Distributed Online Hybrid Cloud Management for Profit-Driven Multimedia Cloud Computing

Ping Lu, Quanying Sun, Kaiyue Wu, and Zuqing Zhu, *Senior Member, IEEE*

***Abstract*—It is known that with a hybrid cloud, a multimedia cloud service provider (MCSP) can quickly extend its services to multiple geographical locations with quality-of-service (QoS) guarantees. Meanwhile, to maximize its profit, the MCSP needs an online management mechanism to operate the hybrid cloud efficiently. In this paper, we study how to maximize an MCSP's profit from provisioning multimedia services to geographically distributed users with a hybrid cloud. We first design a service provisioning model to manage the resources in the hybrid cloud. Here, in order to make the model practical and address the different situations in private and public clouds, we consider different time granularities for resource reservations. Then, we leverage the Lyapunov optimization technique to maximize the profit of MCSP and propose an online algorithm that can manage the hybrid cloud in the distributed manner. Specifically, the algorithm determines the access control and routing of each multimedia service request, and allocates the resources in the hybrid cloud accordingly. We also apply the ϵ-persistent technique to ensure that the worst-case latency of the provisioned requests is bounded. Finally, the proposed algorithm is evaluated with extensive simulations using both synthetical and real traces. Simulation results indicate that the algorithm can manage the hybrid cloud efficiently and maximize the profit of MCSP.**

***Index Terms*—Datacenter management, hybrid cloud, Lyapunov optimization, QoS-aware requests scheduling.**

I. Introduction

RECENTLY, with the rapid development of the Internet, especially the mobile networks, the demands on multimedia services, such as online gaming, social network, video conference, etc., are growing exponentially. These services usually associate with a sequence of media processing tasks, e.g., caching, transcoding and rendering, which may consume large amounts of computing and storage resources [1]. Meanwhile, the tasks can be highly dynamic and exhibit huge peak resource requirements. Moreover, since the multimedia services can occupy a lot of computing resources and hence can consume a large amount of energy, how to support them efficiently on energy-constrained devices (e.g., smartphones) will be challenging [2], [3]. Therefore, in order to handle the multimedia services cost-effectively, more and more service providers adopt the cloud infrastructure, as it can respond to demands timely and allocate computing, communication and storage resources adaptively according to the requirements [4]. Due to the fact that the computing tasks are off-loaded to the cloud, the energy-consumption of the end-user devices can also be reduced significantly [5], [6].

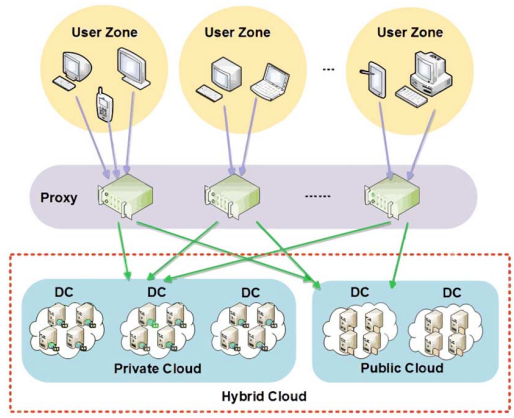

Fig. 1. Multimedia services in hybrid cloud environment.

The quality-of-service (QoS) of multimedia cloud computing depends heavily on the underlying network architecture. It is known that deploying datacenters (DCs) close to end users can reduce network latency, improve user experience, and help to promote the services to multiple geographical locations [7]. Hence, there is an increasing need to build geographically distributed (geo-distributed) cloud systems with multiple DCs. However, due to the high capital expenditure (CAPEX) and operational expenditure (OPEX) of geo-distributed multi-DC cloud systems, it is not realistic for most of the multimedia cloud service providers (MCSPs) to build the systems on their own. Therefore, the hybrid cloud in Fig. 1 becomes a cost-effective alternative [8], which uses both private and public DCs. Depending on the scale of its applications, an MCSP can own and operate a few geo-distributed DCs as the private cloud. Meanwhile, if the capacity of the private cloud is not enough, it can rent resources from a public cloud and create virtual machines (VMs) dynamically.

Manuscript received January 18, 2015; revised May 03, 2015; accepted May 28, 2015. Date of publication June 02, 2015; date of current version July 15, 2015. This work was supported in part by the NSFC Project 61371117, in part by the Fundamental Research Funds for the Central Universities under Grant WK2100060010, in part by the Natural Science Research Project for Universities in Anhui under Grant KJ2014ZD38, and in part by the Strategic Priority Research Program of the CAS under Grant XDA06011202. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Ali Begen.

The authors are with the School of Information Science and Technology, University of Science and Technology of China, Hefei 230027, China (e-mail: lpbest@mail.ustc.edu.cn; sqy0410@mail.ustc.edu.cn; wooloo@mail.ustc.edu.cn; zqzhu@ieee.org).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMM.2015.2441004

With hybrid cloud, an MCSP can quickly extend its services to multiple geographical locations with QoS guarantees. However, in order to manage the hybrid cloud well and provide multimedia services to geo-distributed end users, one needs to properly address several issues. First of all, the latency that end users in different locations experience should be maintained carefully to ensure consistent QoS. That is to say, the worst-case service latency should be bounded for all the users. Secondly, we need to manage the servers in private cloud and the VMs in public cloud coordinately so that the service-level agreement (SLA) violations where the end users encounter can be minimized and the profit of the MCSP can be maximized. Thirdly, an efficient online algorithm that can achieve profit-driven request scheduling and resource management for the multimedia cloud computing is highly desired. Specifically, the algorithm should be able to handle the service requests dynamically according to their arrival pattern and obtain the optimal profit for the MCSP. Finally, because the hybrid cloud is geo-distributed, we need to realize the online hybrid cloud management in the distributed manner for better scalability.

In this work, we investigate how to maximize an MCSP's profit from provisioning multimedia services to geo-distributed users with hybrid cloud. We first design a service provisioning model to manage the resources in hybrid cloud. Here, in order to make the model practical and address the different situations in private and public clouds, we consider different time granularities for resource reservations. The theoretical analysis on offline optimization is then provided based on the assumption on the distribution of the requests' arrivals. Next, we apply the Lyapunov optimization technique [9] on the model to maximize the profit of MCSP and propose an online algorithm that can manage the hybrid cloud in the distributed manner. Specifically, the algorithm determines the access control and routing of each multimedia service request, and allocates the resources in the hybrid cloud accordingly. Meanwhile, we leverage the ϵ-persistent technique [9] to ensure that the worst-case latency of the provisioned requests is bounded. Finally, we evaluate our proposal with extensive simulations using both synthetical and real traces.

The contributions of our work can be summarized as follows.

- We propose an online algorithm for hybrid cloud management, which can maximize the MCSP's time-average profit and ensure that the worst-case latency of all the provisioned requests is bounded.
- We make sure that the proposed algorithm operates in the distributed manner, i.e., the request scheduling in each zone proxy and the resource allocation in every private or public DC are handled independently, while only a small amount of status information needs to be exchanged among the proxies and DCs.
- We consider different time granularities for the resource reservations in private and public DCs, and prove that the proposed algorithm is within a constant gap from the profit-optimal performance while it can achieve this without any pre-knowledge on the requests' arrival pattern.

The rest of the paper is organized as follows. Section II provides a survey of the related work. In Section III, we design the service provisioning model for resource management in hybrid cloud. The theoretical analysis on offline optimization is provided in Section IV, and the online scheduling algorithm is proposed in Section V, in which we also analyze the delays of the provisioned requests and the profit optimality. We present the performance evaluation with simulations in Section VI. Finally, Section VII summarizes the paper.

II. Related Work

As more and more multimedia services have been migrated to the cloud infrastructure, the service scheduling and resource management for multimedia cloud computing have attracted a lot of research interests. Previous studies have considered how to manage multimedia services in single-DC-based cloud environment [10]–[14]. Nan *et al.* investigated the workload scheduling and resource allocation in such clouds, and used the queueing theory to evaluate the QoS of multimedia services [10]–[12]. In [13], the authors leveraged the queueing networks to analyze the capacity of video-on-demand (VoD) in single-DC clouds and designed an algorithm for dynamic resource allocation. The mechanisms to realize long-term cost-optimal video transcoding in multimedia clouds were studied in [14].

As geo-distributed DCs can provide multimedia services to multiple regions with enhanced user experience, people have also investigated the multimedia cloud computing that uses them. In [15], the authors proposed a demand prediction method and used both the one-shot and Δ-shot mechanisms to minimize the total costs of service provisioning of social media applications in geo-distributed DCs. An interesting request redirection scheme for cloud-centric media networks was designed in [16], which dispatches requests to scaling-enabled VMs for minimizing the total cost of service provisioning. Note that both of these studies conducted optimization based on the pre-knowledge of the requests' arrival pattern, either through prediction or detection. Lin *et al.* leveraged genetic algorithm to optimize the multi-service task scheduling for multimedia services in a hierarchical cloud system [17]. The authors of [18] studied the networking scheme among DCs for multimedia clouds and proposed a heuristic algorithm to minimize the resource cost. However, since the algorithms proposed in [17], [18] are not exact ones, they cannot provide the optimum solution, especially the long-term optimum.

With the idea of hybrid cloud, one can promote the advantages of geo-distributed DCs cost-effectively [19]–[22]. Zhang *et al.* proposed to categorize the service requests as base and flash-crowd workloads and then serve them in private and public DCs, respectively [23]. By doing so, they realized proactive workload management in hybrid cloud, which can improve resource utilization and reduce caching/replication overheads as well. They used a different service model from ours, and we do not need to detect the arrival pattern of the requests. By analyzing the real traces of two VoD applications, the authors of [24] introduced several hybrid cloud based strategies to minimize the bandwidth cost of carrying VoD.

Since its introduction, Lyapunov optimization [9] has been considered as a powerful technique for task scheduling because it can achieve the time-average optimum within a constant gap

to it without any pre-knowledge on tasks' statistical information. Zhang *et al.* used Lyapunov optimization technique to reduce the energy cost of video transcoding in single-DC-based multimedia clouds [25]. In [26], an online algorithm for scheduling jobs in geo-distributed DCs was proposed to optimize energy consumption and service fairness jointly with Lyapunov optimization. However, these two studies tried to control the requests' queueing delay by adjusting the queue lengths, which might not be flexible and adaptive.

The cloud-based content delivery schemes that distribute multimedia contents to end users have been studied in [27], [28]. Hu *et al.* discussed the problem of online request dispatch and video deployment in cloud-centric content delivery network (CDN) [27]. They proposed an algorithm to ensure that the mean latency is bounded for the requests. Note that since the cloud-based content delivery needs to consider request dispatch and contents placement, it is different from the multimedia cloud computing that we study in this work. Specifically, our work needs to address the request scheduling and resource management (e.g., servers or virtual machines (VMs) provisioning) in hybrid cloud. Moreover, our proposed algorithm can guarantee that each provisioned request is completed within a worst-case latency, i.e., making the worst-case latency but not the mean latency bounded. The work in [28] leveraged Lyapunov optimization technique to manage the content delivery services in hybrid cloud. As it was for content delivery, the work only addressed content placement and migration in the private and public DCs, but did not consider how to allocate resources to provision servers or VMs. While in our work, we study the resource management in hybrid cloud for multimedia cloud computing and conduct optimization with the consideration of different time granularities for resource reservations in private and public DCs. More importantly, the proposed algorithm in [28] has to be implemented in the centralized way for arranging the content placement, while our proposal can operate distributedly.

III. System Model and Problem Formulation

In this section, we describe the service provisioning model for resource management in hybrid cloud and formulate the problem of profit-driven request scheduling and resource management for multimedia cloud computing.

A. Multimedia Services in Hybrid Cloud

We consider a hybrid cloud as illustrated in Fig. 1, in which multimedia service requests are processed in the DCs and the proxies handle the access control and routing of the requests. In this work, we assume that the hybrid cloud operates as a discrete-time system [29] and the service provisioning scheme in it can be changed every time interval Δt. Hence, we can normalize system time with the time slot (TS) Δt and obtain the normalized system time as $t = \{1, 2, \ldots\}$. Note that the actual choice of Δt depends on the control and management mechanism in the hybrid cloud. Basically, Δt should be at least longer than the maximum round-trip time (RTT) among the proxies and the DCs to ensure effective information exchanges, and the study in [28] suggested that Δt should be on the magnitude of seconds or tens of seconds.

We denote the DCs in the hybrid cloud with set $\mathcal{D}$ and assign an index j to each of them. Without loss of generality, we assume that the first $|\mathcal{M}|$ DCs in $\mathcal{D}$ (i.e., the indices satisfy $j \in [1, |\mathcal{M}|]$) are private DCs (denoted with set $\mathcal{M}$), while the rest $|\mathcal{N}|$ DCs in $\mathcal{D}$ are public ones (denoted with set $\mathcal{N}$). Hence, we have $\mathcal{D} = \mathcal{M} \cup \mathcal{N}$. In a private DC j, the MCSP installs S_j servers for multimedia cloud computing. On the other hand, the rented resources in a public DC j can support at most S_j VMs. Apparently, the servers and VMs should carry homogenous resources for serving the multimedia service requests. Then, the resources in each server or VM in a DC j can be quantified as e_j, which means that each server or VM in the DC can process e_j requests within one TS.

The MCSP provides multimedia services to end users located in a few geo-distributed zones (denoted with set $\mathcal{Z}$). We assign an index i to each zone. There is a service proxy in each zone, and the requests from end users are first sent to the proxies in their zones and then routed to the hybrid cloud for further processing. We use $A_i(t)$ to represent the number of requests that the proxy in zone i receives within TS t. We define A_{max} as the maximum arrival rate of the requests, i.e.

$$A_i(t) \leq A_{max}, \qquad \forall i, t.$$

For the system, we define A_i as the time-average expectation number of the incoming requests in zone i, i.e.

$$A_i = \lim_{T \to \infty} \frac{1}{T} \sum_{t=1}^{T} A_i(t).$$

There is an SLA on the service provisioning of the requests, which means that each request should be processed within duration d, otherwise, it will be dropped due to SLA violation.

B. Requests Scheduling in Service Proxy

Since the incoming requests in zone i will be aggregated at the proxy before being routed to one of the DCs for further processing, we introduce a decision variable $a_i(t)$ to denote the number of requests that the proxy in zone i admits in TS t. Apparently, we have

$$0 \leq a_i(t) \leq A_i(t), \qquad \forall i \in \mathcal{Z} \tag{1}$$

and define the time-average expectation number of admitted requests in zone i by a_i. To avoid unnecessary latency, each proxy routes all the admitted requests to their corresponding DCs in the current TS.

We also introduce another set of decision variables $\{a_{i,j}(t)\}$, each of which denotes the number of requests that DC j receives from zone i in TS t. Then, we have

$$a_i(t) = \sum_{j \in \mathcal{D}} a_{i,j}(t), \qquad \forall i \in \mathcal{Z}. \tag{2}$$

And we define $a_{i,j}$ as the time-average expectation number of requests that DC j receives from zone i, then we have $a_i = \sum_{j \in \mathcal{D}} a_{i,j}, \forall i$.

C. Resource Management in Hybrid Cloud

It is known that when the workload is relatively low, we can switch the servers in the private DCs from working to idling and

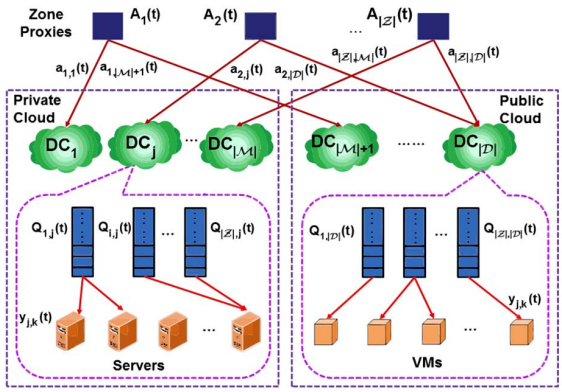

Fig. 2. Service provisioning model for multimedia services in hybrid cloud.

achieve reduction on OPEX (e.g., the energy cost). Since the TS in the system is usually much longer than the duration of the switching transition [29], we can ignore the transition overhead in following analysis. Similarly, the VMs in the public DCs can also be released dynamically for cost saving. Then, the state of the k-th server/VM in DC j in TS t can be represented with a decision variable $y_{j,k}(t)$, such that

$$y_{j,k}(t) = \begin{cases} 1, & \text{server/VM is in working state} \\ 0, & \text{server/VM is in idle/released state.} \end{cases} \tag{3}$$

If we denote the total number of working servers/VMs in DC j in TS t as $y_j(t)$, we have

$$y_j(t) = \sum_{k=1}^{S_j} y_{j,k}(t), \qquad \forall j \in \mathcal{D} \tag{4}$$

and $y_j(t)$ is bounded by S_j as

$$y_j(t) \le S_j, \qquad \forall j \in \mathcal{D}. \tag{5}$$

Let y_j and $y_{i,k}$ denote the time-average expectation value of $y_j(t)$ and $y_{j,k}(t)$, respectively. Then we have $y_j = \sum_{k=1}^{S_j} y_{j,k}, \forall j$.

Meanwhile, we notice that the VMs provided by public DCs are usually charged based on a relatively long time granularity, e.g., on the magnitude of hours or even longer. Therefore, we define the time granularity for VM rentals as T_g, which is also normalized with the duration of TS (i.e., Δt). Then, once a VM is rented, the MCSP needs to pay for a whole T_g at least. Hence, the MCSP should try to make full use of the VM within T_g, otherwise, its down payment would be wasted. This means that in the hybrid cloud, once a VM is rented, it will stay in the working state for at least T_g. To model this, we introduce an auxiliary parameter $Y_{j,k}(t)$ for the k-th VM in DC j to record the remaining TS' in the current T_g, and have

$$T_g \cdot y_{j,k}(t) \ge Y_{j,k}(t), \qquad \forall j \in \mathcal{N}. \tag{6}$$

As shown in Fig. 2, we make the requests from different zones be buffered in independent queues in each DC for further processing. $Q_{i,j}(t)$ denotes the number of buffered requests from zone i in TS t in DC j. We then define a decision variable $x_{i,j}(t)$ to represent the number of working servers/VMs that are processing the requests in $Q_{i,j}(t)$. To avoid wasting the resources, we apply an upper-bound on $x_{i,j}(t)$ as

$$0 \le x_{i,j}(t) \le \left\lceil \frac{Q_{i,j}(t)}{e_j} \right\rceil, \qquad \forall i \in \mathcal{Z}, j \in \mathcal{D}. \tag{7}$$

The total number of working servers/VMs in DC j in TS t for request-processing is

$$x_j(t) = \sum_{i \in \mathcal{Z}} x_{i,j}(t). \tag{8}$$

Apparently, $x_j(t)$ and $y_j(t)$ have the relation

$$x_j(t) \le y_j(t), \qquad \forall j \in \mathcal{D}. \tag{9}$$

We define x_j and $x_{i,j}$ as the time-average expectation number of $x_j(t)$ and $x_{i,j}(t)$, respectively. Then we have $x_j = \sum_{i \in \mathcal{Z}} x_{i,j}, \forall j$.

To minimize SLA violations, we may purposely drop certain buffered requests in the DCs to make sure that the rest ones can be processed timely, i.e., within the worst-case latency d. We use $b_{i,j}(t)$ to represent the number of requests that are dropped from $Q_{i,j}(t)$ in TS t, and $b_{i,j}(t)$ should be bounded by A_{max}, which is the maximum arrival rate of the requests.

$$0 \le b_{i,j}(t) \le A_{max}, \qquad \forall i \in \mathcal{Z}, j \in \mathcal{D} \tag{10}$$

We define b_i and $b_{i,j}$ as the time-average expectation number of $b_i(t)$ and $b_{i,j}(t)$, respectively. Then we have $b_i = \sum_{j \in \mathcal{D}} b_{i,j}, \forall i$.

For each individual TS, we update $Q_{i,j}(t)$ according to the service provisioning scheme in the hybrid cloud with

$$\begin{aligned} Q_{i,j}(t+1) = \max\{Q_{i,j}(t) - x_{i,j}(t) \cdot e_j - b_{i,j}(t), 0\} \\ + a_{i,j}(t), \qquad \forall i \in \mathcal{Z}, j \in \mathcal{D}. \end{aligned} \tag{11}$$

D. Profit-Driven Service Model

The MCSP's profit is the margin between revenue and cost. The revenue comes from serving the multimedia service requests, and hence it can be formulated as a function $f(\cdot)$ of the number of served requests. Here, we need to point out that the optimization techniques discussed in the following sections are applicable as long as $f(\cdot)$ is a concave and continuous function [9]. Hence, for simplification, we use the formulation in [30] and assume that $f(\cdot)$ is a linear one as

$$R = \sum_{i \in \mathcal{Z}} f(a_i - b_i) = \sum_{i \in \mathcal{Z}} \eta_i \cdot (a_i - b_i)$$

where η_i is the revenue per served request for zone i.

The cost includes three parts, i.e., the energy cost of private DCs, the payment for renting the resources from public DCs, and the communication cost for data transfers.[1] The energy consumption in a private DC in TS t is calculated as

$$P_j(t) = \varepsilon \cdot \{p_W \cdot y_j(t) + p_I \cdot [S_j - y_j(t)]\}, \qquad \forall j \in \mathcal{M}$$

[1]Note that in this work, we do not consider the CAPEX and other OPEX in the hybrid cloud.

where p_W and p_I are the energy consumption per server when it is in working and idle state for a TS, respectively, ε is the power usage efficiency (PUE) of each DC, which is defined as the ratio of its total power consumption to that of the servers [29]. Then, the energy cost of the private DCs is

$$C_m(t) = \sum_{j \in \mathcal{M}} \alpha_j \cdot P_j(t) \tag{12}$$

where α_j is the electricity price for DC j. And the time-average expectation energy cost (i.e., C_m) of the private DCs is $\sum_{j \in \mathcal{M}} \alpha_j \cdot \varepsilon \cdot [(p_W - p_I) \cdot y_j + p_I \cdot S_j]$..

As for the payment for renting the resources from the public DCs, we consider the time granularity for VM rentals (i.e., T_g) and get

$$C_n(t') = \sum_{j \in \mathcal{N}} \beta_j \cdot \sum_{k=1}^{S_j} y_{j,k}(t' \cdot T_g) \tag{13}$$

where β_j is the average rental cost per VM in public DC j for a time granularity T_g. The time-average expectation cost (i.e., C_n) of the public DCs is $\frac{1}{T_g} \sum_{j \in \mathcal{N}} \beta_j \cdot y_j$.

The communication cost is from transmitting the data of multimedia services from the end users to the hybrid cloud, which is calculated as

$$C_t(t) = \sum_{i \in \mathcal{Z}} \sum_{j \in \mathcal{D}} \gamma_{i,j} \cdot e_j \cdot x_{i,j}(t), \qquad \forall i \in \mathcal{Z} \tag{14}$$

where $\gamma_{i,j}$ is the average communication cost per request for the situation in which the requests are from zone i and get processed in DC j. The time-average expectation cost of the communication (i.e., C_t) is $\sum_{i \in \mathcal{Z}} \sum_{j \in \mathcal{D}} \gamma_{i,j} \cdot e_j \cdot x_{i,j}$.

To this end, we can see that the total time-average cost is the summation of the three parts

$$C = C_m + C_n + C_t \tag{15}$$

and in order to maximize the time-average profit of the MCSP, we need to solve the following optimization problem:

$$\begin{aligned} Maximize \quad & P = R - C, \\ s.t. \quad & \text{Eqs. } (1) - (10). \end{aligned} \tag{16}$$

Meanwhile, we should make sure that the lengths of all the queues $\{Q_{i,j}(t)\}$ would not increase towards infinite.

IV. Theoretical Analysis on Offline Optimization

We first analyze the optimization problem in (16) in an offline manner. Basically, at time t, we have a set of pending requests and need to obtain their service provisioning schemes to maximize the profit of the MCSP. More specifically, we need to determine the request routing $a_{i,j}$ and the server/VM allocation $x_{i,j}$ for each queue $Q_{i,j}(t)$, while the dropped requests $b_{i,j}$ and the state of each server/VM $y_{j,k}$ should also be obtained. We model the requests' arrivals with the Poisson process, and thus each queue $Q_{i,j}(t)$ becomes an M/D/1 queue. Then, the time-average latency is gotten as [31]

$$d_{i,j} = \frac{1}{2 \cdot e_j \cdot x_{i,j}} \cdot \frac{2e_j \cdot x_{i,j} - a_{i,j}}{e_j \cdot x_{i,j} - a_{i,j}}, \qquad \forall i, j \tag{17}$$

where $e_j \cdot x_{i,j}$ is the processing capacity of queue $Q_{i,j}(t)$. Here, we should limit the queueing delay less than d, and get

$$a_{i,j} \leq \frac{2 \cdot d \cdot e_j^2 \cdot x_{i,j}^2 - 2 \cdot e_j \cdot x_{i,j}}{2 \cdot d \cdot e_j \cdot x_{i,j} + 1}, \qquad \forall i, j. \tag{18}$$

Here, we define $\xi(t)$ as

$$\xi(t) = \sum_{i \in \mathcal{Z}} \eta_i \cdot (a_i(t) - b_i(t)) - \sum_{j \in \mathcal{D}} \mu_j \cdot y_j(t) - \sum_{i \in \mathcal{Z}} \sum_{j \in \mathcal{D}} \gamma_{i,j} \cdot e_j \cdot x_{i,j}(t)$$

where the formulation of μ_j is

$$\mu_j = \begin{cases} \alpha_j \cdot \varepsilon \cdot (p_W - p_I), & \forall j \in \mathcal{M} \\ \frac{\beta_j}{T_g}, & \forall j \in \mathcal{N}. \end{cases}$$

And let ξ denote the time-average expectation value of $\xi(t)$. Then we can transform the optimization problem in (16) into an offline optimization problem as follows.

$$\begin{aligned} Maximize \quad & \xi, \\ s.t. \quad & \text{Eq.(18)}, \\ & x_j \leq y_j \leq S_j, \qquad \forall j, \\ & \sum_j a_{i,j} \leq A_i, \qquad \forall i, \\ & 0 \leq b_{i,j} \leq A_{max}, \qquad \forall i, j. \end{aligned} \tag{19}$$

Note that the optimization in (19) gets the optimal result when and only when $y_j = x_j$ and $b_i = 0$. Hence, the problem can be reduced to

$$\begin{aligned} Maximize \quad & \sum_{i \in \mathcal{Z}} \sum_{j \in \mathcal{D}} \{\eta_i \cdot a_{i,j} - (\mu_j + \gamma_{i,j} \cdot e_j) \cdot x_{i,j}\}, \\ s.t. \quad & \text{Eq.(18)}, \\ & \sum_i x_{i,j} \leq S_j, \quad \forall j, \\ & \sum_j a_{i,j} \leq A_i, \quad \forall i. \end{aligned} \tag{20}$$

Finally, we can see that the optimal policy for the problem above is to allocate the requests one-by-one from the zone that provides the highest revenue to the DC that has the lowest cost, which is intuitive. Hence, we design the procedure in *Algorithm 1* for the offline optimization. Specifically, we first sort the zone-DC pairs in descending order of the profit and then allocate the requests sequentially. In *Line* 6, we keep allocating the requests in zone i to DC j until the rest of the servers/VMs in DC j can not guarantee that the requests' time-average latency is below d or all the requests have been allocated. When we get the optimal request routing $\{a_{i,j}\}$ and the server/VM allocation $\{x_{i,j}\}$, we determine the state of each server/VM according to them, as shown in *Lines* 12-13.

Algorithm 1 Offline Optimization Algorithm

Input: A_i and S_j.
1: $x'_j = 0, \quad \forall j$;
2: $a'_i = 0, \quad \forall i$;
3: sort the zone-DC pairs $\{(i,j), \forall i,j\}$ in descending order of $\eta_i - (\mu_j + \gamma_{i,j} \cdot e_j)$;
4: **for all** each zone-DC pair (i,j) in the sorted order **do**
5: **if** $\eta_i > \mu_j + \gamma_{i,j} \cdot e_j$ **then**
6: $a_{i,j} = \min\{A_i - a'_i, \frac{2 \cdot d \cdot e_j^2 \cdot (S_j - x'_j)^2 - 2 \cdot e_j \cdot (S_j - x'_j)}{2 \cdot d \cdot e_j \cdot (S_j - x'_j) + 1}\}$
7: $x_{i,j} = \frac{d \cdot a_{i,j} + 1 - \sqrt{d^2 \cdot a_{i,j}^2 + 1}}{2 \cdot d \cdot e_j}$;
8: $x'_j = x'_j + x_{i,j}$;
9: $a'_i = a'_i + a_{i,j}$;
10: **end if**
11: **end for**
12: $y_{j,k} = 1, \quad \forall j, \quad \forall k \in \{1, \cdots, x'_j\}$;
13: $y_{j,k} = 0, \quad \forall j, \quad \forall k \in \{x'_j + 1, \cdots, S_j\}$;

Note that the analysis above needs to know the pending requests in advance and can only guarantee the time-average latency of the served requests. In the following sections, we investigate the optimization problem in (16) further and design an online algorithm that does not require any pre-knowledge on the distribution of the requests' arrivals and can guarantee the worst-case latency for each served request.

V. Lyapunov Optimization and Online Scheduling Algorithm

In this section, we use the Lyapunov optimization technique to maximize the time-average profit of MCSP and propose an efficient online scheduling algorithm.

A. Lyapunov Optimization

Different from the offline optimization, the online optimization should determine all the variables for each TS t. Here, we redefine the problem and take all the related constraints into consideration

$$\begin{aligned} Maximize \quad & \xi, \\ s.t. \quad & \text{Eqs. } (1) - (10). \end{aligned} \tag{21}$$

In order to control the worst-case queueing delay in $Q_{i,j}(t)$, we apply the ϵ-persistent technique [9] and define a virtual queue $G_{i,j}(t)$ for $Q_{i,j}(t)$. Initially, we have $G_{i,j}(0) = 0$ and it is updated as seen in (22), at the bottom of the page. With all the queues in all the DCs for all the zones, we denote $\boldsymbol{Q}(t)$ as the queue matrix (i.e., $Q_{i,j}(t), \forall i,j$) for the hybrid cloud in TS t. Similarly, we can also get a queue matrix for $\{G_{i,j}(t), \forall i,j\}$ as $\boldsymbol{G}(t)$. Let $\boldsymbol{\Theta}(t) = [\boldsymbol{Q}(t), \boldsymbol{G}(t)]$, and then we have the Lyapunov function as

$$L(\boldsymbol{\Theta}(t)) = \frac{1}{2}\left\{\sum_{i \in \mathcal{Z}} \sum_{j \in \mathcal{D}} [Q_{i,j}^2(t) + G_{i,j}^2(t)]\right\} \tag{23}$$

and the Lyapunov drift function is $\Delta(\boldsymbol{\Theta}(t)) = \mathbb{E}\{L(\boldsymbol{\Theta}(t+1)) - L(\boldsymbol{\Theta}(t)) | \boldsymbol{\Theta}(t)\}$, and the Lyapunov drift-minus-profit function as

$$\Phi(\boldsymbol{\Theta}(t)) = \Delta(\boldsymbol{\Theta}(t)) - V \cdot \mathbb{E}\{\xi(t) | \boldsymbol{\Theta}(t)\}. \tag{24}$$

Here, V is the parameter to balance the queue steadiness and the profit. Finally, we transform the original optimization problem, which is to maximize the profit P in (16) and to limit the lengths of $\{Q_{i,j}(t), \forall i,j,t\}$ as well into the one below [9]:

$$Minimize \quad \Phi(\boldsymbol{\Theta}(t)) = \Delta(\boldsymbol{\Theta}(t)) - V \cdot \mathbb{E}\{\xi(t) | \boldsymbol{\Theta}(t)\}. \tag{25}$$

Also, we notice that $\Phi(\boldsymbol{\Theta}(t))$ satisfies the following inequality:

$$\Phi(\boldsymbol{\Theta}(t)) \leq B + \Phi_1(\boldsymbol{\Theta}(t)) + \Phi_2(\boldsymbol{\Theta}(t)) + \Phi_3(\boldsymbol{\Theta}(t)) + \Phi_4(\boldsymbol{\Theta}(t)). \tag{26}$$

Here, B is a constant number that satisfies the inequality below:

$$\begin{aligned} B \geq \frac{1}{2} \sum_{i,j} \mathbb{E}\Big\{ & a_{i,j}^2(t) + [x_{i,j}(t) \cdot e_j + b_{i,j}(t)]^2 \\ & + [\epsilon - x_{i,j}(t) \cdot e_j - b_{i,j}(t)]^2 | \boldsymbol{\Theta}(t) \Big\} \end{aligned} \tag{27}$$

and the expressions of $\Phi_1(\boldsymbol{\Theta}(t)) - \Phi_4(\boldsymbol{\Theta}(t))$ are as follows:

$$\Phi_1(\boldsymbol{\Theta}(t)) = \sum_{i,j} \mathbb{E}\{[Q_{i,j}(t) - V \cdot \eta_i] \cdot a_{i,j}(t) | \boldsymbol{\Theta}(t)\} \tag{28}$$

$$\Phi_2(\boldsymbol{\Theta}(t)) = \sum_{i,j} \mathbb{E}\{[V \cdot \eta_i - Q_{i,j}(t) - G_{i,j}(t)] \cdot b_{i,j}(t) | \boldsymbol{\Theta}(t)\} \tag{29}$$

$$\begin{aligned} \Phi_3(\boldsymbol{\Theta}(t)) = \sum_{i,j} \mathbb{E}\{ & [V \cdot \gamma_{i,j} - Q_{i,j}(t) - G_{i,j}(t)] \cdot e_j \\ & \times x_{i,j}(t) | \boldsymbol{\Theta}(t)\} \end{aligned} \tag{30}$$

$$\Phi_4(\boldsymbol{\Theta}(t)) = \sum_{j} \mathbb{E}\{V \cdot \mu_j \cdot y_j(t) | \boldsymbol{\Theta}(t)\}. \tag{31}$$

B. Distributed Online Scheduling Algorithm

Based on the formulations obtained with the Lyapunov optimization technique, we propose an online scheduling algorithm that can work in the distributed manner to handle both the request scheduling in the proxies and the resource management for the servers/VMs in the DCs.

1) Request Scheduling in Zone Proxies: In each TS t, the proxy in zone i should decide the number of admitted requests (i.e., $a_i(t)$) and which DC in the hybrid cloud to route each of

$$G_{i,j}(t+1) = \begin{cases} \max(G_{i,j}(t) - x_{i,j}(t) \cdot e_j - b_{i,j}(t) + \varepsilon, 0), & \text{if } Q_{i,j}(t) > x_{i,j}(t) \cdot e_j + b_{i,j}(t) \\ 0, & \text{otherwise} \end{cases} \tag{22}$$

them to (i.e., $a_{i,j}(t)$). As (28) shows that $\Phi_1(\mathbf{\Theta}(t))$ is related to $a_{i,j}(t)$, the proxies can obtain $\{a_{i,j}(t)\}$ and $\{a_i(t)\}$ by minimizing $\Phi_1(\mathbf{\Theta}(t))$. We assume that the zones are isolated and not overlapping with each other, and hence for any two different zones i_1 and i_2, their decision variables $a_{i_1,j}(t)$ and $a_{i_2,j}(t)$ are independent. Then, we solve a set of independent optimization sub-problems, each of which is for a zone, to obtain $\{a_{i,j}(t)\}$ and $\{a_i(t)\}$. Since the zones are handled independently, the request scheduling is conducted distributedly. For zone i, the optimization sub-problem is

$$\begin{aligned} Minimize \quad & \left(\sum_{j\in\mathcal{D}} Q_{i,j}(t)\cdot a_{i,j}(t)\right) - V\cdot\eta_i\cdot a_i(t), \\ s.t. \quad & \text{Eqs. } (1)-(2). \end{aligned} \tag{32}$$

The optimization in (32) can be solved as follows. First of all, we relax it by assuming that the optimal value of $a_i(t)$ is known as $a_i^*(t)$, and transform it into

$$\begin{aligned} Minimize \quad & \sum_{j\in\mathcal{D}} Q_{i,j}(t)\cdot a_{i,j}(t), \\ s.t. \quad & \text{Eqs. } (1)-(2). \end{aligned}$$

Hence, the optimal values of $\{a_{i,j}(t), \forall j\in\mathcal{D}\}$ will be

$$a_{i,j}^*(t) = \begin{cases} a_i^*(t), & j^* = \arg\min\{Q_{i,j}(t), \forall j\in\mathcal{D}\} \\ 0, & \text{otherwise.} \end{cases} \tag{33}$$

The solution in (33) means that in zone i, the proxy should route all the admitted requests to the DC j^*, whose queue $Q_{i,j^*}(t)$ has the shortest length. Then, we return to the original optimization in (32) and transform it into

$$\begin{aligned} Minimize \quad & [Q_{i,j^*}(t) - V\cdot\eta_i]\cdot a_i(t) \\ s.t. \quad & 0\le a_i(t)\le A_i(t). \end{aligned}$$

And we can obtain the optimal value of $a_i(t)$ as

$$a_i^*(t) = \begin{cases} 0, & Q_{i,j^*}(t) > V\cdot\eta_i \\ A_i(t), & \text{otherwise} \end{cases} \tag{34}$$

which means that if the shortest queue $Q_{i,j^*}(t)$ is longer than $V\cdot\eta_i$, the proxy in zone i should drop all the incoming requests in TS t, otherwise, all the requests will be admitted and routed to DC j^*. Hence, (34)–(33) determine the request scheduling in the zone proxies.

2) Request Dropping in Hybrid Cloud: In each TS t, DC j should determine the numbers of buffered requests to drop $\{b_{i,j}(t), \forall i\}$. With (29), we can see that $\Phi_2(\mathbf{\Theta}(t))$ is related to $b_{i,j}(t)$. Hence, the DCs can obtain $\{b_{i,j}(t)\}$ by minimizing $\Phi_2(\mathbf{\Theta}(t))$. Apparently, for any two different DCs j_1 and j_2, their decision variables $b_{i,j_1}(t)$ and $b_{i,j_2}(t)$ are independent. Then, we solve a set of independent optimization sub-problems, each of which is for a DC, to obtain $\{b_{i,j}(t)\}$. For DC j, the optimization sub-problem is

$$\begin{aligned} Minimize \quad & [V\cdot\eta_i - Q_{i,j}(t) - G_{i,j}(t)]\cdot b_{i,j}(t), \\ s.t. \quad & 0\le b_{i,j}(t)\le A_{max}. \end{aligned} \tag{35}$$

Then, we get the optimal values of $b_{i,j}(t)$ as

$$b_{i,j}^*(t) = \begin{cases} A_{max}, & Q_{i,j}(t) + G_{i,j}(t) \ge V\cdot\eta_i \\ 0, & \text{otherwise} \end{cases} \tag{36}$$

which means that if DC j observes $Q_{i,j}(t) + G_{i,j}(t) \ge V\cdot\eta_i$, it should drop A_{max} buffered requests in $Q_{i,j}(t)$, otherwise, no request will be dropped.

3) Server Management in Private Cloud: For DC j in the private cloud (i.e., $j\in\mathcal{M}$), it needs to decide the server allocation variable $x_{i,j}(t)$ for queue $Q_{i,j}(t)$. Meanwhile, after allocating the servers, the DC should update the states of the servers (i.e., the decision variables $\{y_{j,k}(t), \forall k\}$). Note that $\Phi_3(\mathbf{\Theta}(t))$ and $\Phi_4(\mathbf{\Theta}(t))$ are related to $x_{i,j}(t)$, $x_j(t)$ and $y_{j,k}(t)$. Then, with (30)–(31), we obtain the optimization sub-problem for each private DC j as follows:

$$\begin{aligned} Minimize \quad & \left(\sum_{i\in\mathcal{Z}} [V\cdot\gamma_{i,j} - Q_{i,j}(t) - G_{i,j}(t)]\cdot e_j\cdot x_{i,j}(t)\right) \\ & + V\cdot\mu_j\cdot y_j(t), \\ s.t. \quad & \text{Eqs. } (4)-(5), (7)-(9). \end{aligned} \tag{37}$$

Due to the constraint that $0\le x_j(t)\le y_j(t)$, (37) can be minimized when and only when $y_j(t) = x_j(t)$. Therefore, the optimization can be reduced to

$$\begin{aligned} Minimzie \quad & \left(\sum_{i\in\mathcal{Z}} [V\cdot\gamma_{i,j} - Q_{i,j}(t) - G_{i,j}(t)]\cdot e_j\cdot x_{i,j}(t)\right) \\ & + V\cdot\mu_j\cdot x_j(t), \\ s.t. \quad & \text{Eqs. } (4)-(5), (7)-(8). \end{aligned} \tag{38}$$

We design an algorithm to solve the optimization problem in (38). *Algorithm* 2 shows the detailed procedure. *Lines* 1-2 are for initialization. Basically, we sort all the queues $\{Q_{i,j}(t), \forall i\}$ in DC j in descending order of $Q_{i,j}(t) + G_{i,j}(t) - V\cdot\gamma_{i,j}$. Then, in the sorted order, we allocate servers to each queue as shown in *Lines* 3-9. Specifically, we allocate at most $\lceil\frac{Q_{i,j}(t)}{e_j}\rceil$ servers to each queue until all the S_j servers have been allocated or we encounter $[Q_{i,j}(t) + G_{i,j}(t) - V\cdot\gamma_{i,j}]\cdot e_j < V\cdot\mu_j$. Then, *Lines* 10-12 update the states of the servers. As mentioned above, in the optimal case, we have $y_j(t) = x_j(t)$ (as shown in *Line* 10).

Algorithm 2 Server Management in Private DC j

Input:
$Q_{i,j}(t)$ and $G_{i,j}(t)$.
1: $x_j(t) = 0, x_{i,j}(t) = 0, \forall i$;
2: sort the queues $\{Q_{i,j}(t), \forall i\}$ in descending order of $Q_{i,j}(t) + G_{i,j}(t) - V\cdot\gamma_{i,j}$;
3: **for** each $Q_{i,j}(t)$ in the sorted order **do**
4: **if** $[Q_{i,j}(t) + G_{i,j}(t) - V\cdot\gamma_{i,j}]\cdot e_j \ge V\cdot\mu_j$ **then**
5: $x_{i,j}(t) = \min\left\{\frac{Q_{i,j}(t)}{e_j}, S_j - x_j(t)\right\}$;
6: **end if**
7: allocate $x_{i,j}(t)$ servers to queue $Q_{i,j}(t)$;
8: $x_j(t) = x_j(t) + x_{i,j}(t)$;
9: **end for**
10: $y_j(t) = x_j(t)$;
11: $y_{j,k} = 1, \forall k\in\{1,\cdots,y_j(t)\}$;
12: $y_{j,k} = 0, \forall k\in\{y_j(t)+1,\cdots,S_j\}$;

4) VM Management in Public Cloud: For DC j in the public cloud (i.e., $j \in \mathcal{N}$), it also needs to decide the VM allocation variable $x_{i,j}(t)$ for queue $Q_{i,j}(t)$ and the VM state variable $y_{j,k}(t)$ for each VM. Similarly, with (30)–(31), we obtain the optimization sub-problem for each public DC j as follows.

$$\begin{aligned} Minimize \quad & \left(\sum_{i\in\mathcal{Z}}[V\cdot\gamma_{i,j}-Q_{i,j}(t)-G_{i,j}(t)]\cdot e_j\cdot x_{i,j}(t)\right) \\ & +V\cdot\mu_j\cdot y_j(t), \\ s.t. \quad & \text{Eqs. } (4)-(9). \end{aligned} \tag{39}$$

With (4), we can transform (6) into

$$y_j(t)\geq\sum_{k=1}^{S_j}\lceil\frac{Y_{j,k}(t)}{T_g}\rceil. \tag{40}$$

Then, in order to minimize (39) under the constraints in (9) and (40), we get

$$y_j(t)=\max\left\{x_j(t),\sum_{k=1}^{S_j}\lceil\frac{Y_{j,k}(t)}{T_g}\rceil\right\}. \tag{41}$$

The aforementioned optimization can be solved with *Algorithm* 3. *Lines* 1-3 are for the initialization, in which *Lines* 2 updates the auxiliary parameters $\{Y_{j,k}(t),\forall k\}$ to determine the states of the VMs. Then, similar to those in *Algorithm* 2, *Lines* 4-11 allocate VMs to each queue. We handle the situation that the MCSP needs to rent more VMs from the public cloud in *Lines* 12-20.

Algorithm 3 VM Management in Public DC j

Input:
$Q_{i,j}(t)$, $G_{i,j}(t)$ and $Y_{j,k}(t-1)$.
1: $x_j(t)=0, x_{i,j}(t)=0,\forall i$;
2: $Y_{j,k}(t)=\max\{Y_{j,k}(t-1)-1,0\},\forall k$;
3: $y_{j,k}(t)=\lceil\frac{Y_{j,k}(t)}{T_g}\rceil,\forall k$;
4: sort the queues $\{Q_{i,j}(t),\forall i\}$ in descending order of $Q_{i,j}(t)+G_{i,j}(t)-V\cdot\gamma_{i,j}$;
5: **for** each $Q_{i,j}(t)$ in the sorted order **do**
6: **if** $[Q_{i,j}(t)+G_{i,j}(t)-V\cdot\gamma_{i,j}]\cdot e_j\geq V\cdot\mu_j$ **then**
7: $x_{i,j}(t)=\min\{\lceil\frac{Q_{i,j}(t)}{e_j}\rceil, S_j-x_j(t)\}$;
8: **end if**
9: allocate $x_{i,j}(t)$ servers to queue $Q_{i,j}(t)$;
10: $x_j(t)=x_j(t)+x_{i,j}(t)$;
11: **end for**
12: $y_j(t)=\max\{x_j(t),\sum_{k=1}^{S_j}\lceil\frac{Y_{j,k}(t)}{T_g}\rceil\}$;
13: $\Delta=y_j(t)-\sum_{k=1}^{S_j}\lceil\frac{Y_{j,k}(t)}{T_g}\rceil, k=0$;
14: **while** $\Delta>0$ **do**
15: $k=k+1$;
16: **if** $y_{j,k}(t)=0$ **then**
17: $y_{i,k}(t)=1, Y_{j,k}(t)=T_g$;
18: $\Delta=\Delta-1$;
19: **end if**
20: **end while**

5) Distributed Online Hybrid Cloud Management: Based on the discussion above, we can get the optimal solution for profit-driven request scheduling and resource management in the hybrid cloud. Then, we propose *Algorithm* 4 as the overall procedure for the distributed online scheduling and resource management. The algorithm covers the operations in both the proxies and DCs. *Lines* 1-5 are for the proxies, where they determine the schemes for admitting requests and forwarding them to the DCs. *Lines* 6-17 are for the DCs.[2] We drop a proper number of requests in each zone as shown in *Line* 14. Note that this operation would not change the queue-updating strategy described in (11).

Algorithm 4 Distributed Online Hybrid Cloud Management

1: **for** each zone $i\in\mathcal{Z}$ **do**
2: proxy receives information on $\{Q_{i,j}(t),\forall j\}$ from DCs;
3: proxy obtains $\{a_{i,j}(t),\forall j\}$ with (33)–(34);
4: proxy sends requests to DCs according to $\{a_{i,j}(t)\}$;
5: **end for**
6: **for** each DC $j\in\mathcal{D}$ **do**
7: send information on $\{Q_{i,j}(t),\forall i\}$ to proxy in zone i;
8: **if** DC j is a private DC **then**
9: manage servers with *Algorithm* 2;
10: **else**
11: manage VMs with *Algorithm* 3;
12: **end if**
13: calculate $\{b_{i,j}(t),\forall i\}$ with (36);
14: drop $\min\{\max\{Q_{i,j}(t)-x_{i,j}(t)\cdot e_j,0\},b_{i,j}(t)\}$ requests in each queue $Q_{i,j}(t),\forall i$;
15: update $\{Q_{i,j}(t)\}$ with (11);
16: update $\{G_{i,j}(t)\}$ with (22);
17: **end for**

C. Performance Analysis

1) Queueing Delays: The maximum queue lengths of $G_{i,j}(t)$ and $Q_{i,j}(t)$ are as follows. By combining (22) and (36), we can get the maximum length of $G_{i,j}(t)$ as in (42). The maximum length of $Q_{i,j}(t)$ can be obtained with (11) and (36), as shown in (43).

$$G_{i,j}^{max}=V\cdot\eta_i+\epsilon,\quad \forall i\in\mathcal{Z},j\in\mathcal{D}, \tag{42}$$

$$\begin{aligned} Q_{i,j}^{max}=&V\cdot\min\left(\eta_i,\frac{\gamma_{i,j}\cdot e_j+\mu_j}{e_j}\right)+A_{max}, \\ & \forall i\in\mathcal{Z},j\in\mathcal{D}. \end{aligned} \tag{43}$$

To satisfy the constraint on the worst-case latency, i.e., each request should be processed within duration d, we need to ensure that $G_{i,j}^{max}$ and $Q_{i,j}^{max}$ satisfy the inequality below [9]:

$$\frac{Q_{i,j}^{max}+G_{i,j}^{max}}{\epsilon}\leq d,\quad \forall i\in\mathcal{Z},j\in\mathcal{D} \tag{44}$$

which means that the value of ϵ is bounded as

$$\epsilon\geq\max\left(\frac{Q_{i,j}^{max}+G_{i,j}^{max}}{d}\right),\quad \forall i,j. \tag{45}$$

[2]Note that even though we organize the operations in the proxies and DCs as *Lines* 1-5 and *Lines* 6-17 in *Algorithm* 5.2.5, they should be run in parallel and distributedly on the proxies and DCs in the hybrid cloud system.

2) Optimality Analysis:

Proposition 1: The time-average profit obtained by Algorithm 4 based on Lyapunov optimization can approach to the theoretical maximum ξ^* within a constant gap and it satisfies

$$\xi \geq \xi^* - \frac{B}{V}. \tag{46}$$

Proof: According to [9], there exists a feasible solution $\{a^f_{i,j}(t), b^f_{i,j}(t), x^f_{i,j}(t), y^f_j(t), \forall i,j,t\}$ to the optimization problem in (21), which satisfies that $\mathbb{E}\{a^f_{i,j}(t)\} = a^*_{i,j}, \mathbb{E}\{b^f_{i,j}(t)\} = b^*_{i,j}, \mathbb{E}\{x^f_{i,j}(t)\} = x^*_{i,j}, \mathbb{E}\{y^f_j(t)\} = y^*_j$, where $\{a^*_{i,j}, b^*_{i,j}, x^*_{i,j}, y^*_j, \forall i,j\}$ are from the optimum solution. As the feasible solution should satisfy (26), we get

$$\begin{aligned}\Phi(\mathbf{\Theta}(t)) = \Delta(\mathbf{\Theta}(t)) - V\cdot\mathbb{E}\{\xi(t)|\mathbf{\Theta}(t)\} \leq B \\ + \sum_{i,j}\mathbb{E}\{(Q_{i,j}(t) - V\cdot\eta_i)\cdot a^f_{i,j}(t)|\mathbf{\Theta}(t)\} \\ + \sum_{i,j}\mathbb{E}\{(V\cdot\eta_i - Q_{i,j}(t) - G_{i,j}(t))\cdot b^f_{i,j}(t)|\mathbf{\Theta}(t)\} \\ + \sum_{i,j}\mathbb{E}\{(V\cdot\gamma_{i,j} - Q_{i,j}(t) - G_{i,j}(t))\cdot e_j\cdot x^f_{i,j}(t)|\mathbf{\Theta}(t)\} \\ + \sum_{j}\mathbb{E}\{V\cdot\mu_j\cdot y^f_j(t)|\mathbf{\Theta}(t)\}. \end{aligned} \tag{47}$$

Since the feasible solution is independent of $\mathbf{\Theta}(t)$, we can take expectations of (47) and simplify it as

$$\begin{aligned}\mathbb{E}\{L(\mathbf{\Theta}(t+1)) - L(\mathbf{\Theta}(t))\} - V\cdot\mathbb{E}\{\xi(t)\} \leq B - V\cdot\xi^* \\ + \sum_{i,j}\mathbb{E}\{Q_{i,j}(t)\cdot(a^*_{i,j} - b^*_{i,j} - e_j\cdot x^*_{i,j})\} \\ - \sum_{i,j}\mathbb{E}\{G_{i,j}(t)\cdot(b^*_{i,j} + e_j\cdot x^*_{i,j})\}.\end{aligned}$$

Note that we update $Q_{i,j}(t)$ according to (11) and all the queues keep stable, which means $a^*_{i,j} \leq b^*_{i,j} + e_j\cdot x^*_{i,j}$. Therefore, we have

$$\mathbb{E}\{L(\mathbf{\Theta}(t+1)) - L(\mathbf{\Theta}(t))\} - V\cdot\mathbb{E}\{\xi(t)\} \leq B - V\cdot\xi^*. \tag{48}$$

Then, we summarize both sides of (48) for all the TS' (i.e., $t \in [1,T]$) and get

$$\frac{1}{T}\cdot\left\{\mathbb{E}\{L(\mathbf{\Theta}(t+1))\} - \mathbb{E}\{L(\mathbf{\Theta}(0))\} - \sum_{t=1}^{T} V\cdot\mathbb{E}\{\xi(t)\}\right\} \leq B - V\cdot\xi^*.$$

Note that $L(\mathbf{\Theta}(t+1)) \geq 0$, $L(\mathbf{\Theta}(0)) = 0$ and the definition of ξ, we can prove

$$\xi \geq \xi^* - \frac{B}{V}.$$

■

VI. Performance Evaluation

In this section, we evaluate the performance of the proposed algorithm with simulations that use both synthetical and real-data traces. Each simulation uses the TS as $\Delta t = 10$ seconds and runs for 8,640 TS' (i.e., 24 hours) to obtain sufficient statistical accuracy. The VM rental in the public cloud has the time granularity as 1 hour, i.e., $T_g = 360$ TS'. We assume that the hybrid cloud consists of 4 private DCs and 6 public DCs and provides multimedia services to 20 zones. We place 200 servers in each private DC and assume that each server can process two service requests in a TS. In every public DC, we assume that the MCSP can rent at most 500 VMs and each VM can process one request in a TS. In the private DCs, we assume that the power consumption of a server is 200 W and 100 W when it is in the working and idling states, respectively. The energy cost is calculated with a constant price model, similar to that in [32]. Here, we consider that all the VMs are the on-demand m3.medium instances of Amazon EC2, each of which costs \$0.07 /hour.[3] And the communication cost $\gamma_{i,j}$ is randomly distributed within $[\$10^{-5}, \$10^{-4}]$ for each request. The revenue from serving a request is \$0.001, according to [32].

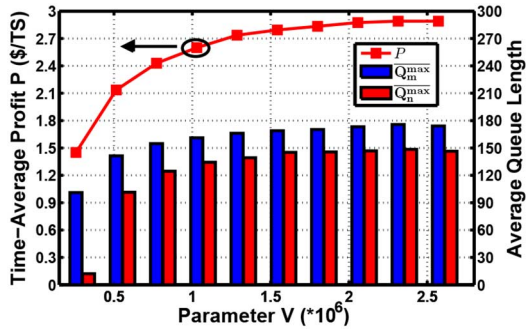

Fig. 3. Impacts of parameter V (using synthetical traces).

The simulations investigate the impacts of adjustable parameter V and worst-case latency d on the time-average profit and queue lengths. Here, we use $Q^{max}_{i,j}$ and $\overline{Q_{i,j}}$ to denote the maximum and average queue length of $Q_{i,j}(t)$, respectively. Then, for the hybrid cloud, Q^{max}_m (i.e., $\max\{Q^{max}_{i,j}, \forall i,j \in \mathcal{M}$) and Q^{max}_n (i.e., $\max\{Q^{max}_{i,j}, \forall i,j \in \mathcal{N}$) represent the maximum queue lengths in the private and public DCs, respectively. Similarly, $\overline{Q^{max}_m}$ (i.e., $\max\{\overline{Q_{i,j}}, \forall i,j \in \mathcal{M}$) and $\overline{Q^{max}_n}$ (i.e., $\max\{\overline{Q_{i,j}}, \forall i,j \in \mathcal{N}$) record the maximums of average queue lengths.

A. Simulations With Synthetical Traces

We first evaluate the proposed algorithm with simulations that use synthetical traces. Specifically, the number of requests that come to each zone proxy, i.e., $A_i(t)$, follows a uniform distribution with the time-average arrival rate as $\frac{1}{2}\cdot A_{max}$. Note that $A_i(t)$ can follow any distribution, as long as its time-average value stays the same.

1) Impact of Adjustable Parameter V : We use $A_{max} = 400$ requests/TS, set the worst-case latency as $d = 10$ TS' and run simulations with different values of V. Fig. 3 shows how the time-average profit P changes with V in the simulations. The results indicate that we can obtain an increased P with a larger V. Moreover, P increases faster with V when V is smaller, and the trend slows down for a relatively large V, which makes

[3][Online]. Available: http://aws.amazon.com/ec2/pricing/

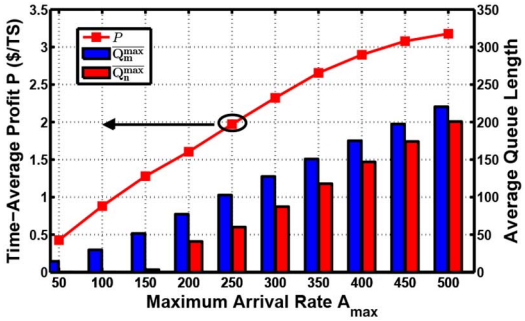

Fig. 4. Impacts of request arrival rate (using synthetical traces).

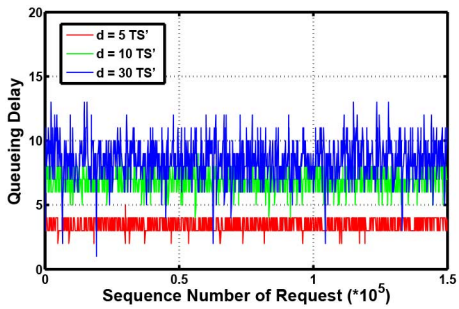

Fig. 5. Queueing delay of requests in $Q_{1,1}(t)$ (using synthetical traces).

P converge to its maximum value eventually. This observation verifies that by using the Lyapunov optimization technique, the proposed algorithm makes the time-average profit approach to its maximum value within the $O(1/V)$ gap.

Fig. 3 also shows the results on queue lengths (i.e., $\overline{Q_m^{max}}$ and $\overline{Q_n^{max}}$). We can see that the maximums of the average queue lengths increase with V too. This means that we have a tradeoff between the profit and the delay, which can be adjusted with V. Moreover, we observe that $\overline{Q_m^{max}}$ is larger than $\overline{Q_n^{max}}$ when V is the same. This is because the cost for processing a request in the private cloud is less than that in the public cloud. Hence, as long as the private cloud is still available, we always allocate the requests to it first.

2) Impact of Request Arrival Rate: The simulations also use different maximum request arrival rate A_{max} to study its impact on system performance. We use $\epsilon = A_{max}$ to get the value of V that provides the maximum time-average profit. Fig. 4 plots the results on P, $\overline{Q_m^{max}}$ and $\overline{Q_n^{max}}$ when A_{max} changes. We can see that due to service saturation, the time-average profit P increases more slowly when the maximum request arrival rate A_{max} increases. Meanwhile, $\overline{Q_m^{max}}$ and $\overline{Q_n^{max}}$ increase with A_{max} approximately linearly. It is interesting to notice that $\overline{Q_n^{max}}$ approaches to 0 when $A_{max} \leq 150$ requests/TS, which means that the public DCs are barely used in such light-load situations. This observation confirms that our algorithm can tailor the resource reservation in the hybrid cloud intelligently and dynamically.

3) Impact of Worst-Case Delay: To evaluate the impact of the worst-case latency d, we fix $A_{max} = 400$ requests/TS and $\epsilon = A_{max}$ and obtain the value of V for reaching the maximum time-average profit for different d. We first verify that all the provisioned requests are completed within d. Fig. 5 shows the queueing delay of requests in queue $Q_{1,1}(t)$ for different d. It can be seen that the actual queueing delays are all shorter than d. Table I shows the impact of d on time-average profit P and queue lengths. We observe that P increases with d, which is because a larger d makes the delay constraint looser and hence less requests would be dropped. We also notice that the growth speed of P reduces gradually with d, especially when $d > 15$ TS'. This is due to the fact that when d is sufficiently long, we can successfully serve most of the incoming requests, and thus P would not increase with d significantly any more. Both Q_m^{max} and Q_n^{max} are very close to the value of A_{max} (i.e., 400), when $d \leq 15$ TS'. This observation verifies (42)–(43). As the number of dropped requests becomes less for a longer d, Q_m^{max} and Q_n^{max} increase with d afterwards. $\overline{Q_m^{max}}$ and $\overline{Q_n^{max}}$ also increase with d, and $\overline{Q_m^{max}}$ is larger than $\overline{Q_n^{max}}$ because the profit-driven scheme prefers to use the private cloud.

TABLE I
IMPACT OF WORST-CASE LATENCY d (USING SYNTHETICAL TRACES)

d (TS')	5	10	15	20	25	30
P ($/TS)	1.76	2.91	3.17	3.18	3.18	3.19
Q_m^{max}	400	400	469	475	495	491
Q_n^{max}	400	400	448	468	481	509
$\overline{Q_m^{max}}$	146.99	177.67	178.09	179.11	180.46	179.57
$\overline{Q_n^{max}}$	86.54	149.61	151.64	152.97	157.83	158.52

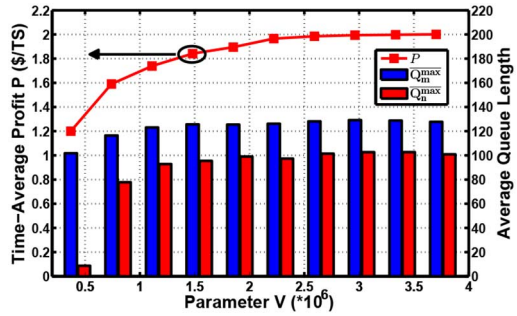

Fig. 6. Real trace of dynamic requests.

B. Simulations With Real Traces

To evaluate the performance of the proposed algorithm in more practical environments, we design the simulations to use a real trace of YouTube requests in a campus network[4,5] [33]. Fig. 6 shows the number of requests per TS from the real trace, and the maximum request arrival rate is $A_{max} = 36$ requests/TS. As the load is too low when comparing to the capacity of the hybrid cloud, we introduce a scaling factor θ.

We first perform simulations with different V and have $d = 10$ TS' and $\theta = 20$ (i.e., $A_{max} = 720$ requests/TS). As shown in Fig. 7, P still increases with V and the growth speed is slower for a larger V. These results verify that the proposed algorithm can still approach to the theoretical optimal value of P within $O(1/V)$ gap in the dynamic environment with the real trace.

[4]Note that we just use the trace to emulate the request arrivals in a practical situation. Therefore, even though the trace was for content delivery originally, we use it to simulate the multimedia cloud computing as discussed above.

[5]"YouTube traces from the campus network," [Online]. Available: http://traces.cs.umass.edu/index.php/Network/Network

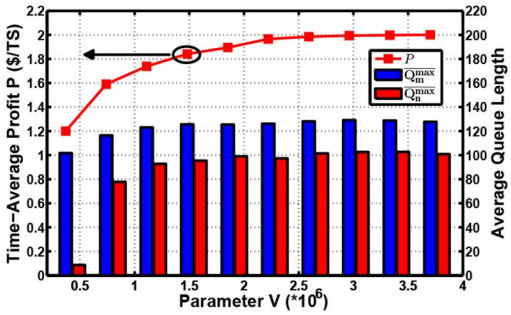

Fig. 7. Impacts of parameter V (using real traces).

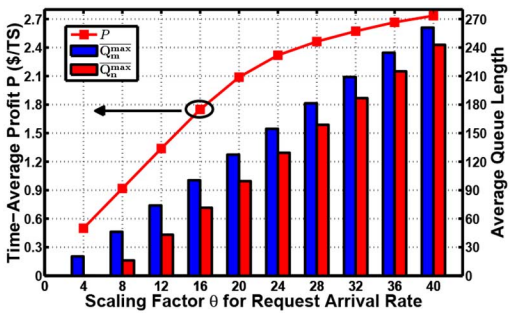

Fig. 8. Impacts of request arrival rate (using real traces).

TABLE II
IMPACT OF WORST-CASE LATENCY d (USING REAL TRACES)

d (TS')	5	10	15	20	25	30
P ($/TS)	1.46	2.09	2.18	2.18	2.19	2.20
Q_m^{max}	720	720	820	920	1000	1120
Q_n^{max}	720	720	820	900	960	1120
$\overline{Q_m^{max}}$	114.98	127.96	134.98	144.48	153.01	162.43
$\overline{Q_n^{max}}$	85.67	100.17	110.42	122.09	139.31	145.49

The figure also indicates that $\overline{Q_m^{max}}$ and $\overline{Q_n^{max}}$ change with V following the similar trend as that in Fig. 3.

We then run simulations with different θ to investigate the impacts of A_{max} on the algorithm's performance. Again, we use $\epsilon = A_{max}$ to get the value of V that provides the maximum time-average profit. Fig. 8 illustrates that the growth speed of P decreases with θ, while $\overline{Q_m^{max}}$ and $\overline{Q_n^{max}}$ increases with θ approximately linearly. Finally, we evaluate the impacts of d with $\theta = 20$. The results in Table II show similar trends as those in Table I. Fig. 9 plots the actual queueing delay of the requests in queue $Q_{1,1}(t)$, which shows that all the provisioned requests are completed within d. Here, we notice that the actual queueing delay exhibits a larger volatility when d is longer. Specifically, when the request arrival rate is relatively high (e.g., $t \in [3000, 4000]$ TS'), the requests experience significantly longer queueing delays when $d = 30$ TS'. This is because a lot of requests will be buffered in the queue under such condition.

We also use the T-slot lookahead algorithm in [9], [34] as the benchmark and compare our proposed algorithm with it. Basically, the T-slot lookahead algorithm is a quasi-online scheme as it needs to know future request arrivals in advance. Here, we denote the lookahead period as T_l TS'. Then, the T-slot

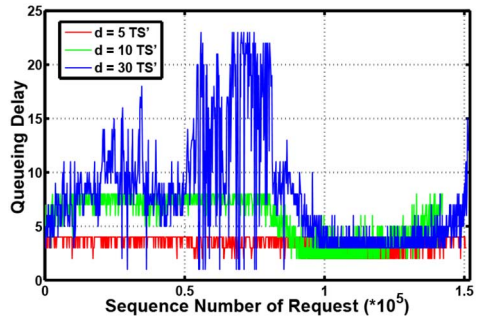

Fig. 9. Queueing delay of each served request in $Q_{1,1}(t)$ (using real traces).

TABLE III
COMPARISON WITH T-SLOT LOOKAHEAD ALGORITHM (USING REAL TRACES)

		$d = 5$ $T_l = 10$	$d = 5$ $T_l = 20$	$d = 10$ $T_l = 10$	$d = 10$ $T_l = 20$
P ($/TS)	Ours	1.15	1.15	1.40	1.40
	T-slot	1.24	1.29	1.59	1.71
Computation Time (msec)	Ours	4.3	4.3	4.5	4.5
	T-slot	461.7	495.8	488.1	510.9

lookahead algorithm can optimize the request scheduling and resource management operations within the T_l TS' jointly. We set $V = 10^5$ and get the minimum ϵ using (42)–(45). The simulation environment is Matlab 2012b running on a personal computer with 3.10 GHz Intel Core i3-2100 CPU and 4.00 GB RAM. Table III shows the results on the profit and computation time with the real trace, using different worst-case delays and lookahead periods.[6] It can be seen that the profit from our algorithm is close to that from the T-slot lookahead algorithm, while the computation time of our algorithm is two-magnitude shorter than that of the T-slot lookahead algorithm. These results verify that our proposed algorithm can operate in a truly online manner and obtain reasonably good solutions on the request scheduling and resource management without any pre-knowledge on the request arrivals.

VII. CONCLUSIONS

In this work, we investigated how to maximize an MCSP's profit from provisioning multimedia services to geo-distributed users with hybrid cloud. We first designed a service provisioning model to manage the resources in hybrid cloud. Then, we leveraged the Lyapunov optimization technique to maximize the profit of MCSP and proposed an online algorithm that can manage the hybrid cloud in the distributed manner. Also, we applied the ϵ-persistent technique to ensure that the worst-case latency of the provisioned multimedia requests would be bounded. Finally, we evaluated our proposal with extensive simulations using both synthetical and real traces. Simulation results indicated that the proposed algorithm can manage the hybrid cloud efficiently and maximize the profit of MCSP.

[6]Here, "ours" and "T-slot" are the abbreviations for our online algorithm and the T-slot lookahead algorithm, respectively. The computation time is the average time used for optimizing the request scheduling and resource management within one TS.

REFERENCES

[1] W. Zhu, C. Luo, J. Wang, and S. Li, "Multimedia cloud computing," *IEEE Signal. Process. Mag.*, vol. 28, pp. 59–69, May 2011.

[2] I. Bisio, R. Pan, F. Lavagetto, M. Marchese, A. Sciarrone, C. Fra, and M. Valla, "Smartphone-based automatic place recognition with Wi-Fi signals for location-aware services," in *Proc. ICC*, Jun. 2012, pp. 4943–4948.

[3] I. Bisio, F. Lavagetto, M. Marchese, and A. Sciarrone, "Comparison of situation awareness algorithms for remote health monitoring with smartphones," in *Proc. GLOBECOM*, Dec. 2014, pp. 2454–2459.

[4] Y. Wen, X. Zhu, J. Rodrigues, and C. Chen, "Cloud mobile media: Reflections and outlook," *IEEE Trans. Multimedia*, vol. 16, no. 4, pp. 885–902, Jun. 2014.

[5] M. Altamimi, R. Palit, K. Naik, and A. Nayak, "Energy-as-a-Service (EaaS): On the efficacy of multimedia cloud computing to save smartphone energy," in *Proc. CLOUD*, Jun. 2012, pp. 764–771.

[6] I. Jung, J. Insoon, Y. Yu, E. Hyeonsang, and H. Yeom, "Enhancing QoS and energy efficiency of realtime network application on smartphone using cloud computing," in *Proc. APSCC*, Dec. 2011, pp. 203–207.

[7] A. Greenberg, J. Hamilton, D. Maltz, and P. Patel, "The cost of a cloud: Research problems in data center networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 39, pp. 68–73, Jan. 2009.

[8] M. Hajjat, X. Sun, Y. Sung, D. Maltz, S. Rao, K. Sripanidkulchai, and M. Tawarmalani, "Cloudward bound: Planning for beneficial migration of enterprise applications to the cloud," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 40, pp. 243–254, Aug. 2010.

[9] M. Neely, *Stochastic Network Optimization With Application to Communication and Queueing Systems*. San Rafael, CA, USA: Morgan and Claypool, 2010.

[10] X. Nan, Y. He, and L. Guan, "Optimal resource allocation for multimedia cloud in priority service scheme," in *Proc. ISCAS*, May 2012, pp. 1111–1114.

[11] X. Nan, Y. He, and L. Guan, "Optimization of workload scheduling for multimedia cloud computing," in *Proc. ISCAS*, May 2013, pp. 2872–2875.

[12] X. Nan, Y. He, and L. Guan, "Towards optimal resource allocation for differentiated multimedia services in cloud computing environment," in *Proc. ICASSP*, May 2014, pp. 684–688.

[13] Y. Wu, C. Wu, B. Li, X. Qiu, and F. Lau, "CloudMedia: When cloud on demand meets video on demand," in *Proc. ICDCS*, Jun. 2011, pp. 268–277.

[14] G. Gao, W. Zhang, Y. Wen, Z. Wang, W. Zhu, and Y. Tan, "Cost optimal video transcoding in media cloud: Insights from user viewing pattern," in *Proc. ICME*, Jul. 2014, pp. 1–6.

[15] Y. Wu, C. Wu, B. Li, L. Zhang, Z. Li, and F. Lau, "Scaling social media applications into geo-distributed clouds," in *Proc. INFOCOM*, Mar. 2012, pp. 684–692.

[16] J. Tang, W. Tay, and Y. Wen, "Dynamic request redirection and elastic service scaling in cloud-centric media networks," *IEEE Trans. Multimedia*, vol. 16, no. 5, pp. 1434–1445, Aug. 2014.

[17] C. Lin, H. Chin, and D. Deng, "Dynamic multiservice load balancing in cloud-based multimedia system," *IEEE Syst. J*, vol. 8, no. 1, pp. 225–234, Mar. 2014.

[18] W. Gong, Z. Chen, J. Yan, and Q. Shuai, "An optimal VM resource allocation for near-client-datacenter for multimedia cloud," in *Proc. ICUFN*, Jul. 2014, pp. 249–254.

[19] R. Van den Bossche, K. Vanmechelen, and J. Broeckhove, "Cost-optimal scheduling in hybrid IaaS clouds for deadline constrained workloads," in *Proc. CLOUD*, Jul. 2010, pp. 228–235.

[20] L. Bittencourt, E. Madeira, and N. da Fonseca, "Scheduling in hybrid clouds," *IEEE Commun. Mag.*, vol. 50, no. 9, pp. 42–47, Sep. 2012.

[21] M. Shifrin, R. Atar, and I. Cidon, "Optimal scheduling in the hybrid-cloud," in *Proc. IFIP/IEEE Int. Symp. Integr. Netw. Manage.*, May 2013, pp. 51–59.

[22] R. Duan, R. Prodan, and X. Li, "Multi-objective game theoretic scheduling of bag-of-tasks workflows on hybrid clouds," *IEEE Trans. Cloud Comput.*, vol. 2, no. 1, pp. 29–42, Jan. 2014.

[23] H. Zhang, G. Jiang, K. Yoshihira, and H. Chen, "Proactive workload management in hybrid cloud computing," *IEEE Trans. Netw. Service Manage.*, vol. 11, no. 1, pp. 90–100, Mar. 2014.

[24] H. Li, L. Zhong, J. Liu, B. Li, and K. Xu, "Cost-effective partial migration of VoD services to content clouds," in *Proc. CLOUD*, Jul. 2011, pp. 203–210.

[25] W. Zhang, Y. Wen, J. Cai, and D. Wu, "Toward transcoding as a service in a multimedia cloud: Energy-efficient job-dispatching algorithm," *IEEE Trans. Veh. Technol.*, vol. 63, no. 5, pp. 2002–2012, Jun. 2014.

[26] S. Ren, Y. He, and F. Xu, "Provably-efficient job scheduling for energy and fairness in geographically distributed data centers," in *Proc. ICDCS*, Jun. 2012, pp. 22–31.

[27] H. Hu, Y. Wen, T. Chua, Z. Wang, J. Huang, W. Zhu, and D. Wu, "Community based effective social video contents placement in cloud centric CDN network," in *Proc. ICME*, Jul. 2014, pp. 1–6.

[28] X. Qiu, H. Li, C. Wu, Z. Li, and F. Lau, "Cost-minimizing dynamic migration of content distribution services into hybrid clouds," *IEEE Trans. Parallel Distrib. Syst.*, to be published.

[29] F. Liu, Z. Zhou, H. Jin, B. Li, B. Li, and H. Jiang, "On arbitrating the power-performance tradeoff in SaaS clouds," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 10, pp. 2648–2658, Oct. 2014.

[30] R. Urgaonkar, U. Kozat, K. Igarashi, and M. Neely, "Dynamic resource allocation and power management in virtualized data centers," in *Proc. NOMS*, Apr. 2010, pp. 479–486.

[31] R. Cahn, *Wide Area Network Design: Concepts and Tools for Optimization*. San Mateo, CA, USA: Morgan Kaufmann, 1998.

[32] H. Xu and B. Li, "Joint request mapping and response routing for geo-distributed cloud services," in *Proc. INFOCOM*, Apr. 2013, pp. 854–862.

[33] M. Zink, K. Suh, Y. Gu, and J. Kurose, "Characteristics of YouTube network traffic at a campus network - measurements, models, and implications," *Comput. Netw.*, vol. 53, pp. 501–514, Mar. 2009.

[34] M. Neely, "Opportunistic scheduling with worst case delay guarantees in single and multi-hop networks," in *Proc. INFOCOM*, Apr. 2011, pp. 1728–1736.

Ping Lu, photograph and biography not available at the time of publication.

Quanying Sun, photograph and biography not available at the time of publication.

Kaiyue Wu, photograph and biography not available at the time of publication.

Zuqing Zhu, (S'02–M'06–SM'12) photograph and biography not available at the time of publication.