Cost-Efficient Resource Provisioning for Dynamic Requests in Cloud Assisted Mobile Edge Computing

Xiao Ma, *Member, IEEE*, Shangguang Wang, *Senior Member, IEEE*, Shan Zhang, *Member, IEEE*, Peng Yang, *Member, IEEE*, Chuang Lin, *Senior Member, IEEE*, and Xuemin Shen, *Fellow, IEEE*

Abstract—Mobile edge computing is emerging as a new computing paradigm that provides enhanced experience to mobile users via low latency connections and augmented computation capacity. As the amount of user requests is time-varying, while the computation capacity of edge hosts is limited, Cloud Assisted Mobile Edge (CAME) computing framework is introduced to improve the scalability of the edge platform. By outsourcing mobile requests to clouds with various types of instances, the CAME framework can accommodate dynamic mobile requests with diverse quality of service requirements. In order to provide guaranteed services at minimal system cost, the edge resource provisioning and cloud outsourcing of the CAME framework should be carefully designed in a cost-efficient manner. Specifically, two fundamental issues should be answered: (1) what is the optimal edge computation capacity configuration? and (2) what types of cloud instances should be tenanted and what is the amount of each type? To solve these issues, we formulate the resource provisioning in CAME framework as an optimization problem. By exploiting the piecewise convex property of this problem, the Optimal Resource Provisioning (ORP) algorithms with different instances are proposed, so as to optimize the computation capacity of edge hosts and meanwhile dynamically adjust the cloud tenancy strategy. The proposed algorithms are proved to be with polynomial computational complexity. To evaluate the performance of the ORP algorithms, extensive simulations and experiments are conducted based on both the widely-used traffic models and the Google cluster usage tracelogs, respectively. It is shown that the proposed ORP algorithms outperform the local-first and cloud-first benchmark algorithms in system flexibility and cost-efficiency.

Index Terms—mobile edge computing, resource provisioning, computation offloading

1 Introduction

With the popularity of intelligent devices, substantial computation-intensive delay-sensitive mobile applications keep emerging and have drawn enormous attentions, e.g., face recognition and augmented reality [1], [2]. To meet the quality of service (QoS) requirements of those applications, sufficient computation capacity should be provided. Nevertheless, mobile devices are usually resource-constrained in terms of computation capacity and battery lifetime. Mobile cloud computing has been proposed to deal with this conflict [3], [4]. In mobile cloud computing, mobile requests are transmitted through the wireless access network, and are further delivered to remote clouds through the Internet backhaul. However, the wide area network delay and jitters incurred during computation offloading are unpredictable, which may violate the QoS requirements of delay-sensitive applications, such as online gaming [9]. In addition, with the prevalence of mobile computing (e.g., virtual reality and augmented reality), offloading all the mobile requests to remote clouds will induce substantial communication and computation burden, which is beyond the capacity of the Internet core network and centralized data centers, respectively [6].

Mobile edge computing (MEC) is an ideal paradigm to address these problems. By deploying edge hosts within the wireless access network, mobile users are able to access sufficient computation resources without suffering from the uncontrollable Internet delay [7]. Due to the advantage of low latency, extensive efforts have been devoted to the potential applications of MEC [13], [14], [15], [16], [17], [18], [19]. In MEC, computation resources of edge hosts are pre-configured, yet computation requests of mobile users fluctuate significantly both in the short and long run [20]. Conventionally, computation resources are usually over-provisioned to provide guaranteed performance, leading to high cost and low resource utilization. For example, more than 10,000 computers should be provisioned to guarantee the operation of the online game World of Warcraft [21]. How to achieve cost-efficient resource provisioning for dynamic requests is crucial to mobile edge operators.

- *X. Ma and S. Wang are with the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China. E-mail: maxiaoxiao13@gmail.com, sgwang@bupt.edu.cn.*
- *S. Zhang is with the School of Computer Science and Engineering, Beihang University, Beijing 100191, China. E-mail: zhangshan18@buaa.edu.cn.*
- *P. Yang and X. Shen are with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, Ontario N2L 3G1, Canada. E-mail: {p38yang, sshen}@uwaterloo.ca.*
- *C. Lin is with the Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China. E-mail: chlin@tsinghua.edu.cn.*

Manuscript received 19 Sept. 2018; revised 10 Feb. 2019; accepted 23 Feb. 2019. Date of publication 5 Mar. 2019; date of current version 3 Sept. 2021. (Corresponding author: Shan Zhang.)
Recommended for acceptance by B. Di Martino.
Digital Object Identifier no. 10.1109/TCC.2019.2903240

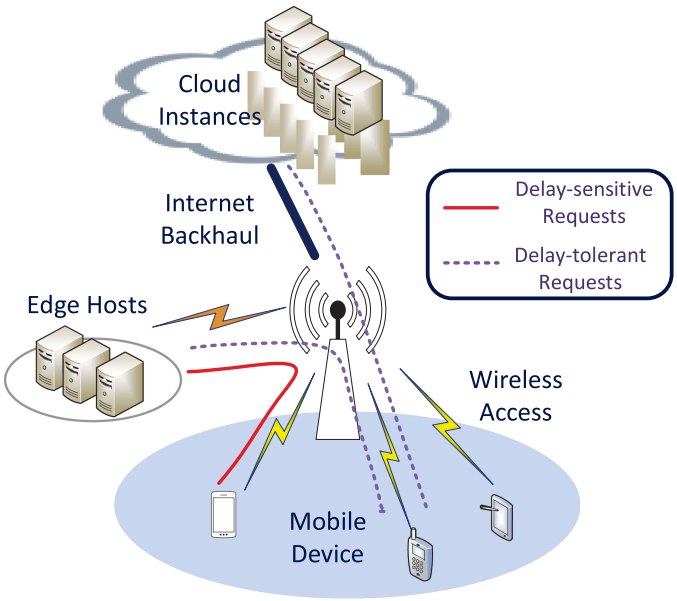


Fig. 1. Resource provisioning in CAME.

To cater dynamic requests with cost-efficiency, this work adopts the CAME framework, in which fixed edge hosts are deployed and elastic cloud resources are dynamic provisioned to accommodate dynamic requests [11]. Typically, commercial public computation resources are provided in elastic encapsulated instances (e.g., Amazon EC2,[1] Microsoft Azure[2] and Google Cloud Platform[3]). On-demand instances, reserved instances and spot instances are the main cloud instances, which are differentiated by pricing schemes [29]. The system cost of the CAME framework consists of two parts: *edge provisioning cost*, cost of deploying edge hosts; *cloud tenancy cost*, cost of tenanting instances from public clouds. As computation capacity of edge hosts is fixed and may not be sufficient, which directly affects system cost and QoS performance, two fundamental questions should be answered to achieve cost-efficiency: (1) What's the optimal edge computation capacity configuration? (2) What types of cloud instances should be tenanted and what's the amount of each type?

This paper addresses the above issues by investigating the resource provisioning problem with dynamic requests. Mobile requests are considered to be with diversified QoS requirements (including delay-sensitive and delay-tolerant requests), as shown in Fig. 1. Edge hosts execute all delay-sensitive and part of the delay-tolerant requests, and cloud instances are dynamically tenancted to serve the outsourced delay-tolerant requests. Among the three types of cloud instances, spot instances are not taken into consideration due to the possibility of service interruption [29]. Thus the resource provisioning problem in CAME can be divided into three categories: with on-demand instances, with reserved instances and hybrid strategy.

We solve the resource provisioning problem by formulating an optimization problem that aims at minimizing the long-term average system cost. Resource provisioning with single-type cloud instances (i.e., with on-demand instances and with reserved instances) is first investigated, based on which resource provisioning with hybrid strategy is further explored. Based on queuing analysis, it is proved that this optimization problem is piecewise convex. By exploiting this property, the Optimal Resource Provisioning (ORP) algorithms are developed to deal with different categories, i.e., Optimal Resource Provisioning with On-Demand instances (ORP-OD), Optimal Resource Provisioning with Reserved instances (ORP-R) and Optimal Resource Provisioning with Hybrid Strategy(ORP-HS), respectively. In specific, the optimal computation capacity of edge hosts is determined by solving convex optimization problems in each segment of the resource provisioning problem. Then, the cloud tenancy strategy is further devised according to users' QoS requirements.

The main contributions of this work are as follows.

- The CAME framework is exploited to achieve cost-efficient resource provisioning for dynamic mobile requests. Considering different types of cloud instances, the resource provisioning problem is divided into three categories: with on-demand instances, with reserved instances and hybrid strategy. Based on queuing analysis, the resource provisioning problem is formulated as an optimization problem, which is proved to be piecewise convex.
- By leveraging the piecewise convex property, the Optimal Resource Provisioning (ORP) algorithms are designed. Convex optimization problems are solved in each segment to determine the optimal computation capacity of edge hosts, and cloud tenancy strategy is further optimized based on the QoS requirements of mobile requests.
- To evaluate the ORP algorithms, extensive simulations based on two widely-used traffic models and Google cluster tracelogs are implemented. Results demonstrate that the ORP algorithms outperform other benchmark algorithms in system flexibility and cost-efficiency.

This paper is organized as follows. Section 2 discusses the related work. In Section 3, the system model is presented and problem formulation is further described. Algorithm design is provided in Section 4, and simulations and analysis are presented in Section 5. Finally, the conclusion is given in Section 6.

2 Related Work

Mobile cloud computing provides an ideal solution to alleviating the conflict between resource-intensive mobile applications and resource-constrained mobile devices. Design and implementation of systems that support code computation or computation offloading have been presented in [3], [4]. Computation offloading problems have then been studied in extensive works. A user-oriented offloading framework has been proposed in [22] to implement efficient online computation offloading and the conditions in which mobile users can be benefited from computation offloading have been investigated in [23], [24]. However, uncontrollable WAN delay and jitters induced in conventional mobile cloud computing can significantly degrade the QoS for mobile users.

1. https://aws.amazon.com/cn/ec2/.
2. https://azure.microsoft.com/en-us/pricing/.
3. https://cloud.google.com/pricing/.

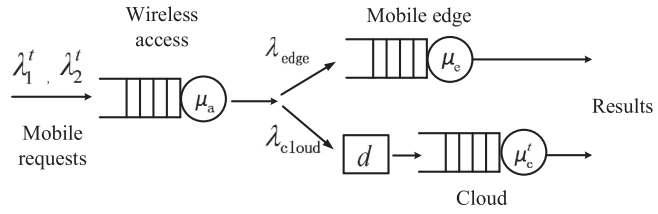


Fig. 2. Analytical model.

MEC is envisioned as an effective paradigm to address this challenge, by deploying edge hosts within wireless access network and in close proximity to mobile devices. Many efforts have been devoted to fully exploiting the advantages of MEC. Workload scheduling problems in MEC have been well studied in substantial existing works. Tong et al. have designed a hierarchical edge cloud architecture to schedule the peak mobile workloads [25]. Workload placement and resource provisioning decisions have further been made to achieve high resource utilization. Sun et al. have proposed the LEARN algorithm for the replica placement of Avatar virtual disks in the cloudlet network [12]. With this algorithm, adaptive Avatar handoff can be achieved with low E2E latency.

In MEC, the resource provisioning problem is also a critical issue since the computation capacity of edge hosts is fixed and constrained. In [15], a service-oriented resource management framework has been proposed for fog computing, in which resource prediction, resource reservation and pricing can be addressed. Kiani et al. have explored hierarchical capacity provisioning in edge computing [17]. A 2-tier edge computing architecture has been proposed and the capacity provisioning problems have been solved by stochastic ordering and optimization problems. Enayet et al. [26] have presented Mobi-Het, a mobility-aware resource allocation architecture, to execute tasks at remote cloudlets with high efficiency and reliability. In existing works of resource provisioning for MEC, dynamics of mobile requests have not been taken into consideration, which can incur overprovisioning of edge hosts and low resource utilization. Federating the internal infrastructures with public clouds provides an effective solution to augment system computation capacity and provision elastic resources for dynamic requests [27], [28]. Cloud-fog interoperation has been explored in an SDN enabled framework, which aims at improving the QoS and optimizing the utilization of network resources [30]. In our previous work, the CAME framework has been proposed to augment the computation capacity of MEC by tenanting computation resources from public clouds [11]. In this paper, to achieve cost-efficient resource provisioning for dynamic requests, we investigate the resource provisioning problem in CAME.

In the CAME framework, the flexibility of MEC with dynamic requests is enhanced by exploiting the scalability of cloud resources. Different from most existing works in which the computation capacity of edge hosts is assumed to be given, we aim to investigate the optimal edge computation capacity and further dynamically tune the usage of cloud resources. The preliminary results of resource provisioning with on-demand instances have demonstrated the effectiveness of the CAME framework in enhancing the scalability of system resources [32]. In this paper, the resource provisioning problem for both delay-tolerant and delay-sensitive mobile requests is investigated. Furthermore, to fully exploit the advantages of different cloud instances (e.g., availability, economy and scalability), three resource provisioning categories (i.e., with on-demand instances, with reserved instances and hybrid strategy) are studied. By addressing this resource provisioning problem, different QoS requirements of mobile applications can be satisfied at the minimum system cost.

TABLE 1
Table of Notations

Notation	Description
T	The number of time intervals
λ_1^t	Average arrival rate of delay-sensitive requests
λ_2^t	Average arrival rate of delay-tolerant requests
μ_{a}	Average transmission rate of wireless access network
μ_{e}	Serving rate of mobile edge
μ_{c}^t	Serving rate of cloud instances
d	WAN latency from mobile edge to the cloud
D_{comm}^t	Transmission delay in wireless access network
μ_{s}^t	Computation rate to serve delay-sensitive requests
D_{comp}^t	Computation delay of delay-tolerant requests
D^t	System delay in the tth interval
D_1	Delay constraint of delay-sensitive mobile requests
D_2	Delay constraint of delay-tolerant mobile requests
c_p	Pricing rate of on-demand instances
μ_{o}^t	Serving rate of on-demand instances
r	Discount ratio of reserved over on-demand instances
μ_{r}	Serving rate of reserved instances
$S(\mu_{\mathrm{e}}, \mu_{\mathrm{c}}^t)$	System cost

3 System Model and Problem Formulation

In the resource provisioning problem, Mobile requests are considered to be with different QoS requirements (including delay-sensitive and delay-tolerant requests). As shown in Fig. 1, all delay-sensitive requests are transmitted to edge hosts via the wireless access network, while part of delay-tolerant requests can further be outsourced to the remote cloud through Internet backhaul [10]. The request scheduling issue has been well studied in [11], [12], which is not the focus of this work. Delay-tolerant requests are allocated to edge hosts and cloud instances in proportion to the available computation capacity of each part. A set of T intervals is considered as a cycle (e.g., there are 24 intervals in a cycle for daily requests). Non-homogeneous Poisson traffic model is adopted to characterize the dynamic delay-sensitive and delay-tolerant requests, with the average arrival rates of λ_1^t, λ_2^t ($t \in \{1, 2, \ldots, T\}$) respectively [33]. Then the CAME framework can be modeled as a queuing network, as presented in Fig. 2. When offloading mobile requests to edge hosts or further outsourced to the cloud, additional communication requests (program codes or data) are incurred. μ_{a} represents the transmission rate of communication requests in the wireless access network, and d is the WAN latency from mobile edge to the cloud. Denote by μ_{e} the computation capacity of edge hosts and μ_{c}^t represents the serving rate of cloud instances. Then the system delay can be calculated based on queuing theory [34]. The notations are listed in Table 1.

3.1 System Delay

In the wireless access network, the communication traffic is related to mobile computation requests. For example, when

computation requests of a face recognition request arrive at edge hosts, the figure or data including features extracted from the figure should also be transmitted via the wireless access network. Thus, the arrival rate of communication requests in the access network can be represented as $\rho_1(\lambda_1^t) + \rho_2(\lambda_2^t)$, where $\rho_1(\lambda_1^t)$ and $\rho_2(\lambda_2^t)$ are functions increasing with λ_1^t and λ_2^t, respectively ($\rho_1(\lambda_1^t)$ can be a linear function of λ_1^t, which means c_1 units of data should be transmitted when one unit of delay-sensitive requests is offloaded to edge hosts from mobile devices [11]). Denote by D_{comm}^t the communication delay in the wireless access network. Then, D_{comm}^t can be calculated as

$$D_{\text{comm}}^t = \frac{1}{\mu_{\text{a}} - [\rho_1(\lambda_1^t) + \rho_2(\lambda_2^t)]}, \tag{1}$$

where $\mu_{\text{a}} > \rho_1(\lambda_1^t) + \rho_2(\lambda_2^t)$ to ensure the system stability.

To satisfy the QoS requirements, all delay-sensitive requests should be executed at edge hosts, with higher priority compared with delay-tolerant requests. Suppose the service time of edge hosts and cloud instances are exponentially distributed, then the service processes of mobile edge and cloud can be modeled as M/M/1 queues in each time interval [31]. Thus the serving rate demanded to execute delay-sensitive requests in the tth time interval is

$$\mu_{\text{s}}^t = \frac{1}{D_1 - \frac{1}{\mu_{\text{a}} - [\rho_1(\lambda_1^t) + \rho_2(\lambda_2^t)]}} + \lambda_1^t, \tag{2}$$

where D_1 represents the QoS requirement of the delay-sensitive mobile requests. As all delay-sensitive mobile requests are executed at mobile edge, it holds that

$$\mu_{\text{e}} \geq \max\{\mu_{\text{s}}^t, t = 1, 2, ...T\}. \tag{3}$$

When no computation capacity is provisioned for delay-tolerant requests at edge nodes (i.e., $\mu_{\text{e}} = \mu_{\text{s}}^t$), all delay-tolerant requests are outsourced to the cloud and the computation delay is $\frac{1}{\mu_{\text{c}}^t - \lambda_2^t} + d$. When no cloud instances are tenanted (i.e., $\mu_{\text{c}}^t = 0$), all delay-tolerant requests are executed at edge hosts and the computation delay is $\frac{1}{\mu_{\text{e}} - \mu_{\text{s}}^t - \lambda_2^t}$. Otherwise ($\mu_{\text{e}} > \mu_{\text{s}}^t,\ \mu_{\text{c}}^t > 0$), delay-tolerant requests scheduled to edge hosts and cloud instances are proportional to the available computation capacities of each part, and the arrival rates are $\frac{\mu_{\text{e}} - \mu_{\text{s}}^t}{\mu_{\text{e}} - \mu_{\text{s}}^t + \mu_{\text{c}}^t} \lambda_2^t$ and $\frac{\mu_{\text{c}}^t}{\mu_{\text{e}} - \mu_{\text{s}}^t + \mu_{\text{c}}^t} \lambda_2^t$, respectively. The computation delay is the weighted sum of the delay at each part with the weights of $\frac{\mu_{\text{e}} - \mu_{\text{s}}^t}{\mu_{\text{e}} - \mu_{\text{s}}^t + \mu_{\text{c}}^t}$ and $\frac{\mu_{\text{c}}^t}{\mu_{\text{e}} - \mu_{\text{s}}^t + \mu_{\text{c}}^t}$. Based on above analysis, the computation delay of delay-tolerant requests can be summarized as follows:

$$D_{\text{comp}}^t = \begin{cases} \frac{2}{\mu_{\text{e}} - \mu_{\text{s}}^t + \mu_{\text{c}}^t - \lambda_2^t} + \frac{d\mu_{\text{c}}^t}{\mu_{\text{e}} - \mu_{\text{s}}^t + \mu_{\text{c}}^t}, & \mu_{\text{e}} > \mu_{\text{s}}^t, \mu_{\text{c}}^t > 0, \\ \frac{1}{\mu_{\text{c}}^t - \lambda_2^t} + d, & \mu_{\text{e}} = \mu_{\text{s}}^t, \mu_{\text{c}}^t > 0, \\ \frac{1}{\mu_{\text{e}} - \mu_{\text{s}}^t - \lambda_2^t}, & \mu_{\text{e}} > \mu_{\text{s}}^t, \mu_{\text{c}}^t = 0, \end{cases} \tag{4}$$

where $\lambda_2^t < \mu_{\text{e}} - \mu_{\text{s}}^t + \mu_{\text{c}}^t$ to ensure the stability of queues.

The overall system delay of delay-tolerant requests includes the saverage computation delay (when executed at mobile edge or further offloaded to the cloud) and the communication delay in the wireless access network. Thus, the system delay D^t can be represented as

$$D^t = D_{\text{comp}}^t + D_{\text{comm}}^t. \tag{5}$$

3.2 System Cost

In the resource provisioning problem of CAME, the system cost includes edge provisioning cost and cloud tenancy cost. On-demand instances, reserved instances and spot instances are the main cloud instances, with respect to different pricing schemes. Spot instances allow users to access spare computation resources for even greater discount compared to on-demand price. Nevertheless, applications scheduled to spot instances are required to have flexible start and end times in case of interruption. Furthermore, the pricing rate of spot instances varies with the relationship between supply and demand of cloud resources. To ensure the QoS of the CAME framework, spot instances are not considered in this work. Thus, the resource provisioning problem is divided into three categories: with on-demand instances, with reserved instances and hybrid strategy. In these cases, the system cost can be calculated as follows.

3.2.1 Edge Provisioning Cost

The dominating monetary cost of deploying a mobile edge goes to server purchasing [35]. In addition, the computation rates of mobile edge increases with the number of servers [36]. Therefore, the edge provisioning cost can be expressed as

$$C(\mu_{\text{e}}) = c_{\text{e}}\mu_{\text{e}}. \tag{6}$$

A typical server with 9.6 GHz costs \$3000, with an amortization of 3 years [35], [37].

In practical application, provisioning larger computation capacity at mobile edge yields much more expenses on parallelism. A convex pricing model is also employed in this work to approximate the edge provisioning cost as

$$C(\mu_{\text{e}}) = c_{\text{a}}(\mu_{\text{e}})^{\theta}, \tag{7}$$

where c_{a} and θ are constants, and $c_{\text{a}} > 0, \theta > 1$. Notice that the linear pricing model is a special case of convex pricing model with $\theta = 1$.

3.2.2 Cloud Tenancy Cost

On-demand instances are charged by per hour or even per second without long-term reservation. Denote by c_{p} and μ_{o}^t the pricing rate and serving rate of on-demand instances, respectively. Then, the cost of on-demand instances can be represented as

$$C^t(\mu_{\text{o}}^t) = c_{\text{p}}\mu_{\text{o}}^t. \tag{8}$$

Reserved instances provide users with a significant price discount (up to 75 percent [29]) compared with on-demand instances, while a one-year or three-year commitment is demanded. Therefore, the cost of reserved cloud instances is given by

$$C(\mu_{\text{r}}) = rc_{\text{p}}\mu_{\text{r}}, \tag{9}$$

where r represents the discount ratio of reserved instances over on-demand instances and μ_{r} denotes the serving rate of reserved instances.

3.3 Problem Formulation

In the resource provisioning problem of CAME, the computation capacity of edge hosts and cloud tenancy strategy should be jointly optimized to achieve minimum system cost, formulated as follows:

$$\begin{aligned}(\mathbf{P1}): \min\ & S(\mu_{\mathrm{e}}, \mu_{\mathrm{c}}^t) = C(\mu_{\mathrm{e}}) + \frac{1}{T}\sum_{t=1}^{T} C^t(\mu_{\mathrm{c}}^t) \\ s.t.\ \ & \mu_{\mathrm{e}} - \mu_{\mathrm{s}}^t \geq 0 && t \in \{1,2,\ldots,T\} \\ & \mu_{\mathrm{c}}^t \geq 0 && t \in \{1,2,\ldots,T\} \\ \mathbf{C1}: & \mu_{\mathrm{e}} - \mu_{\mathrm{s}}^t + \mu_{\mathrm{c}}^t > \lambda_2^t && t \in \{1,2,\ldots,T\} \\ \mathbf{C2}: & D^t(\mu_{\mathrm{e}}, \mu_{\mathrm{c}}^t) \leq D_2 && t \in \{1,2,\ldots,T\},\end{aligned} \tag{10}$$

where D_2 represents the delay requirement of delay-tolerant requests. In this problem, the objective is to minimize the system cost. As the usage of cloud resources varies with different time intervals, the long-term average cloud tenancy cost is $\frac{1}{T}\sum_{t=1}^{T} C^t(\mu_{\mathrm{c}}^t)$. **C1** constraints serving rates to ensure the stability of system, and **C2** guarantees the system delay to meet the QoS requirements.

As the computation capacity of edge hosts μ_{e} is unchanged through all time intervals, **(P1)** can be solved as follows:

Step1. Given the computation capacity of edge hosts μ_{e}, obtain the optimal cloud tenancy strategy μ_{c}^t ($t \in \{1, 2, \ldots, T\}$) satisfying

$$\begin{aligned}(\mathbf{P2}): \min\ & C^t(\mu_{\mathrm{c}}^t) \\ s.t.\ \ & \mu_{\mathrm{c}}^t \geq 0 \\ \mathbf{C3}: \ & D^t(\mu_{\mathrm{c}}^t) \leq D_2 \\ & \mu_{\mathrm{c}}^t > \lambda_2^t - (\mu_{\mathrm{e}} - \mu_{\mathrm{s}}^t).\end{aligned} \tag{11}$$

Denote by $\mu_{\mathrm{c}}^{t*}(\mu_{\mathrm{e}})$ the optimal solution of *(P2)*, then the cloud tenancy cost can be represented as $C^t(\mu_t^{\mathrm{c}*}(\mu_{\mathrm{e}}))$.

Step2. Determine the optimal computation capacity of edge hosts μ_{e} satisfying

$$\begin{aligned}(\mathbf{P3}): \min\ & \frac{1}{T}\sum_{t=1}^{T} C^t(\mu_t^{\mathrm{c}*}(\mu_{\mathrm{e}})) + C(\mu_{\mathrm{e}}) \\ s.t.\ \ & \mu_{\mathrm{e}} - \mu_{\mathrm{s}}^t \geq 0 \quad t \in \{1,2,\ldots,T\}.\end{aligned} \tag{12}$$

When μ_{e} is given, the optimal cloud tenancy strategy can be represented as a function of μ_{e} by solving *P2*. Then the optimal computation capacity of edge hosts μ_{e} can be further determined by solving the piecewise convex optimization problem (*P3*). Based on above analysis, the ORP algorithms are designed to address the three categories of the resource provisioning problem.

4 Algorithm Design

This part provides algorithm design for the resource provisioning problem in CAME. Three optimal resource provisioning algorithms, i.e., ORP-OD, ORP-R and ORP-HS, are designed to deal with the optimal resource problem with on-demand, reserved instances and hybrid strategy, respectively. The details and analysis are as follows.

4.1 Resource Provisioning with On-Demand Instances

In the category of on-demand instances, cloud resources are tenanted on demand without long-term reservation [29]. Thus, in the problem formulation of (*P1*),

$$\begin{aligned}&\mu_{\mathrm{c}}^t = \mu_{\mathrm{o}}^t, \\ &C^t(\mu_{\mathrm{c}}^t) = C^t(\mu_{\mathrm{o}}^t).\end{aligned} \tag{13}$$

The relationship between the optimal usage of on-demand instances and computation capacity of edge hosts is first explored. Then the computation capacity of edge hosts is further optimized based on the above results.

4.1.1 Optimize Usage of On-Demand Instances

The optimal usage of on-demand instances can be obtained by solving *(P2)* when the computation capacity of edge hosts is given. As the cloud tenancy cost $C^t(\mu_{\mathrm{o}}^t)$ increases with the serving rate of on-demand instances μ_{o}^t, solving *(P2)* is equivalent to searching for the minimum μ_{o}^t within the constraints.

To ensure *C3*, it should be satisfied that

$$\alpha \cdot (\mu_{\mathrm{o}}^t)^2 + \beta(\mu_{\mathrm{e}})\mu_{\mathrm{o}}^t + \gamma(\mu_{\mathrm{e}}) \geq 0, \tag{14}$$

where α, $\beta(\mu_{\mathrm{e}})$ and $\gamma(\mu_{\mathrm{e}})$ are given by

$$\begin{aligned}&\alpha = D(\lambda_2^t) - d, \\ &\beta(\mu_{\mathrm{e}}) = (2D(\lambda_2^t) - d)(\mu_{\mathrm{e}} - \mu_{\mathrm{s}}^t) + \lambda_2^t d - D(\lambda_2^t)\lambda_2^t - 2, \\ &\gamma(\mu_{\mathrm{e}}) = D(\lambda_2^t)(\mu_{\mathrm{e}} - \mu_{\mathrm{s}}^t)^2 - D(\lambda_2^t)\lambda_2^t(\mu_{\mathrm{e}} - \mu_{\mathrm{s}}^t) \\ &\qquad\quad - 2(\mu_{\mathrm{e}} - \mu_{\mathrm{s}}^t),\end{aligned} \tag{15}$$

where $D(\lambda_2^t)$ represents the computation delay requirement of delay-tolerant applications, i.e., $D(\lambda_2^t) = D_2 - D_{\mathrm{comm}}^t$. The round trip delay $d < D(\lambda_2^t)$ to guarantee the QoS of delay-tolerant requests.

Define

$$\Delta(\mu_{\mathrm{e}}) = [\beta(\mu_{\mathrm{e}})]^2 - 4\alpha\gamma(\mu_{\mathrm{e}}). \tag{16}$$

It can be proved that for any μ_{e}, there exists $\Delta(\mu_{\mathrm{e}}) > 0$. The details are as follows.

Proof.

$$\begin{aligned}\Delta(\mu_{\mathrm{e}}) &= [\beta(\mu_{\mathrm{e}})]^2 - 4\alpha\gamma(\mu_{\mathrm{e}}) \\ &= d^2(\mu_{\mathrm{e}} - \mu_{\mathrm{s}}^t)^2 + \delta(\mu_{\mathrm{e}} - \mu_{\mathrm{s}}^t) + \epsilon,\end{aligned} \tag{17}$$

where

$$\begin{aligned}&\delta = 2d\lambda_2^t D(\lambda_2^t) - 2\lambda_2^t d^2 - 4d \\ &\epsilon = [D(\lambda_2^t)]^2(\lambda_2^t)^2 + d^2(\lambda_2^t)^2 - 2dD(\lambda_2^t)(\lambda_2^t)^2 \\ &\qquad + 4D(\lambda_2^t)\lambda_2^t - 4d\lambda_2^t + 4.\end{aligned} \tag{18}$$

Since $d^2 > 0$ and $\delta^2 - 4d^2\epsilon = 32d^2\lambda_2^t(d - D(\lambda_2^t)) < 0$, there exists

$$\Delta(\mu_{\mathrm{e}}) = d^2(\mu_{\mathrm{e}} - \mu_{\mathrm{s}}^t)^2 + \delta(\mu_{\mathrm{e}} - \mu_{\mathrm{s}}^t) + \epsilon > 0, \tag{19}$$

for any μ_{e}. □

As $a > 0$ and $\Delta(\mu_e) > 0$, Eq. (14) can be rewritten as $\mu_o^t \le f_d^t(\mu_e)$ or $\mu_o^t \ge f_u^t(\mu_e)$, where

$$f_d^t(\mu_e) = \frac{-\beta(\mu_e) - \sqrt{\Delta(\mu_e)}}{2\alpha}$$
$$f_u^t(\mu_e) = \frac{-\beta(\mu_e) + \sqrt{\Delta(\mu_e)}}{2\alpha}. \quad (20)$$

Theorem 1. *When the computation capacity of edge hosts μ_e is given, the optimal usage of on-demand instances is given by*

$$\mu_o^{t*}(\mu_e) = \begin{cases} 0 & \mu_e \ge \mu_s^t + \lambda_2^t + \frac{1}{D(\lambda_2^t)} \\ f_u^t(\mu_e) & \mu_s^t < \mu_e < \mu_s^t + \lambda_2^t + \frac{1}{D(\lambda_2^t)} \\ \lambda_2^t + \frac{1}{D(\lambda_2^t) - d} & \mu_e = \mu_s^t. \end{cases} \quad (21)$$

Proof. Define $F(\mu_o^t)$ as

$$\begin{aligned} F(\mu_o^t) = {} & (D(\lambda_2^t) - d)(\mu_o^t)^2 \\ & + (2D(\lambda_2^t)\mu_e + \lambda_2^t d - D(\lambda_2^t)\lambda_2^t - \mu_e d - 2)\mu_o^t \\ & + (D(\lambda_2^t)(\mu_e)^2 - D\lambda_2^t \mu_e - 2\mu_e). \end{aligned} \quad (22)$$

Then,

$$\begin{aligned} & F(\lambda_2^t - (\mu_e - \mu_s^t)) \\ & = (D(\lambda_2^t) - d)(\lambda_2^t - (\mu_e - \mu_s^t))^2 \\ & \quad + (2D(\lambda_2^t)(\mu_e - \mu_s^t) + \lambda_2^t d - D(\lambda_2^t)\lambda_2^t - (\mu_e - \mu_s^t)d - 2) \\ & \quad * (\lambda_2^t - (\mu_e - \mu_s^t)) \\ & \quad + D(\lambda_2^t)(\mu_e - \mu_s^t)^2 - D(\lambda_2^t)\lambda_2^t(\mu_e - \mu_s^t) - 2(\mu_e - \mu_s^t) \\ & = -2\lambda_2^t < 0 \end{aligned} \quad (23)$$

Since there is $F(\mu_o^t) \ge 0$ when $\mu_o^t \le f_d^t(\mu_e)$ and $\mu_o^t \ge f_u^t(\mu_e)$, and $F(\lambda_2^t - (\mu_e - \mu_s^t)) < 0$, the following holds:

$$f_d^t(\mu_e) < \lambda_2^t - (\mu_e - \mu_s^t) < f_u^t(\mu_e). \quad (24)$$

To ensure the constraints in *(P2)*, it should satisfy that

$$\begin{aligned} & \mu_o^t \ge 0 \\ & \mu_o^t \ge \lambda_2^t - (\mu_e - \mu_s^t) \\ & \mu_o^t \ge f_u^t(\mu_e) \text{ or } \mu_o^t \le f_d^t(\mu_e). \end{aligned} \quad (25)$$

Combining Eqs. (24) and (25), when given μ_e, the optimal usage of on-demand instances can be obtained as presented in Eq. (21). □

Therefore, in the resource provisioning with on-demand instances, the optimal usage of on-demand instances can be represented by a function of μ_e, as shown in Eq. (21).

4.1.2 Optimize Computation Capacity of Edge Hosts

According to Theorem 1, we can conclude that the objective function of (*P3*) is a piecewise function over μ_e, and the boundaries are $(\mu_s^t + \lambda_2^t + \frac{1}{D(\lambda_2^t)})$, $t \in \{1, 2, \ldots, T\}$. Sort the boundaries and let

$$b_j = \mu_s^{t(j)} + \lambda_2^{t(j)} + \frac{1}{D(\lambda_2^{t(j)})}, \quad (26)$$

where $t(j)$ represents the time interval with the jth smallest $(\mu_s^t + \lambda_2^t + \frac{1}{D(\lambda_2^t)})$. Define $b_0 = \max\{\mu_s^t, j \in \{1, 2, \ldots, T\}\}$ and $b_{T+1} = +\infty$. Then there is the following conclusion.

Theorem 2. *The long-term average system cost $\frac{1}{T}\sum_{t=1}^{T} C^t(\mu_o^{t*}(\mu_e)) + C(\mu_e)$ is convex over μ_e in each domain bounded by b_j, $j \in \{0, 1, 2, \ldots, T, T+1\}$.*

Proof. As indicated in Eq. (21), the minimum cloud tenancy cost $C^t(\mu_c^{t*}(\mu_e))$ equals $c_p f_u^t(\mu_e)$ when $\mu_e \in (\mu_s^t, \mu_s^t + \lambda_2^t + \frac{1}{D(\lambda_2^t)})$. Thus, there exists

$$C^t(\mu_c^{t*}(\mu_e)) = \frac{c_p * (-\beta(\mu_e) + \sqrt{\Delta(\mu_e)})}{2\alpha}. \quad (27)$$

Based on the results of Eqs. (15) and (17), the second derivative of $C^t(\mu_c^{t*}(\mu_e))$ over μ_e can be given by

$$[C^t(\mu_c^{t*}(\mu_e))]''_{\mu_e} = \frac{2c_p\lambda_2^t d^2}{(\sqrt{\beta(\mu_e)^2 - 4\alpha\gamma(\mu_e)})^3}. \quad (28)$$

For each $t \in \{1, 2, \ldots, T\}$, there exists $[C^t(\mu_c^{t*}(\mu_e))]''_{\mu_e} > 0$ when $\mu_e \in (\mu_s^t, \mu_s^t + \lambda_2^t + \frac{1}{D(\lambda_2^t)})$, thus $C^t(\mu_c^{t*}(\mu_e))$ is a convex function over μ_e. In addition, as revealed in Eq. (7), $C(\mu_e)$ is a convex function over μ_e, $(\mu_e \ge 0)$. Thus the long-term system cost $\frac{1}{T}\sum_{t=1}^{T} C^t(\mu_{cloud}^{t*}(\mu_e)) + C(\mu_e)$ is the sum of several convex functions in each segment bounded by b_j, $j \in \{0, 1, 2, \ldots, T, T+1\}$. Thus, Theorem 2 can be proved. □

It can be inferred from Theorem 2 that *(P3)* is a piecewise convex optimization problem over μ_e. Thus, to obtain the optimal computation capacity of edge hosts, we just need to solve convex optimization problems in each domain of μ_e bounded by b_j ($j \in \{0, 1, 2, \ldots, T, T+1\}$) [38], and the optimal usage of on-demand instances can be further determined based on Theorem 1. Above all, the resource provisioning with on-demand instances can be solved by ORP-OD, as shown in Algorithm 1.

As the gradient of system cost over μ_e can be expressed in a closed form (as shown in step 4), the optimal solutions of the convex optimization problems can be obtained by bisection method, as illustrated in step 5 to step 13. Then, *(P3)* can be solved by selecting from the solutions of these convex optimization problems. Since the sorting of boundaries induces the main calculation, the computation complexity of this algorithm is $O(T \log T)$.

4.2 Resource Provisioning with Reserved Instances

In the resource provisioning with reserved instances, the serving rate of cloud instances remains unchanged through the T time intervals, due to the long-term commitment of reserved instances [29]. Denote by μ_r the serving rate of reserved instances, and we have

$$\begin{aligned} & \mu_c^t = \mu_r, \\ & C^t(\mu_c^t) = C(\mu_r), \end{aligned} \quad (29)$$

in the problem formulation *(P1)*.

As the serving rate of reserved instances remains unchanged, the provisioned computation capacity of both

edge hosts and cloud instances should ensure the QoS of mobile requests in the long run. Thus, when the computation capacity of edge hosts μ_{e} is provided, the optimal usage of reserved instances can be given by

$$\mu_{\mathrm{r}}^{*}(\mu_{\mathrm{e}}) = \max_{t\in\{1,2,\ldots,T\}}\{\mu_{\mathrm{o}}^{t*}(\mu_{\mathrm{e}})\}, \tag{30}$$

where $\mu_{\mathrm{o}}^{t*}(\mu_{\mathrm{e}})$ is as shown in Eq. (21). Then *(P3)* is transformed into

$$\begin{aligned}(\mathbf{P4}) : \min\{&C(\mu_{\mathrm{r}}^{*}(\mu_{\mathrm{e}})) + C(\mu_{\mathrm{e}})\}\\ s.t.\ \ &\mu_{\mathrm{e}} - \mu_{\mathrm{s}}^{t} \geq 0 \quad t \in \{1,2,\ldots,T\}.\end{aligned} \tag{31}$$

Remark. According to the proof of Theorem 2, $\mu_{\mathrm{o}}^{t*}(\mu_{\mathrm{e}})$ is convex in each domain of μ_{e} bounded by (b_{j-1}, b_j), $(j \in \{1,2...,T,T+1\})$. Then $\max_{t\in\{1,2,\ldots,T\}}\{\mu_{\mathrm{o}}^{t*}(\mu_{\mathrm{e}})\}$ is convex in each domain of μ_{e}. In addition, $C(\mu_{\mathrm{e}})$ is a convex function over μ_{e}. Thus, (P4) is a piecewise convex optimization problem and can be solved as summarized in Algorithm 2.

Algorithm 1. ORP-R: Optimal Resource Provisioning with On-Demand Instances

Input:
The arrival rates of delay-sensitive and delay-tolerant requests in each time interval $[\lambda_1^t, \lambda_2^t]$, $t \in \{1,2,\ldots,T\}$.
Output:
The optimal computation capacity of edge hosts μ_{e}^{*};
The optimal usage of on-demand instances in each time interval μ_{o}^{t*}, $t \in \{1,2,\ldots,T\}$.

1: Define $\mu_{\mathrm{e\_0}} = \max_{t\in\{1,2,\ldots,T\}}\{\mu_t\}$, representing the original solution of μ_{e}, and $\mu_{\mathrm{e\_j}}$ as the optimal solution in the jth segment.
2: Sort $(\mu_{\mathrm{s}}^{t} + \lambda_2^t + \frac{1}{D(\lambda_2^t)})$, $t \in \{1,2,\ldots,T\}$ and denote as b_j, $(j \in \{0,1,2,\ldots,T,T+1\})$.
3: **for** each segment of μ_{e} bounded by (b_{j-1}, b_j), where $j \in \{1,2,\ldots,T\}$, $b_{j-1}, b_j \geq \mu_{\mathrm{e\_0}}$ **do**
4: Obtain the gradient of system cost by $[S(\mu_{\mathrm{e}})]' = \frac{c_p}{T}\sum_{i=j}^{T}[f_{\mathrm{u}t(i)}^{\mathrm{u}}(\mu_{\mathrm{e}})]' + c_0\theta(\mu_{\mathrm{e}})^{\theta-1}$.
5: **if** $[S(b_{j-1})]' * [S(b_j)]' > 0$ **then**
6: **if** $[S(b_{j-1})]' > 0$ **then**
7: $\mu_{\mathrm{e\_j}} = b_{j-1} + \varepsilon$.
8: **else**
9: $\mu_{\mathrm{e\_j}} = b_j - \varepsilon$.
10: **end if**
11: **else**
12: Search the extreme point $\mu_{\mathrm{e\_ex}}$ ($[S(\mu_{\mathrm{e\_ex}})]' = 0$) with bisection search method. $\mu_{\mathrm{e\_j}} = \mu_{\mathrm{e\_ex}}$.
13: **end if**
14: **end for**
15: $\mu_{\mathrm{e}\_\{T+1\}} = b_T + \varepsilon$
16: $\mu_{\mathrm{e}}^{*} = \arg\min_{\mu_{\mathrm{e}}\in\{\mu_{\mathrm{e}\_1},\ldots,\mu_{\mathrm{e}\_\{T+1\}}\}}\{S(\mu_{\mathrm{e}}, \mu_t^{\mathrm{o}*}(\mu_{\mathrm{e}}))\}$.
17: Compute μ_{o}^{t*} according to Eq. (21).

As shown in Algorithm 2, in each domain of μ_{e} bounded by (b_{j-1}, b_j), $j \in \{1,2,\ldots,T\}$, the optimal solution of μ_{e} can be obtained by Gradient Descending from Step 3 to Step 15, as *(P4)* is a convex optimization problem over μ_{e} in each domain. It is intuitive that the computation complexity from Step 3 to Step 7 is $O(T)$. From Step 8 to Step 14, gradient descending takes $O(\log(1/\sigma))$ iterations, with each iteration yielding the complexity of $O(T)$ to calculate the gradient ϱ. Thus the computation complexity from Step 8 to Step 14 is $O(T\log(1/\sigma))$, and Algorithm 2 yields the computation complexity of $O(T^2\log(1/\sigma))$.

Algorithm 2. ORP-R: Optimal Resource Provisioning with Reserved Instances

Input:
The arrival rates of delay-sensitive and delay-tolerant requests in each time interval $[\lambda_1^t, \lambda_2^t]$, $t \in \{1,2,\ldots,T\}$.
Output:
The optimal computation capacity of edge hosts μ_{e}^{*};
The optimal usage of reserved instances μ_{r}.

1: Sort $(\mu_{\mathrm{s}}^{t} + \lambda_2^t + \frac{1}{D(\lambda_2^t)})$, $t \in \{1,2,\ldots,T\}$ and denote as b_j, $(j \in \{0,1,2,\ldots,T,T+1\})$.
2: **for** each domain of μ_{e} bounded by (b_{j-1}, b_j), where $j \in \{1,2,\ldots,T\}$, $b_{j-1}, b_j \geq \mu_{\mathrm{e\_0}}$ **do**
3: Set the original value $\mu_{\mathrm{e}} = b_{j-1}$.
4: Find the time interval i that has the largest $f_{\mathrm{u}}^{t}(\mu_{\mathrm{e}})$, i.e., $i = \arg\max_{t\in\{1,2,\ldots,T\}} f_{\mathrm{u}}^{t}(\mu_{\mathrm{e}})$.
5: $\varrho = r * c_p * [f_{\mathrm{u}}^{i}(\mu_{\mathrm{e}})]' + c_0 * \theta * (\mu_{\mathrm{e}})^{\theta-1}$, representing the local gradient.
6: Let the learning rate $\tau = 1$.
7: Let the learning precision $\sigma = 10^{-5}$.
8: **while** $\mu_{\mathrm{e}} < b_j$ && $\mu_{\mathrm{e}} > b_{j-1}$ && $\phi > \sigma$ **do**
9: $\mu_{\mathrm{e}}^{\mathrm{old}} = \mu_{\mathrm{e}}$.
10: $\mu_{\mathrm{e}} = \mu_{\mathrm{e}} - \tau * \varrho$.
11: $\phi = (\mu_{\mathrm{e}} - \mu_{\mathrm{e}}^{\mathrm{old}})^2$.
12: $i = \arg\max_{t\in\{1,2,\ldots,T\}} f_{\mathrm{u}}^{t}(\mu_{\mathrm{e}})$.
13: $\varrho = r * c_p * [f_{\mathrm{u}}^{i}(\mu_{\mathrm{e}})]' + c_0 * \theta * (\mu_{\mathrm{e}})^{\theta-1}$.
14: **end while**
15: $\mu_{\mathrm{e\_}j} = \mu_{\mathrm{e}}^{\mathrm{old}}$.
16: **end for**
17: $\mu_{\mathrm{e}\_\{T+1\}} = b_T + \varepsilon$
18: $\mu_{\mathrm{e}}^{*} = \arg\min_{\mu_{\mathrm{e}}\in\{\mu_{\mathrm{e}\_1},\ldots,\mu_{\mathrm{e}\_\{T+1\}}\}}\{S(\mu_{\mathrm{e}}, \mu_{\mathrm{r}}^{*}(\mu_{\mathrm{e}}))\}$.
19: Compute μ_{r}^{*} according to Eq. (30).

4.3 Resource Provisioning with Hybrid Strategy

In the resource provisioning with hybrid strategy, a combination of on-demand and reserved instances are leased to complement the computation capacity of edge hosts. Thus, in the problem formulation of resource provisioning with hybrid strategy, *(P1)* is transformed into

$$\begin{aligned}&\mu_{\mathrm{c}}^{t} = \mu_{\mathrm{o}}^{t} + \mu_{\mathrm{r}},\\ &C^{t}(\mu_{\mathrm{c}}^{t}) = C^{t}(\mu_{\mathrm{o}}^{t}) + C(\mu_{\mathrm{r}}),\end{aligned} \tag{32}$$

where $\mu_{\mathrm{o}}^{t} \geq 0$, $\mu_{\mathrm{r}} \geq 0$.

According to the conclusion of Theorem 1, to ensure the QoS of mobile requests, when μ_{e}, the tenanted cloud instances μ_{c}^{t} should satisfy

$$\mu_{\mathrm{c}}^{t} \geq \mu_{t}^{\mathrm{o}*}(\mu_{\mathrm{e}}), \tag{33}$$

where $\mu_{t}^{\mathrm{o}*}(\mu_{\mathrm{e}})$ is presented as in Eq. (21). In addition, as $\mu_{\mathrm{c}}^{t} = \mu_{\mathrm{r}} + \mu_{\mathrm{o}}^{t} \geq \mu_{\mathrm{r}}$, we have

$$\mu_{\mathrm{c}}^{t} \geq \max\{\mu_{\mathrm{o}}^{t*}(\mu_{\mathrm{e}}), \mu_{\mathrm{r}}\}. \tag{34}$$

Therefore, in the resource provisioning problem with hybrid strategy, the minimum provisioned on-demand instances can be summarized as

$$\mu_{\mathrm{o}}^{t**} = \begin{cases} 0 & \mu_{\mathrm{e}} \geq \mu_{\mathrm{s}}^t + \lambda_2^t + \frac{1}{D(\lambda_2^t)} \\ \max\{f_{\mathrm{u}}^t, \mu_{\mathrm{r}}\} - \mu_{\mathrm{r}}, & \mu_{\mathrm{s}}^t < \mu_{\mathrm{e}} < \mu_{\mathrm{s}}^t + \lambda_2^t + \frac{1}{D(\lambda_2^t)} \\ \max\{\lambda_2^t + \frac{1}{D(\lambda_2^t)-d}, \mu_{\mathrm{r}}\} - \mu_{\mathrm{r}} & \mu_{\mathrm{e}} = \mu_{\mathrm{s}}^t. \end{cases} \tag{35}$$

It has been proved that f_{u}^t is a convex function over μ_{e} (see Eq. (28)), thus, $\max\{f_{\mathrm{u}}^t, \mu_{\mathrm{r}}\} - \mu_{\mathrm{r}}$ is convex over $\langle \mu_{\mathrm{e}}, \mu_{\mathrm{r}} \rangle$, and it is not difficult to conclude that $\mu_{\mathrm{o}}^{t**}(\mu_{\mathrm{e}}, \mu_{\mathrm{r}})$ is piecewise convex over $\langle \mu_{\mathrm{e}}, \mu_{\mathrm{r}} \rangle$. By substituting (32), (35) into *(P1)*, the resource provisioning problem with hybrid strategy can be reduced into

$$\begin{aligned} (\mathbf{P5}): \min\{C(\mu_{\mathrm{e}}) + C(\mu_{\mathrm{r}}) + \frac{1}{T}\sum_{t=1}^{T} C^t(\mu_{\mathrm{o}}^{t**}(\mu_{\mathrm{e}}, \mu_{\mathrm{r}}))\} \\ s.t. \quad \mu_{\mathrm{e}} - \mu_{\mathrm{s}}^t \geq 0 \qquad t \in \{1, 2, \ldots, T\} \\ \mu_{\mathrm{r}} \geq 0 \qquad t \in \{1, 2, \ldots, T\} \\ \mu_{\mathrm{e}} - \mu_{\mathrm{s}}^t + \mu_{\mathrm{r}} + \mu_{\mathrm{o}}^{t**}(\mu_{\mathrm{e}}, \mu_{\mathrm{r}}) > \lambda_2^t \quad t \in \{1, 2, \ldots, T\}, \end{aligned} \tag{36}$$

Since $C(\mu_{\mathrm{e}})$ is convex (or linear) with μ_{e}, and $C(\mu_{\mathrm{r}})$ is linear with μ_{r}, the objective function of *(P5)* is piecewise convex over $\langle \mu_{\mathrm{e}}, \mu_{\mathrm{r}} \rangle$. Therefore, *(P5)* is a piecewise convex optimization problem over $\langle \mu_{\mathrm{e}}, \mu_{\mathrm{r}} \rangle$, and can be solved by Gradient Descending, similar to Algorithm 2, and we call this algorithm Optimal Resource Provisioning with Hybrid Strategy (ORP-HS).

As *(P5)* is a piecewise convex optimization problem over a bivector (i.e., $\langle \mu_{\mathrm{e}}, \mu_{\mathrm{r}} \rangle$), solving the convex optimization problems in each segment can induce $O(\log^2(1/\sigma))$ iterations. Thus the total computation complexity of ORP-HS is $O(T^2 \log^2(1/\sigma))$. Denote by f the calculation that can be completed in each time interval. It is intuitive that to ensure the feasibility of ORP-HS, the calculation time should not exceed one time interval, i.e.,

$$\frac{\varpi T^2 \log^2(1/\sigma)}{f} < 1, \tag{37}$$

where ϖ is a constant that depends on the algorithm setting. Thus the precision σ should satisfy that

$$\sigma > \mathrm{e}^{-\sqrt{\frac{f}{\varpi T^2}}}. \tag{38}$$

It is shown in Eq. (38) that to ensure the feasibility of the algorithm, the precision σ is lower bounded. To achieve higher precision (smaller value of σ) in ORP-HS, one can either increase the computing capability during each time interval, or reduce the number of time intervals in a cycle.

5 Simulation Results

This section evaluates the proposed ORP algorithms by conducting extensive simulations. Simulation results based on two widely-used traffic models are first presented to validate the benefits of the ORP algorithms in term of cost-efficiency and flexibility. Then, to evaluate the ORP algorithms in practical, trace-driven experiments based on Google cluster tracelogs are further performed.

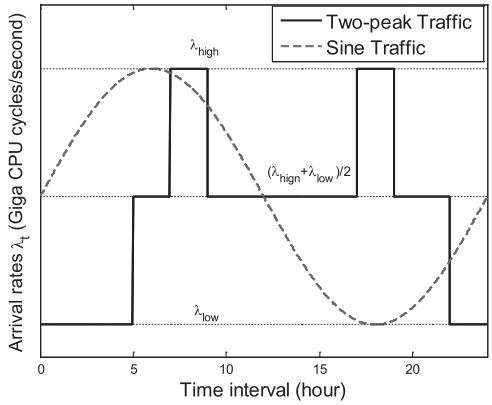


Fig. 3. Traffic patterns of mobile requests [39].

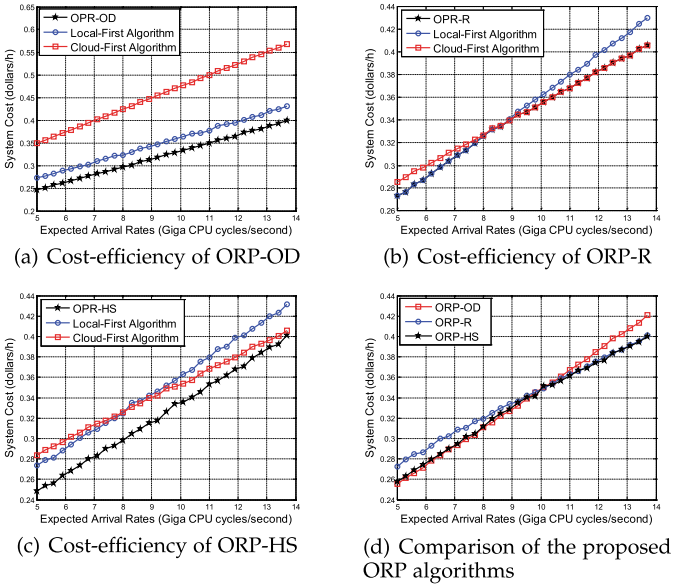
In these simulations, there is a base station with both computation resources (edge hosts) and communication resources (the wireless access network). Edge hosts are deployed within or in close proximity to the base station. The average transmission rate of the wireless access network is 100 Mbps. The cloud instance M4.large of Amazon EC2 [29] is leased to augment the system computation capacity, and pricing rate (when tenanted on demand) is 0.0208 dollar/GHz per hour. The reserved instances are charged with a discount of 50 percent over on-demand instances. The average round-trip delay from edge hosts to the cloud is set to 50 milliseconds [8], [9]. The face recognition application in [1] is considered as the representative of delay-tolerant applications, with the computation requests of 1000 Mege CPU cycles and transmitted block of 420 KB. To guarantee the QoS of the face recognition application, the system delay when executing delay-tolerant requests should not exceed 400 milliseconds [2]. The online game First Person Avatar in [9] is the representative of delay-sensitive applications, of which the processing delay should not exceed 100 milliseconds.

The local-first and the cloud-first algorithms are selected as the benchmark algorithms in these simulations. In the local-first algorithm, both the delay-sensitive and delay-tolerant requests are served at mobile edge. Sufficient edge hosts should be deployed to ensure the QoS of these diversified requests through all time intervals. In the cloud-first algorithm, edge hosts only provides sufficient computation resources for delay-sensitive applications, and all delay-tolerant requests are outsourced to the cloud.

5.1 Evaluation Based on Two Traffic Models

Simulations based on different traffic models are conducted to evaluated the ORP algorithms. Two widely-used cellular traffic models in Fig. 3 are employed in these simulations, as the daily traffic pattern of mobile requests at mobile edge is similar to the communication traffic in cellular network. The Two-peak Traffic Model is constantly adopted to characterize the dynamics of daily traffic at public places (e.g., bus stations), and the peaks represent the requests at rush hours. The Sine Traffic Model is frequently used to validate the effectiveness of network deployment algorithms [39]. Based on these traffic models, the performance of the ORP algorithms is evaluated, in term of both cost-efficiency and system flexibility.

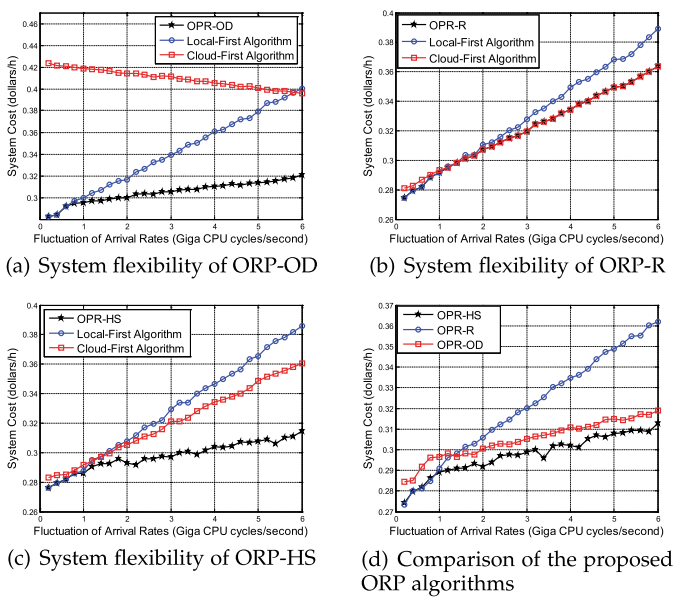
In the resource provisioning problem, the composition of mobile requests can significantly influence the results of the



(a) Cost-efficiency of ORP-OD (b) Cost-efficiency of ORP-R

(c) Cost-efficiency of ORP-HS (d) Comparison of the proposed ORP algorithms

Fig. 4. System cost over expectation of traffic pattern 1, with delay-tolerant dominant requests.



(a) System flexibility of ORP-OD (b) System flexibility of ORP-R

(c) System flexibility of ORP-HS (d) Comparison of the proposed ORP algorithms

Fig. 5. System cost over fluctuation of traffic pattern 1, with delay-tolerant dominant requests.

ORP algorithms. In the resource provisioning problem with delay-sensitive dominant requests, sufficient edge hosts should be deployed to guarantee the QoS of delay-sensitive requests. Yet for the delay-tolerant dominant resource provisioning problem, more scalable cloud instances can be leased to execute the outsourced delay-tolerant requests. The ORP algorithms with delay-tolerant dominant requests is first evaluated, then the results with delay-sensitive dominant requests are further analyzed.

5.1.1 Delay-Tolerant Dominant Resource Provisioning

The delay-tolerant dominant resource provisioning problem is investigated, in which the arrival rate of delay-tolerant requests is much larger than that of delay-sensitive requests. The performance of the ORP algorithms is compared to the two benchmark algorithms in terms of cost-efficiency and flexibility, with the results illustrated in Figs. 4 and 5, respectively.

Fig. 4 shows system cost of different algorithms with increasing mobile computation load, i.e., expectation of arrival rates. The variance of mobile request arrival rates is 2 Giga CPU cycles/second. In this simulation, the Two-peak Traffic Model is adopted. Compared to the benchmark algorithms, the ORP algorithms can always yield the minimum system cost, as shown in Fig. 4. Fig. 4a illustrates the system cost when provisioning resources with on-demand instances. Due to the relatively low dynamics of mobile requests and high pricing rate of on-demand instances, the system cost of cloud-first algorithm is the highest among the three algorithms. Fig. 4b shows the results in resource provisioning with reserved instances. The system computation resources of the three algorithms remain unchanged through all time intervals, since the computation capacity of edge hosts and reserved instances are fixed. As the pricing rate of edge hosts increases with the serving rate, thus the system cost of local-first algorithm grows faster with increasing mobile computation load than cloud-first algorithm. In the OPR-R algorithm, the system cost can always be minimized by provisioning the computation resources (edge hosts or reserved instances) with lower pricing rate.

In Fig. 4c, with increasing arrival rates of mobile requests, the system cost of ORP-HS approximates that of the cloud-first algorithm. As when the arrival rates of requests increase, the pricing rate of edge hosts becomes larger than reserved instances. To reduce the system cost, more reserved instances are tenanted in the ORP-HS algorithm rather than increasing the computation capacity of edge hosts. The results of the proposed ORP algorithms are compared, as shown in Fig. 4d. As the expected arrival rates increase, mobile requests become less fluctuated. Nevertheless, the pricing rate of the reserved instances is much lower than on-demand instances. Thus, the system cost of ORP-OD increases faster than the ORP-R algorithm. In the ORP-HS algorithm, the system cost can be significantly reduced by exploiting the advantages of both on-demand instances (scalability) and reserved instances (economy).

Flexibility of the ORP algorithms is evaluated by comparing the system cost with changing fluctuations of mobile requests. As illustrated in Figs. 5a, 5b, and 5c, the proposed ORP algorithms can always yield the minimum system cost compared to the benchmark algorithms. In the resource provisioning with on-demand instances (Fig. 5a), the system cost of ORP-OD and the cloud-first algorithm increase much slower (or even decrease) with higher dynamics of requests than that of local-first algorithm. For the local-first algorithm, overprovisioning of edge hosts is incurred to guarantee the QoS at rushing hours, while for the cloud-first and ORP-OD algorithms, cloud instances can be leased on demand to deal with fluctuated requests, resulting in higher resource utilization and lower system cost. In the resource provisioning with reserved instances (Fig. 5b), both the mobile edge and cloud have fixed computation capacities. Substantial computation resources should be provisioned both at mobile edge and cloud to guarantee the QoS in the long run. Since the reserved instances have a fixed pricing rate while the pricing rate of edge hosts varies with the serving rate, the ORP-R

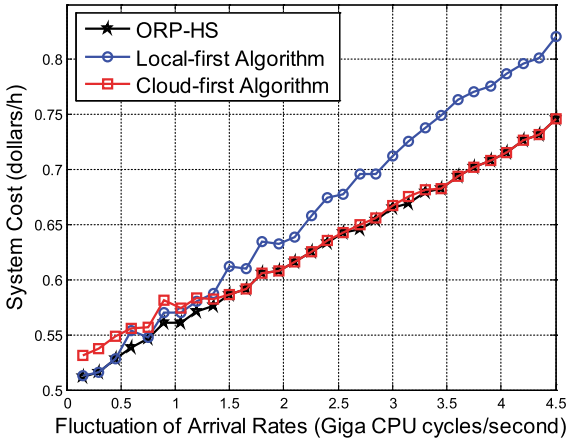


Fig. 6. System cost over fluctuation of traffic pattern 2 with delay-sensitive dominant requests.

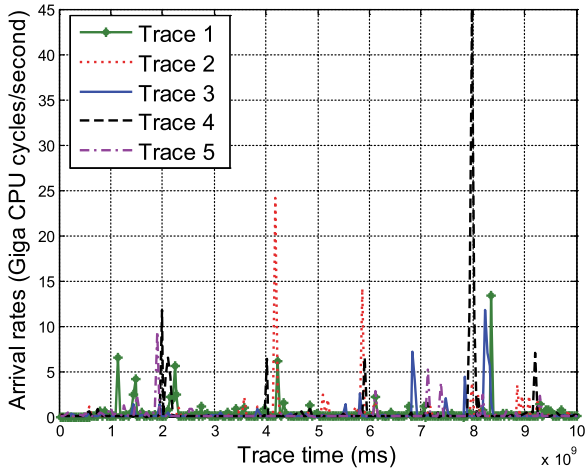


Fig. 7. Arrival rates of mobile requests based on Google Cluster Tracelogs.

algorithm always provisions the computation resources with lower pricing rate between edge hosts and cloud instances. Therefore, ORP-R always yields the lower system cost of the two benchmarks algorithms. As shown in Figs. 5c and 5d, the ORP-HS outperforms all the other algorithms (including the benchmark algorithms in Fig. 5c and the two proposed ORP algorithms in Fig. 5d. In ORP-HS, the system cost can be minimized by achieving an optimal balance between lower pricing rate and higher resource utilization.

Therefore, the local-first and ORP-R algorithms can not scale well with dynamics of mobile requests, yet the ORP-OD and ORP-HS algorithms can achieve higher flexibility by leasing elastic on-demand instances.

5.1.2 Delay-Sensitive Dominant Resource Provisioning

According to above simulation results and analysis, ORP-HS can always outperform ORP-OD and ORP-R in reducing system cost and dealing with dynamics of mobile requests. Thus, in the resource provisioning for delay-sensitive dominant mobile requests (1/6 delay-tolerant requests and 5/6 delay-sensitive requests), the performance of ORP-HS is mainly evaluated. Simulation results are shown in Fig. 6.

In this scenario, substantial edge hosts should be deployed at mobile edge to serve delay-sensitive requests. Thus to ensure the QoS at rushing hours, the computation capacity of edge hosts increases rapidly with dynamics of requests. For the ORP-HS and cloud-first algorithms, cloud instances can be leased on demand to achieve high flexibility, while for the local-first algorithm, the QoS can only be ensured by increasing edge hosts, resulting in significantly increasing system cost. According to ORP-HS, when the variance of requests exceeds 1.5 Giga CPU cycles/second, all delay-tolerant requests are outsourced to cloud as the pricing rate of edge hosts is larger than cloud instances. Then ORP-HS introduces the same system cost with the cloud-first algorithm in this case. Therefore, in the resource provisioning with delay-sensitive dominant requests, the performance of ORP-HS is similar to the cloud-first algorithm, especially for the requests with higher dynamics.

5.2 Trace-Driven Evaluation

In this part, trace-driven experiments are preformed to evaluate the performance of ORP algorithms in real life. As MEC has not yet been widely deployed in practical life, computation requests at mobile edge can not be traced precisely. This work employs Google cluster usage traces [40] to evaluate the ORP algorithms. In a Google cluster, a Google compute cell typically consists of a set of machines that are connected by a high-bandwidth network. Since mobile edge can be considered as a cloud of small range, user data of a Google compute cell is used to approximate the user data of mobile edge. The Google cluster tracelogs[4] record the information of tasks in a Google compute cell. The tasks are described by task event tables and task resource usage tables. Task event tables record task event information, such as event types (submit, schedule, fail, finish, etc.), job IDs, task indexes and timestamps when these events happen. With task resource tables, average CPU usage of tasks (denoted as U^{cpu}) can be calculated.

According to [41], the computation requests of the tasks can be computed as

$$R^{\text{comp}} = (t^{\text{finish}} - t^{\text{schedule}}) \cdot U^{\text{cpu}} \cdot C^{\text{cpu}}. \tag{39}$$

Here, t^{finish} and t^{schedule} represent the timestamps the task is finished and scheduled to machines respectively. C^{cpu} is average computation capacity of CPU in the Google cloud. Then, the arrival rate of computation requests can be calculated as

$$\lambda = R^{\text{comp}} \cdot a^{\text{task}} \tag{40}$$

where a^{task} represents the arrival rate of the tasks. In MEC, computation requests are much more delay-sensitive and have less computation requirements than those in conventional cloud computing. Thus, the arrival rates of computation requests at mobile edge can be obtained by slightly modifying the results of Google cluster tracelogs, as shown in Fig. 7. Five groups of trace results are illustrated in this figure. It can be observed that Trace-data 2, 3, 4 are greatly fluctuated while Trace-data 1 and 5 are less fluctuated.

Experiments are performed based on these trace data, and the results are illustrated in Fig. 8. As shown in Figs. 8a, 8b, and 8c, the proposed ORP algorithms always yield the

4. http://code.google.com/p/googleclusterdata/wiki/ClusterData2011_1.

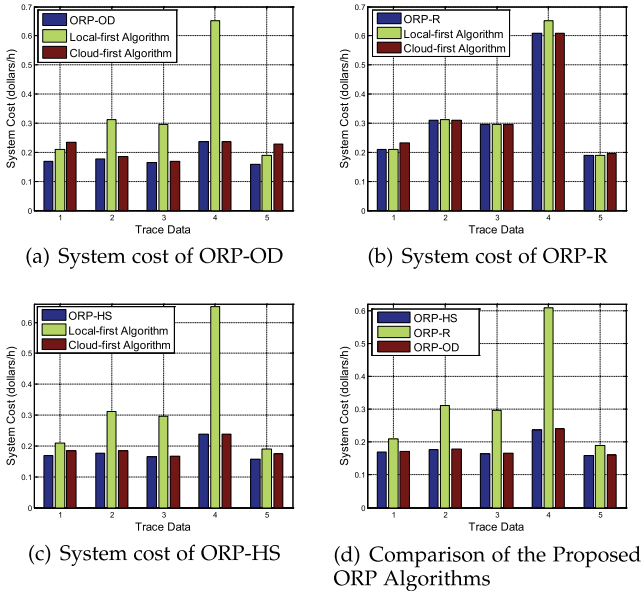


Fig. 8. Performance of different algorithms in trace-driven experiments.

minimum system cost compared with the benchmark algorithms. When arrival rates of mobile requests are fiercely fluctuated (Trace-data 2, 3, 4), the ORP-HS and ORP-OD algorithms yield similar results to the cloud-first algorithm. As in ORP-OD and ORP-HS, cloud resources are tenanted on demand to achieve elastic matching of resource provisioning and dynamic requests. Nevertheless, in the local-first algorithm, edge operators have to deploy substantial edge hosts to meet QoS requirements at rushing hours, resulting in much higher edge provisioning cost (as shown in Figs. 8a and 8c).

Fig. 8d shows the comparison among the three ORP algorithms. The results show that the ORP-R algorithm requires much more system cost than the other algorithms. This is straightforward, as fixed computing capacities are provisioned both at mobile edge and cloud in ORP-R. Low resource utilization and high system cost are incurred when dealing with dynamic requests. In ORP-HS, on-demand instances are mainly leased to serve fluctuated requests, thus introduce similar results to ORP-OD.

Insight. When mobile requests are fiercely fluctuated (Fig. 8) or mobile requests are dominated by delay-sensitive mobile requests (Fig. 6), the results of the ORP algorithms are similar to the cloud-first algorithm, which are much smaller than the local-first algorithm. In these cases, directly offloading all delay-tolerant mobile requests to remote clouds can achieve desired performance. When considering more general cases, extensive simulation results demonstrate that the proposed ORP algorithms can constantly reduce the system cost compared with the benchmark algorithms. In addition, the ORP algorithms show high flexibility with increasing dynamics of mobile requests. Among the three ORP algorithms, the ORP-HS algorithm outperforms the ORP-OD and ORP-R algorithms, as the optimal combination of on-demand and reserved instances can be achieved by solving *(P5)*.

6 Conclusions

In this paper, CAME framework has been introduced to enhance the scalability of MEC when dealing with time varying mobile requests. The resource provisioning problem with multiple cloud instances has been solved, considering different QoS requirements of mobile requests. By leveraging piecewise convexity of this problem, ORP algorithms have been proposed to determine the optimal computation capacity of edge hosts, based on which the optimal cloud tenancy strategy is further explored. The proposed ORP algorithms have been shown to outperform the local-first and cloud-first benchmark algorithms in system flexibility and cost-efficiency. In addition, when dealing with greatly fluctuated mobile requests or delay-sensitive dominant mobile requests, desired performance can be achieved by directly offloading all delay-tolerant mobile requests to remote clouds. For the future work, we will investigate the multiple edge nodes case and jointly address the resource provisioning and workload scheduling issues, where multiplexing gain will be explored.

Acknowledgments

This work was supported by the National Key R&D Program of China (2018YFB1004801) and the National Science Foundation of China (61602054; 61801011). A preliminary version of this paper appears as a conference paper in IEEE GLOBECOM 2017 [32].

References

[1] T. Soyata, R. Muraleedharan, C. Funai, M. Kwon, and W. Heinzelman, "Cloud-vision: Real-time face recognition using a mobile-cloudlet-cloud acceleration architecture," in *Proc. IEEE Symp. Comput. Commun.*, Jul. 2012, pp. 59–66.

[2] T. Chen, L. Ravindranath, S. Deng, P. Bahl, and H. Balakrishnan, "Glimpse: Continuous, real-time object recognition on mobile devices," in *Proc. ACM 13th ACM Conf. Embedded Netw. Sensor Syst.*, Nov. 2015, pp. 155–168.

[3] B. G. Chun, S. Ihm, P. Maniatis, M. Naik, and A. Patti, "Clonecloud: Elastic execution between mobile device and cloud," in *Proc. ACM 6th Conf. Comput. Syst.*, Apr. 2011, pp. 301–314.

[4] E. Cuervo, A. Balasubramanian, D. Cho, A. Wolman, S. Saroiu, R. Chandra, and P. Bahl, "MAUI: Making smartphones last longer with code offload," in *Proc. ACM 8th Int. Conf. Mobile Syst. Appl. Serv.*, Jun. 2010, pp. 49–62.

[5] K. Ashton, "That internet of things thing," *RFiD J.*, vol. 22, pp. 97–114, Jan. 2009.

[6] Cisco, "Cisco global cloud index: Forecast and methodology, 2016C2021," white paper, 2018.

[7] ETSI Group Specification, "Mobile Edge Computing (MEC); Framework and reference architecture," ETSI GS MEC 003 V1.1.1, 2016. [Online]. Available: https://www.etsi.org/deliver/etsi_gs/mec/001_099/003/01.01.01_60/gs_mec003v010101p.pdf. [Accessed: Sep. 3, 2018]

[8] A. Li, X. Yang, S. Kandula, and M. Zhang, "CloudCmp: Comparing public cloud providers," in *Proc. 10th ACM SIGCOMM Conf. Internet Meas.*, Nov. 2010, pp. 1–14.

[9] M. Claypool and K. Claypool, "Latency and player actions in online games," *Commun. ACM*, vol. 49, no. 11, pp. 40–45, Nov. 2006.

[10] K. Ha, P. Pillai, W. Richter, Y. Abe, and M. Satyanarayanan, "Just-in-time provisioning for cyber foraging," in *Proc. 11th Annu. Int. Conf. Mobile Syst. Appl. Services*, Jun. 2013, pp. 153–166.

[11] X. Ma, S. Zhang, W. Li, P. Zhang, C. Lin, and X. Shen, "Cost-Efficient workload scheduling in cloud assisted mobile edge computing," in *Proc. IEEE/ACM 25th Int. Symp. Quality Service*, Jun. 2017, pp. 1–10.

[12] X. Sun and N. Ansari, "Avaptive avatar handoff in the cloudlet network," *IEEE Trans. Cloud Comput.*, May 2018, to appear, DOI: 10.1109/TCC.2017.2701794.

[13] S. Zhang, P. He, K. Suto, P. Yang, L. Zhao, and X. Shen, "Cooperative edge caching in user-centric clustered mobile networks," *IEEE Trans. Mobile Comput.*, vol. 17, no. 8, pp. 1791–1805, Aug. 2018.

[14] S. Sarkar, S. Chatterjee, and S. Misra, "Assessment of the suitability of fog computing in the context of internet of things," *IEEE Trans. Cloud Comput.*, vol. 6, no. 1, pp. 46–59, Jan. 2018.
[15] M. Aazam and E. Huh, "Fog computing micro datacenter based dynamic resource estimation and pricing model for IoT," in *Proc. IEEE 29th Int. Conf. Advanced Inf. Netw. Appl.*, Mar. 2015, pp. 687–694.
[16] M. Jia, J. Cao, and W. Liang, "Optimal cloudlet placement and user to cloudlet allocation in wireless metropolitan area networks," *IEEE Trans. Cloud Comput.*, vol. 5, no. 4, pp. 725–737, Oct. 2017.
[17] A. Kiani, N. Ansari, and A. Khreishah, "Hierarchical capacity provisioning for fog computing," arXiv preprint arXiv:1807.01093, Jul. 2018. [Online]. Available: https://arxiv.org/abs/1807.01093. [Accessed: Sep. 1, 2018].
[18] G. Fortino, D. Parisi, V. Pirrone, and G. D. Fatta, "BodyCloud: A SaaS approach for community body sensor networks," *Future Generation Comp. Syst.*, vol. 35, pp. 62–79, Jun. 2014.
[19] P. Yang, N. Zhang, S. Zhang, L. Yu, J. Zhang, and X. Shen, "Content popularity prediction towards location-aware mobile Edge caching," arXiv preprint arXiv:1809.00232, Sep. 2018. [Online] Available: https://arxiv.org/abs/1809.00232. [Accessed: Dec. 3, 2018].
[20] V. Nae, A. Iosup, and R. Prodan, "Dynamic resource provisioning in massively multiplayer online games," *IEEE Trans. Parallel Distrib. Syst.*, vol. 22, no. 3, pp. 380–395, 2011.
[21] B. Hack, M. Morhaime, J. F. Grollemund, and N. Bradford, "Introduction to vivendi games," Presentation, Jun. 2006. [Online]. Available: http://www.vivendi.com/
[22] J. L. D. Neto, S. Yu, D. F. Macedo, J. M. S. Nogueira, R. Langar, and S. Secci, "ULOOF: A user level online offloading framework for mobile edge computing," *IEEE Trans. Mobile Comput.*, vol. 17, no. 11, pp. 2660–2674, Mar. 2018.
[23] M. V. Barbera, S. Kosta, A. Mei, and J. Stefa, "To offload or not to offload? the bandwidth and energy costs of mobile cloud computing," in *Proc. IEEE INFOCOM*, Jul. 2013, pp. 1285–1293.
[24] K. Kumar and Y. H. Lu, "Cloud computing for mobile users: Can offloading computation save energy?" *IEEE Comput.*, vol. 43, no. 4, pp. 51–56, Apr. 2010.
[25] L. Tong, Y. Li, and W. Gao, "A hierarchical edge cloud architecture for mobile computing," in *Proc. 35th Annu. IEEE Int. Conf. Comput. Commun.*, Jul. 2016, pp. 1–9.
[26] A. Enayet, Md. A. Razzaque, M. M. Hassan, A. Alamri, and G. Fortino, "A mobility-aware optimal resource allocation architecture for big data task execution on mobile cloud in smart cities," *IEEE Commun. Mag.*, vol. 56, no. 2, pp. 110–117, Feb. 2018.
[27] B. Sotomayor, R. S. Montero, I. M. Llorente, and I. Foster, "Virtual infrastructure management in private and hybrid clouds," *IEEE Internet Comput.*, vol. 13, no. 5, pp. 14–22, Sep. 2009.
[28] R. V. Bossche, K. Vanmechelen, and J. Broeckhove, "Cost-optimal scheduling in hybrid IaaS clouds for deadline constrained workloads," in *Proc. IEEE 3rd Int. Conf. Cloud Comput.*, Jul. 2010, pp. 228–235.
[29] Amazon EC2 pricing, 2018. [Online] Available: https://aws.amazon.com/cn/ec2/. [Accessed: Sep. 1, 2018].
[30] P. Yang, N. Zhang, Y. Bi, L. Yu, and X. Shen, "Catalyzing cloud-fog interoperation in 5G wireless networks: An SDN approach," *IEEE Netw.*, vol. 31, no. 5, pp. 14–20, Sep./Oct. 2017.
[31] Y. Niu, B. Luo, F. Liu, J. Liu, and B. Li, "When hybrid cloud meets flash crowd: Towards cost-effective service provisioning," in *Proc. IEEE Conf. Comput. Commun.*, Apr. 2015, pp. 1044–1052.
[32] X. Ma, S. Zhang, P. Yang, N. Zhang, C. Lin, and X. Shen, "Cost-efficient resource provisioning in cloud assisted mobile edge computing," in *Proc. IEEE Global Commun. Conf.*, Dec. 2017. pp. 1–6.
[33] H. Pishro-Nik, *Introduction to Probability, Statistics, and Random Process*, Kappa Research: Galway, Ireland, 2014.
[34] B. Gnedenko and I. N. Kovalenko, *Introduction to Queuing Theory. Mathematical Modeling*, Boston, MA, USA: Birkhaeuser, 1989.
[35] A. Greenberg, J. Hamilton, D. A. Maltz, and P. Patel, "The cost of a cloud: research problems in data center networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 39, no. 1, pp. 68–73, Dec. 2008.
[36] T. Zhao, S. Zhou, X. Guo, Y. Zhao, and Z. Niu, "Pricing policy and computational resource provisioning for delay-aware mobile edge computing," in *Proc. IEEE/CIC Int. Conf. Commun. China*, Jul. 2016, pp. 1–6.
[37] "ThinkServerRD450," 2018. [Online] Available: http://b2b.lenovo.1048 com.cn/Search/search?content=THINKSERVER/. [Accessed: Sep. 3, 2018].
[38] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.
[39] S. Zhang, J. Gong, S. Zhou, and Z. Niu, "How many small cells can be turned off via vertical offloading under a separation architecture?" *IEEE Trans. Wireless Commun.*, vol. 14, no. 10, pp. 5440–5453, Jun. 2015.
[40] Google, "Google Cluster-Usage Traces: Format + Schema," *White Paper*, 2011.
[41] I. S. Moreno, P. Garraghan, P. Towend, and X. Jie, "An approach for characterizing workloads in Google cloud to derive realistic resource utilization models," in *Proc. IEEE 7th Int. Symp. Service-Oriented Syst. Eng.*, Jun. 2013, pp. 49–60.
[42] R. Gravina, C. Ma, P. Pace, G. Aloi, W. Russo, W. Li, and G. Fortino, "Cloud-based activity-aaService cyber-physical framework for human activity monitoring in mobility," *Future Generation Comp. Syst.*, vol. 75, pp. 158–171, Oct. 2017.

Xiao Ma received the PhD degree from the Department of Computer Science and Technology, Tsinghua University, and the BS degree in telecommunication engineering from the Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 2018 and 2013, respectively. She is currently a postdoctoral fellow with the State Key Laboratory of Networking and Switching Technology, BUPT. From October 2016 to April 2017, she visited the Department of Electrical and Computer Engineering, University of Waterloo, Canada. Her research interests include task scheduling, resource deployment and allocation in mobile cloud computing and mobile edge computing. She is a member of the IEEE.

Shangguang Wang received the PhD degree from the Beijing University of Posts and Telecommunications, in 2011. He is a professor and vice-director at the State Key Laboratory of Networking and Switching Technology (BUPT). He has published more than 150 papers, and played a key role at many international conferences, such as general chair and PC chair. His research interests include service computing, cloud computing, and mobile edge computing. He is a senior member of the IEEE, and the editor-in-chief of the International Journal of Web Science.

Shan Zhang (S'13-M'16) received the PhD degree from the Department of Electronic Engineering, Tsinghua University, and the BS degree from the Department of Information, Beijing Institute Technology, Beijing, China, in 2016 and 2011, respectively. She is currently with the School of Computer Science and Engineering, Beihang University, Beijing, China. From August 2016 to December 2017, she was a post doctoral fellow in Department of Electronical and Computer Engineering, University of Waterloo, Ontario, Canada. Her research interests include network resource and traffic management, network virtualization and integration. She received the Best Paper Award at the Asia-Pacific Conference on Communication in 2013. She is a member of the IEEE.

Peng Yang received the PhD and BE degrees from the School of Electronic Information and Communications, Huazhong University of Science and Technology, Wuhan, China, in 2013 and 2018, respectively. He is currently a postdoctoral fellow with the Department of Electrical and Computer Engineering, University of Waterloo, Ontario, Canada. His research focuses on software defined networking, network function virtualization and mobile edge computing. He is a member of the IEEE.

Chuang Lin received the PhD degree in computer science from Tsinghua University, China, in 1994. He is a professor of the Department of Computer Science and Technology, Tsinghua University, Beijing, China. He is an honorary visiting professor, University of Bradford, United Kingdom. His current research interests include computer networks, performance evaluation, network security analysis, and Petri net theory and its applications. He has published more than 300 papers in research journals and IEEE conference proceedings in these areas and has published four books. He is a senior member of the IEEE and the Chinese Delegate in TC6 of IFIP. He served as the Technical Program vice chair, the 10th IEEE Workshop on Future Trends of Distributed Computing Systems (FTDCS 2004); the General Chair, ACM SIGCOMM Asia workshop 2005 and the 2010 IEEE International Workshop on Quality of Service (IWQoS 2010). He is an associate editor of the *IEEE Transactions on Vehicular Technology* and an area editor of the *Computer Networks and the Journal of Parallel and Distributed Computing*.

Xuemin Shen (M'97-SM'02-F'09) received the PhD degree in electrical engineering from Rutgers University, New Brunswick, NJ, in 1990. He is currently a University professor with the Department of Electrical and Computer Engineering, University of Waterloo, Canada. His research focuses on resource management in interconnected wireless/wired networks, wireless network security, social networks, smart grid, and vehicular ad hoc and sensor networks. He is a registered Professional Engineer of Ontario, Canada, an Engineering Institute of Canada fellow, a Canadian Academy of Engineering fellow, a Royal Society of Canada fellow, and a distinguished lecturer of the IEEE Vehicular Technology Society and Communications Society. He received the James Evans Avant Garde Award in 2018 from the IEEE Vehicular Technology Society, the Joseph LoCicero Award in 2015 and the Education Award in 2017 from the IEEE Communications Society. He has also received the Excellent Graduate Supervision Award in 2006 and the Outstanding Performance Award in 2004, 2007, 2010, and 2014 from the University of Waterloo and the Premiers Research Excellence Award (PREA) in 2003 from the Province of Ontario, Canada. He served as the Technical Program Committee chair/co-chair for the IEEE Globecom16, the IEEE Infocom14, the IEEE VTC10 Fall, the IEEE Globecom07, the Symposia Chair for the IEEE ICC10, the Tutorial Chair for the IEEE VTC11 Spring, the chair for the IEEE Communications Society Technical Committee on Wireless Communications, and P2P Communications and Networking. He is the editor-in-chief of the *IEEE Internet of Things Journal* and the vice president on Publications of the IEEE Communications Society. He is a fellow of the IEEE.