time depends only on the net input rate $\lambda - c_\ell$.

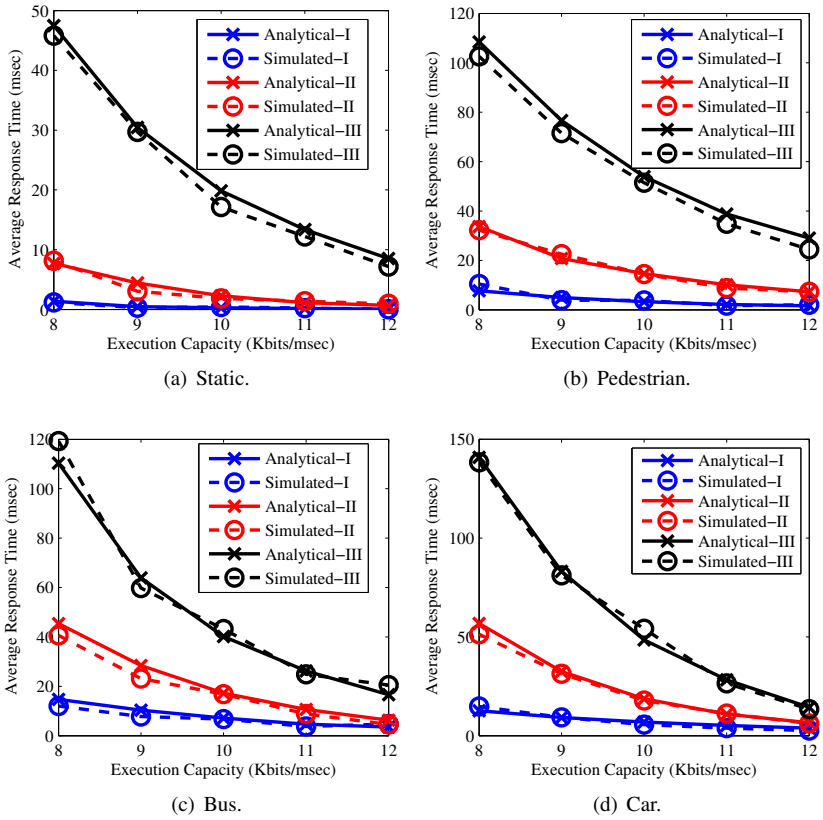
Figure 8 shows the empirical cumulative distribution function (CDF) of the relative error when the users are under different mobility patterns. The 90-th percentiles of the four mobility patterns, static, pedestrian, bus, and car, are 0.15, 0.17, 0.18, and 0.20, respectively. Therefore, for each mobility pattern, the relative error for 90% of the scenarios is below 0.20. It can also be observed from Figure 8 that our analytical model is more accurate under the scenarios when users are static than the scenarios when users are moving (pedestrian, car, bus) with higher throughput variation.

C. Instance Type Selection

In this sub-section, we continue on to present the economical instance selection to provision the required resource capacity for a desired QoS, as derived by our proposed analytical performance model. For different scenarios of the desired QoS and configuration of the wireless channel fitted from the traces, we plot the selected cloud instance types in Figure 9. It can be observed that under realistic channel models, burstable instances are selected for most of the scenarios. Also, the scenarios with stringent QoS requirements, labeled by circles in Figure 9, have low capacity utilizations and tend to have burstable instances be the better choice. Moreover, the scenarios with more bursty demand (i.e., higher SCVs) are also more likely to have burstable instances be the better choice to save monetary costs.

VII. Conclusions and Future Work

This paper considers a mobile cloud computing system. The computation workload that cannot be tackled locally on an end device is offloaded to a backend cloud instance for execution via a wireless communication channel. The stochasticity in the wireless channel leads to burstiness in the workload offloaded to the backend cloud. Our target question is how to provision resources at the backend cloud to satisfy a desired QoS, namely, the average response time. To this end, we model the mobile cloud computing system as a tandem queue network and analyze this model using the fluid flow analysis framework as follows: We first quantify the output process of the first queue, which characterizes the offloading process (Section III). We then turn to the second queue, which characterizes the workload execution process at the



(a) Static. (b) Pedestrian.

(c) Bus. (d) Car.

Fig. 7: Comparison of the average response times obtained by our analytical performance model and simulations for the channels when mobile users are static, pedestrians, on a bus, and in a car.

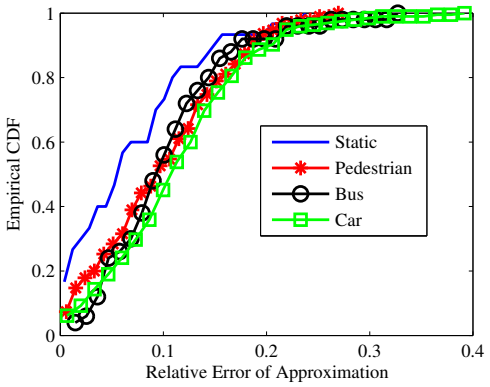


Fig. 8: The empirical cumulative distribution function (CDF) of the relative error under different mobility patterns.

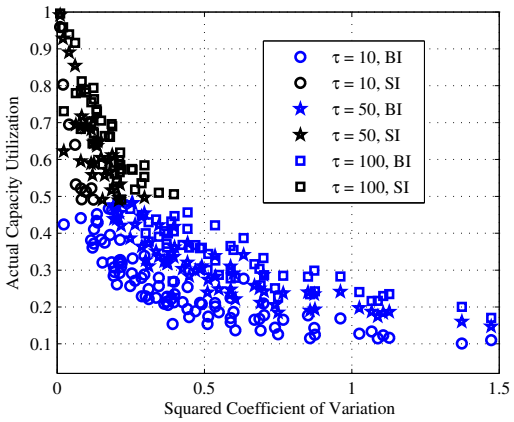


Fig. 9: Instance type selections—burstable instance (BI) or static instance (SI)—for the scenarios with different QoS requirements and channel models fitted from traces.

backend cloud, and derive the analytical relationship between the available resource capacity for workload execution at the backend cloud and the achieved average response time (Section IV). Based on our derived performance model, we finally determine whether subscribing to a traditional static instance or a burstable instance is more economical with the user's desired average response time guaranteed (Section V). Extensive trace-driven simulations validate our performance model and demonstrate the final selection of instance type according to different wireless channel conditions and QoS requirements (Section VI).

In the future, we plan to further generalize our performance model and analysis to evaluate the average response time when multiple users simultaneously offload computation demand to the same (congested) backend cloud. Meanwhile, based on the performance model derived in this paper, we will further investigate how to derive an optimal offloading policy to best trade-off the local computing and offloading costs, as well as the workload delay from the user's perspective.

REFERENCES

[1] B. Sun, Y. Jiang, and D. H. K. Tsang, "When burstable instances meet mobile computing: performance modeling and economic analysis," in *Proc. IEEE Int. Conf. Distrib. Comput. Syst. (ICDCS)*, 2020, pp. 1179–1180.

[2] J. Liu, Y. Mao, J. Zhang, and K. B. Letaief, "Delay-optimal computation task scheduling for mobile-edge computing systems," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, 2016, pp. 1451–1455.

[3] ——, "Offloading in mobile cloudlet systems with intermittent connectivity," *IEEE Trans. Mob. Comput.*, vol. 14, no. 12, pp. 2516–2529, 2015.

[4] J. Zheng, Y. Cai, Y. Wu, and X. Shen, "Dynamic computation offloading for mobile cloud computing: a stochastic game-theoretic approach," *IEEE Trans. Mob. Comput.*, vol. 18, no. 4, pp. 771–786, 2019.

[5] G. Qu, H. Wu, R. Li, and P. Jiao, "DMRO: A deep meta reinforcement learning-based task offloading framework for edge-cloud computing," *IEEE Trans. Netw. Service Manag.*, vol. 18, no. 3, pp. 3448–3459, 2021.

[6] H. Huang, Q. Ye, and H. Du, "Reinforcement learning based offloading for realtime applications in mobile edge computing," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2020, pp. 1–6.

[7] G. Zhang, Y. Chen, Z. Shen, and L. Wang, "Distributed energy management for multiuser mobile-edge computing systems with energy harvesting devices and QoS constraints," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4035–4048, 2019.

[8] Y. Wu, K. Ni, C. Zhang, L. P. Qian, and D. H. K. Tsang, "NOMA assisted multi-access mobile edge computing: a joint optimization of computation offloading and time allocation," *IEEE Trans. Veh. Technol.*, vol. 67, no. 12, pp. 12 244–12 258, 2018.

[9] 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 Trans. Wireless Commun.*, vol. 16, no. 9, pp. 5994–6009, 2017.

[10] S. Guo, D. Wu, H. Zhang, and D. Yuan, "Resource modeling and scheduling for mobile edge computing: A service provider's perspective," *IEEE Access*, vol. 6, pp. 35 611–35 623, 2018.

[11] L. Lei, H. Xu, X. Xiong, K. Zheng, and W. Xiang, "Joint computation offloading and multiuser scheduling using approximate dynamic programming in NB-IoT edge computing system," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 5345–5362, 2019.

[12] H. Shah-Mansouri and V. W. S. Wong, "Hierarchical fog-cloud computing for IoT systems: A computation offloading game," *IEEE Internet Things J.*, vol. 5, no. 4, pp. 3246–3257, 2018.

[13] J. Tang, J. Nie, Z. Xiong, J. Zhao, Y. Zhang, and D. Niyato, "Slicing-based reliable resource orchestration for secure software defined edge-cloud computing systems," *IEEE Internet Things J.*, 2021, DOI: 10.1109/JIOT.2021.3107490.

[14] C. You, K. Huang, H. Chae, and B.-H. Kim, "Energy-efficient resource allocation for mobile-edge computation offloading," *IEEE Trans. Wireless Commun.*, vol. 16, no. 3, pp. 1397–1411, 2016.

[15] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Trans. Netw.*, vol. 24, no. 5, pp. 2795–2808, 2015.

[16] 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, 2012.

[17] D. Raca, J. J. Quinlan, A. H. Zahran, and C. J. Sreenan, "Beyond throughput: A 4G LTE dataset with channel and context metrics," in *Proc. 9th ACM Multimedia Syst. Conf.*, 2018, pp. 460–465.

[18] H. Chen and D. D. Yao, *Fundamentals of queueing networks: performance asymptotics and optimization*. New York:Springer-Verlag, 2001, vol. 46.

[19] A. Elwalid and D. Mitra, "Analysis, approximations and admission control of a multi-service multiplexing system with priorities," in *Proc. IEEE Int. Conf. Comput. Commun. (INFOCOM)*, vol. 2, 1995, pp. 463–472.

[20] A. I. Elwalid and D. Mitra, "Analysis and design of rate-based congestion control of high speed networks, i: stochastic fluid models, access regulation," *Queueing Syst.*, vol. 9, no. 1, pp. 29–63, 1991.

[21] Q. Zhang, A. Heindl, and E. Smirni, "Models of the departure process of a bmap/map/1 queue," *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 33, no. 2, pp. 18–20, 2005.

[22] S. R. Mahabhashyam and N. Gautam, "On queues with markov modulated service rates," *Queueing Syst.*, vol. 51, no. 1, pp. 89–113, 2005.

[23] "Microsoft azure: B-series burstable virtual machine sizes," 2021. [Online]. Available: https://docs.microsoft.com/en-us/azure/virtual-machines/linux/b-series-burstable

[24] "Amazon ec2: Burstable performance instances," 2021. [Online]. Available: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/burstable-performance-instances.html

[25] Y. Jiang, M. Shahrad, D. Wentzlaff, D. H. K. Tsang, and C. Joe-Wong, "Burstable instances for clouds: Performance modeling, equilibrium analysis, and revenue maximization," *IEEE/ACM Trans. Netw.*, vol. 28, no. 6, pp. 2489–2502, 2020.

[26] H. Guo, J. Liu, J. Zhang, W. Sun, and N. Kato, "Mobile-edge computation offloading for ultradense IoT networks," *IEEE Internet Things J.*, vol. 5, no. 6, pp. 4977–4988, 2018.

[27] K. Peng, J. Nie, N. Kumar, C. Cai, J. Kang, Z. Xiong, and Y. Zhang, "Joint optimization of service chain caching and task offloading in mobile edge computing," *Applied Soft Comput.*, vol. 103, p. 107142, 2021.

[28] H. Yang, A. Alphones, Z. Xiong, D. Niyato, J. Zhao, and K. Wu, "Artificial-intelligence-enabled intelligent 6G networks," *IEEE Netw.*, vol. 34, no. 6, pp. 272–280, 2020.

[29] D. Mitra, "Stochastic theory of a fluid model of producers and consumers coupled by a buffer," *Advanc. Appl. Prob.*, vol. 20, pp. 646–676, 1988.

[30] W. R. W. Scheinhardt, *Markov modulated and feed-back fluid queues*. Ph.D. Thesis, Faculty of Mathematical Sciences. University of Twente. AE Enschede. The Netherlands, 1998.

[31] L. Huang and T. T. Lee, "Generalized Pollaczek-Khinchin formula for markov channels," *IEEE Trans. Commun.*, vol. 61, no. 8, pp. 3530–3540, 2013.

[32] P. Leitner and J. Scheuner, "Bursting with possibilities an empirical study of credit-based bursting cloud instance types," in *Proc. IEEE/ACM Int. Conf. Utility Cloud Comput.*, 2015, pp. 227–236.

[33] C. Wang, B. Urgaonkar, N. Nasiriani, and G. Kesidis, "Using burstable instances in the public cloud: Why, when and how?" in *Proc. ACM Meas. Annu. Comput. Syst.*, vol. 1, no. 1, 2017, pp. 11:1–11:28.

[34] C. Wang, B. Urgaonkar, A. Gupta, G. Kesidis, and Q. Liang, "Exploiting spot and burstable instances for improving the cost-efficacy of in-memory caches on the public cloud," in *Proc. Eur. Conf. Comput. Syst.*, 2017, pp. 620–634.

[35] A. F. Baarzi, T. Zhu, and B. Urgaonkar, "BurScale: Using burstable instances for cost-effective autoscaling in the public cloud," in *Proc. ACM Symp. Cloud Comput.*, 2019, pp. 126–138.

[36] A. Ali, R. Pinciroli, F. Yan, and E. Smirni, "CEDULE: A scheduling framework for burstable performance in cloud computing," in *Proc. IEEE Int. Conf. Autonomic Comput. (ICAC)*, 2018, pp. 141–150.

[37] L. Kleinrock, *Queueing systems: theory*. John Wiley, 1975.

[38] D. Anick, D. Mitra, and M. M. Sondhi, "Stochastic theory of a data-handling system with multiple sources," *Bell Syst. Tech. J.*, vol. 61, no. 8, pp. 1871–1894, 1982.

[39] E. G. Coffman, B. M. Igelnik, and Y. A. Kogan, "Controlled stochastic model of a communication system with multiple sources," *IEEE Trans. Inform. Theory*, vol. 37, no. 5, pp. 1379–1387, 1991.

[40] R. Deng, R. Lu, C. Lai, T. H. Luan, and H. Liang, "Optimal workload allocation in fog-cloud computing toward balanced delay and power consumption," *IEEE Internet Things J.*, vol. 3, no. 6, pp. 1171–1181, 2016.

[41] L. Tong, Y. Li, and W. Gao, "A hierarchical edge cloud architecture for mobile computing," in *Proc. IEEE Int. Conf. Comput. Commun. (INFOCOM)*, 2016, pp. 1–9.

[42] J. Xu, Z. Chen, J. Tang, and S. Su, "T-storm: Traffic-aware online scheduling in storm," in *Proc. IEEE Int. Conf. Distrib. Comput. Syst. (ICDCS)*, 2014, pp. 535–544.

[43] Y. Jiang, Z. Huang, and D. H. K. Tsang, "Towards max-min fair resource allocation for stream big data analytics in shared clouds," *IEEE Trans. Big Data*, vol. 4, no. 1, pp. 130–137, 2018.

[44] P. Sadeghi, R. A. Kennedy, P. B. Rapajic, and R. Shams, "Finite-state Markov modeling of fading channels: A survey of principles and applications," *IEEE Signal Process. Mag.*, vol. 25, no. 5, pp. 57–80, 2008.

[45] Q. Zhang and S. A. Kassam, "Finite-state Markov model for Rayleigh fading channels," *IEEE Trans. Commun.*, vol. 47, no. 11, pp. 1688–1692, 1999.

[46] H. C. Tijms, *A first course in stochastic models*. 3rd ed. John Wiley & Sons, 2003.

[47] L. Huang, S. Bi, and Y.-J. A. Zhang, "Deep reinforcement learning for online computation offloading in wireless powered mobile-edge computing networks," *IEEE Trans. Mob. Comput.*, vol. 19, no. 11, pp. 2581–2593, 2019.

[48] E. N. Gilbert, "Capacity of burst-noise channels," *Bell Syst. Tech. J.*, vol. 39, pp. 1253–1265, 1960.

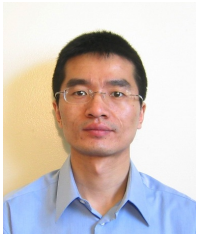
[49] L. A. Johnston and V. Krishnamurthy, "Opportunistic file transfer over a fading channel: a POMDP search theory formulation with optimal threshold policies," *IEEE Trans. Wireless Commun.*, vol. 5, no. 2, pp. 394–405, 2006.

[50] W. Zhang, Y. Wen, K. Guan, D. Kilper, H. Luo, and D. O. Wu, "Energy-optimal mobile cloud computing under stochastic wireless channel," *IEEE Trans. Wireless Commun.*, vol. 12, no. 9, pp. 4569–4581, 2013.
[51] J.-P. Ebert, A. Willig *et al.*, "A Gilbert-Elliott bit error model and the efficient use in packet level simulation," 1999. [Online]. Available: http://www.tkn.tu-berlin.de/awillig/
[52] S. Ross, *Stochastic Process, 2nd Edition*. Hoboken:Wiley, 1996.
[53] "EC2 on-demand instance pricing - Amazon Web Services," 2022. [Online]. Available: https://aws.amazon.com/ec2/pricing/on-demand/
[54] D. Lučanin, I. Pietri, S. Holmbacka, I. Brandic, J. Lilius, and R. Sakellariou, "Performance-based pricing in multi-core geo-distributed cloud computing," *IEEE Trans. Cloud Comput.*, vol. 8, no. 4, pp. 1079–1092, 2016.

Bo Sun (Member, IEEE) is a Postdoctoral Fellow at The Hong Kong University of Science and Technology (HKUST). He received his B.E. degree from Harbin Institute of Technology, Harbin, China, in 2013, and his Ph.D. degree from HKUST in 2018. His research focuses on online optimization, online algorithms, and decision-making under uncertainty with applications to real-world networked systems.

Yuxuan Jiang (Member, IEEE) received his Ph.D. degree from The Hong Kong University of Science and Technology (HKUST) in 2019 and his B.Eng. degree from Huazhong University of Science and Technology, Wuhan, China, in 2013. He is a Postdoctoral Fellow at Dalhousie University, Canada. His research interests lie in the broad area of networking and distributed systems.

Yuan Wu (Senior Member, IEEE) received the Ph.D. degree in Electronic and Computer Engineering from the Hong Kong University of Science and Technology in 2010. He is currently an Associate Professor with the State Key Laboratory of Internet of Things for Smart City, University of Macau and also with the Department of Computer and Information Science, University of Macau. During 2016-2017, he was a visiting scholar with Department of Electrical and Computer Engineering, University of Waterloo. His research interests include resource management for wireless networks, green communications and computing, mobile edge computing and edge intelligence. He was a recipient of the Best Paper Award from the IEEE International Conference on Communications in 2016, and the Best Paper Award from IEEE Technical Committee on Green Communications and Computing in 2017. Dr. Wu is currently on the Editorial Boards of IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, IEEE TRANSACTIONS ON NETWORK SCIENCE AND ENGINEERING, IEEE INTERNET OF THINGS JOURNAL, and IEEE OPEN JOURNAL OF THE COMMUNICATIONS SOCIETY.



Qiang Ye (Member, IEEE) is a Professor in the Faculty of Computer Science at Dalhousie University, Canada. His current research interests lie in the area of communication networks in general. Specifically, he is interested in Wireless Networks, Mobile Computing, Internet of Things (IoT), Network Security, and Machine Learning/Data Analytics. He has published a series of papers in top publication venues such as IEEE/ACM Transactions on Networking (TON), IEEE Transactions on Parallel and Distributed Systems (TPDS), and IEEE Transactions on Wireless Communications (TWC). He received a Ph.D. in Computing Science from the University of Alberta in 2007. His M. Engr. and B. Engr. in Computer Science and Technology are from Harbin Institute of Technology, China. He is a Member of IEEE and ACM.

Danny H.K. Tsang (Fellow, IEEE) received the Ph.D. degree in electrical engineering from the Moore School of Electrical Engineering at the University of Pennsylvania, U.S.A., in 1989. Upon graduation, he joined the Department of Computer Science at Dalhousie University in Canada. He later joined the Department of Electronic and Computer Engineering at The Hong Kong University of Science and Technology (HKUST) in 1992 and is now a Professor in the department. He has also served as Professor and the Founding Acting Thrust Head of the Internet of Things Thrust of the HKUST (Guangzhou) since March 2020. He was a Guest Editor for IEEE Journal on Selected Areas in Communications' special issue on Advances in P2P Streaming Systems, an Associate Editor for Journal of Optical Networking published by the Optical Society of America, and a special issue Guest Editor for IEEE Systems Journal, Transactions on Industrial Informatics, Communications Magazine, and Network Magazine. He currently serves as Technical Editor for IEEE Communications Magazine. He was nominated to become an IEEE Fellow in 2012 and an HKIE Fellow in 2013. During his leave from HKUST in 2000-2001, Dr. Tsang assumed the role of Principal Architect at Sycamore Networks in the United States. He was responsible for the network architecture design of Ethernet MAN/WAN over SONET/DWDM networks. He invented the 64B/65B encoding (US Patent No.: US 6,952,405 B2) and contributed it to the proposal for Transparent GFP in the T1X1.5 standard that was advanced to become the ITU G.GFP standard. The coding scheme has now been adopted by International Telecommunication Union (ITU)'s Generic Framing Procedure recommendation GFP-T (ITU-T G.7041/Y.1303)) and Interfaces for the Optical Transport Network (ITU-T G.709/Y1331). His current research interests include cloud computing, edge computing, NOMA networks, online algorithm design, and smart grids.