On Achieving Cost-Effective Adaptive Cloud Gaming in Geo-Distributed Data Centers

Hao Tian, Di Wu, *Member, IEEE*, Jian He, Yuedong Xu, and Min Chen, *Senior Member, IEEE*

***Abstract*— Cloud gaming has become a new trend for gamers to access high-end video games. By rendering games in the remote cloud and streaming video scenes to the users, games can be played anywhere, anytime, on any device (e.g., smartphones, tablets, or personal computers). In this paper, we address the problem of achieving cost-effective adaptive cloud gaming in geo-distributed data centers from the perspective of cloud gaming service providers (CGSPs). Unlike previous work, we consider a cloud gaming system supported with the adaptive streaming technology. Our purpose is to minimize the overall service cost for CGSPs, by adaptively adjusting the selection of data centers, virtual machine allocation and video bitrate configuration for each user. Meanwhile, we also need to ensure good-enough quality of experience (QoE) for gamers. To this objective, we formulate the problem into a constrained stochastic optimization problem, and apply the Lyapunov optimization theory to drive the corresponding online strategy with provable upper bounds. Due to the diverse QoE requirements of video games, we also take the difference among game genres into account during the algorithm design. Finally, we conduct extensive trace-driven simulations to evaluate the effectiveness of our algorithm and our results show that our proposed algorithm can achieve significant gain over other alternative approaches.**

***Index Terms*— Adaptive streaming, cloud gaming, data centers, Lyapunov optimization.**

I. Introduction

IN RECENT years, a new type of cloud service called *cloud gaming* is becoming more and more popular in the gaming industry. The great potential of cloud gaming has been evidenced by its rapidly expanding scale, and it has attracted remarkable attention from both industry and academia. It is predicted that the overall revenue of the global video games will reach $64 billion in 2017, with online download and stream sales set to capture more than 60% of the market. In addition, the market of cloud gaming is expected to grow ninefolds by 2017, reaching $8 billion sales [1]. Some well-known cloud gaming platforms include OnLive [2], Gaikai [3], G-Cluster [4], and StreamMyGame [5].

Cloud gaming (also known as *gaming on demand*) is a radically different type of online gaming services, in which games are stored, synchronized, and rendered in remote cloud servers and delivered to players using streaming technology. Cloud gaming has become a new trend to play high-end 3-D video games, and makes gaming easier and more affordable for users. Game players are relieved from downloading or installing the original game software, and there is also no need for players to constantly upgrade their hardware. Instead, the cloud will take over all the computation-intensive tasks and stream the video of the gameplay to the player. Users can interact with the game in the cloud by sending control signals (e.g., key strokes and mouse clicks) to the remote gaming servers.

Existing cloud gaming service providers (CGSPs) generally rely on a set of geographically distributed data centers to deliver their services. Upon receiving a user request, the cloud gaming platform will redirect the user request to a particular data center according to certain polices (e.g., proximity and load balancing), and launch a dedicated gaming server or a virtual machine (VM) with specialized graphic hardware to run the requested game. The gaming server (or VM) is responsible for rendering and then streaming encoded game frames back to the user via the broadband network.

However, it is very difficult to design a cost-effective cloud gaming platform that can provide users with high quality of experience (QoE). The challenges are multifolds as follows. First, compared with traditional video-on-demand service, cloud gaming is more interactive and delay sensitive. According to the measurement work in [6], online game players are pretty impatient and sensitive to the interaction delay during game playing. Thus, it is important for a CGSP to adjust resource provisioning for users dynamically to meet the delay constraint. Second, different game genres have different requirements on interaction delay. As pointed out in [7], such requirements on interaction delay vary significantly across different game genres. For example, the increase of interaction delay is intolerable for the first-person shooter (FPS) games, while players of the war strategy games will not

Manuscript received December 10, 2014; revised February 14, 2015 and March 14, 2015; accepted March 22, 2015. Date of publication March 25, 2015; date of current version December 3, 2015. This work was supported in part by the National Science Foundation of China under Grant 61272397 and Grant 61402114, in part by the Guangdong Natural Science Funds for Distinguished Young Scholar under Grant S20120011187, in part by the International Science and Technology Collaboration Program under Grant 2014DFT10070 through the Ministry of Science and Technology, China, in part by the Hubei Provincial Key Project under Grant 2013CFA051, and in part by the Shanghai Pujiang Program under Grant 14PJ1401400. This paper was recommended by Associate Editor S. Shirmohammadi. *(Corresponding author: Di Wu.)*

H. Tian and D. Wu are with the Department of Computer Science, Sun Yat-sen University, Guangzhou 510006, China, and also with the SYSU-CMU Shunde International Joint Research Institute, Foshan 528300, China (e-mail: tianhao5@mail2.sysu.edu.cn; wudi27@mail.sysu.edu.cn).

J. He is with the Department of Computer Science, The University of Texas at Austin, Austin, TX 78712 USA (e-mail: jianhe@cs.utexas.edu).

Y. Xu is with the Department of Electronic Engineering, Fudan University, Shanghai 200433, China (e-mail: ydxu@fudan.edu.cn).

M. Chen is with the School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: minchen@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/TCSVT.2015.2416563

perceive obvious difference for the same interaction delay. Therefore, the decision on resource provisioning should also take the difference among game genres into account. Third, the heterogeneity of user devices (e.g., smartphones, pads, TVs, and tablets) demands adaptive streaming of video games. There exists significant diversity among user devices on the screen resolution, network bandwidth, computation capacity, and so on. Thus, the cloud gaming platform should be able to tune the video bitrate dynamically for each user and adapt to the changes of user demand and network conditions. Fourth, the delivery of cloud gaming services incurs significant monetary cost, such as bandwidth cost for game video streaming, rental cost for allocated VMs, and so on. To survive in the market competition, the CGSP must minimize its service cost as much as possible, in addition to providing good user QoE.

In this paper, we attempt to address the above challenges from the perspective of CGSPs. Our design objective is to achieve cost-effective adaptive cloud gaming (CACG), in which the service cost of CGSPs can be minimized as much as possible, while still ensuring good-enough QoE for game players. Toward this, we consider the adaptation of resource provisioning from three different angles: 1) adaptive data center selection; 2) adaptive VM allocation; and 3) adaptive video bitrate configuration. Mathematically, we formulate the objective problem into a constrained stochastic optimization problem, and exploit Lyapunov optimization theory [8] to solve the transformed problem. Our proposed algorithm is an online algorithm that can optimize data center selection, resource provisioning, and video bitrate adaptation jointly. Our algorithm can also approach the optimality with a provable performance bound. In summary, our main contributions are listed as follows.

1) We consider the problem of delivering CACG from the perspective of CGSPs, and aim at cutting down the service cost of CGSPs, by optimizing data center selection, VM allocation, and video bitrate adaptation jointly. In addition, we also take the difference among game genres into account, and our algorithm can meet the QoE requirements of different game genres.
2) We formulate the problem into a constrained stochastic optimization problem and utilize the Lyapunov optimization theory to solve the problem. We design a joint user dispatching and resource allocation algorithm, called CACG, for the CGSPs. Our proposed online algorithm can achieve adaptive resource allocation for each user and minimize the service cost for CGSPs. Meanwhile, we can also ensure good-enough QoE for game players. Unlike heuristic algorithms, our algorithm can approach the optimality with an explicitly provable performance upper bound.
3) We perform extensive trace-driven simulations to verify the effectiveness of our propose CACG algorithm in the practical settings. Our simulation results shows that, compared with other alternatives, CACG can save at least 25% of service cost, while providing even better QoE for game players.

The rest of this paper is organized as follows. Section II reviews previous work in the area of cloud gaming. Section III provides the details of our system model and problem formulation. Section IV describes the design of our proposed online algorithms. In Section V, we evaluate the effectiveness of our proposed algorithm by simulation. Finally, Section VI concludes the whole paper and discusses some future work.

II. Related Work

In spite that cloud gaming is a relatively new paradigm to deliver games over the Internet, it has received significant attention from industry and academia in recent years.

Quite a few measurement studies have been conducted to better understand the architecture and performance of existing cloud gaming systems. Chen *et al*. [10] measured the response delay of two leading cloud gaming platforms, namely, OnLive and StreamMyGame. They proposed a novel delay measurement method using hooking mechanism in Windows to inject the instrumentation code. Choy *et al*. [11] demonstrated through a large-scale measurement study that the existing cloud infrastructure is hard to meet the strict latency requirements of end users. Manzano *et al*. [12], [13] measured traffic characteristics of cloud gaming, analyzed the differences in network traffic across different games, and also described the detailed protocols for player assignments in OnLive.

To understand the impact of different game genres on the user experience for cloud gaming, Quax *et al*. [7] performed a qualitative comparison of four different game genres using a combined objective and subjective approaches, and showed that games that require intense interactions (e.g., action and racing games) are more sensitive to the delay compared with puzzle and strategy games. Lee *et al*. [14] pointed out that the same latency may have very different impacts on the quality of gaming experience for different games, depending on their delay sensitivity. They also developed a model to predict the real-time strictness of a game based on the rate of player inputs and the changes of game screen. Suznjevic *et al*. [15] analyzed network traffic generated by various types of cloud games, and examined the relationship between traffic characteristics, video patterns, and player inputs.

As cloud gaming relies on the cloud platform for service delivery, another thread of research focused on the problem of dynamic cloud resource provisioning. Unlike the conventional cloud-assisted video streaming [16]–[20], cloud gaming is inherently a latency-sensitive service, which makes the problem of resource provisioning more challenging. Marzolla *et al*. [21] considered the resource provisioning problem in massively multiplayer online games, which is quite different from the scenario in cloud gaming. Duong *et al.* [22] proposed a QoS-aware provisioning strategy to maximize the revenue of the service provider by satisfying predefined QoS requirements, but they only studied resource allocation in a single data center. Wu *et al*. [23] proposed an online control algorithm to cut down the provisioning cost while still ensuring the user QoE requirements, particularly the waiting time of

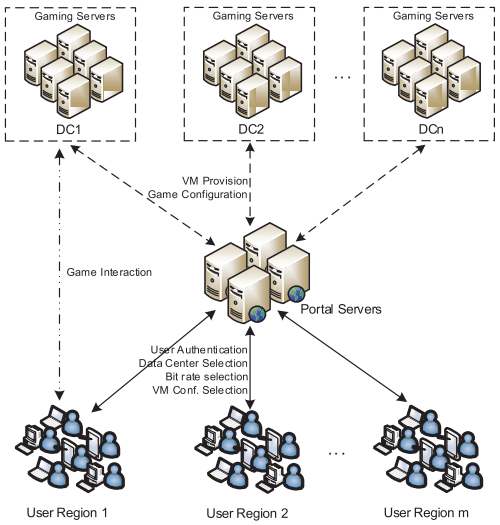


Fig. 1. Typical architecture of a cloud gaming platform.

game players in the queue. Finkel *et al*. [24] considered the case in which each cloud server can only host a subset of game replicas, and designed efficient game distribution strategies to reduce disk space requirements significantly. Hong *et al.* [25] studied the VM placement problem in the cloud gaming platform, however, their work did not take the video bitrate adaptation into account.

This paper differs from previous work in the following aspects: first, instead of considering only single data center, we studied the scenario of request dispatching and resource provisioning among multiple geo-distributed data centers jointly. Second, we considered the integration of adaptive video streaming with cloud gaming, thus game players are able to adaptively configure their video bitrate based on the current network condition. Third, we take the impact of game genres into consideration, therefore, our proposed algorithm is applicable for different games with various latency and bitrate requirements. Finally, our algorithm can achieve a good-enough tradeoff between provisioning cost and user QoE with explicitly provable performance bounds. It can significantly reduce the provisioning cost of CGSPs without sacrificing user QoE.

III. Problem Formulation

A. System Model

In this section, we consider a typical cloud gaming platform, as shown in Fig. 1.

Users can play cloud games with different hardware devices (e.g., laptops, smartphones, and tablets). To improve the QoE for users in different regions, the CGSP normally deploys its service on multiple geographically distributed data centers. In the figure, portal servers are the entry point for gamers to access the cloud gaming platform. The portal server is responsible for user authentication, data center selection, video bitrate selection, and VM configuration. Physically, the portal servers can be a set of cluster servers located in multiple locations. When a user accesses the cloud gaming platform, he should first connect to the portal server and authenticate himself, and then select the game to play. After the portal server learns the bandwidth condition of a user, it then redirects the user request to a proper data center. Based on the results obtained from bandwidth test, the portal server determines the bitrate to encode the game frames. Later a gaming server (e.g., a VM) is created and properly configured in the selected data center. The user can then connect with the gaming server and start game playing. When the game has been launched, the generated video game traffic flows between the user and the gaming server directly, instead of going through the portal server. The portal server only needs to handle control data traffic. Thus, the portal server will not become a bottleneck of the whole system.

Assume that the CGSP deploys its infrastructure on K geo-distributed data centers, denoted by $\mathcal{DC} = \{\mathrm{DC}_1, \mathrm{DC}_2, \ldots, \mathrm{DC}_K\}$. In each data center, we suppose that there are D different available VM configurations (e.g., CPU, GPU, and memory), represented as $\mathcal{C} = \{C_1, C_2, \ldots, C_D\}$. To play a game, each user should be allocated with one properly configured VM. The CGSP can scale up or down the cloud resources (e.g., the number of VMs and the amount of bandwidth) provisioned in each data center to meet the dynamic user demand.

We also assume that each data center stores a replica of all the supported games, so that a user can directly play any game from any data center. By enabling adaptive streaming on gaming servers [26], the game scenes can be encoded into M different bitrates, denoted by $\{B(j), j = 1, \ldots, M\}$. In our model, the time is divided into a series of time slots and each time slot lasts for a period of τ. To facilitate our presentation, we define an indicator $X^t(i, j)$ to indicate whether a user i receives a video stream with the jth bitrate at time slot t

$$X^t(i, j) = \mathbf{1}\ \{\text{user } i \text{ receives a stream with the } j\text{th bitrate}\}.$$

In general, a CGSP will provide a wide range of video games in different categories (e.g., action adventure, FPS, and war strategy). Each category of games has different QoE requirements on bandwidth and delay. According to the category of games played by users, we divide all users into L different groups, denoted as $\mathcal{G} = \{G_1, G_2, \ldots, G_L\}$. Users in the same group play games in the same or similar categories with similar QoE requirements. For example, a user who is playing action adventure games may need a higher bitrate to achieve smooth video quality, while the basic QoE of a war strategy game player can be satisfied with a much lower bitrate. We define a function $G(i)$ to indicate the index of the group that a user i belongs to. Let $A(l)$ be the basic bitrate required by users in the group l. To ensure a basic game experience, a user i should request a video stream with a bitrate no less than $A(G(i))$. To guarantee that the actual video bitrate that a user i receives is no less than the basic bitrate requirement, we must ensure that

$$\sum_{j=1}^{M} B(j) \cdot X^t(i, j) \geq A(G(i)) \tag{1}$$

for each user i, where $A(G(i))$ is the basic bitrate that a user i requires to get a comfortable experience, and the left-hand side

of the inequality is the actual bitrate allocated to the user i by the system.

B. Delay Model

The QoE of game players is sensitive to interaction delay of cloud gaming. In [10], *interaction delay* is defined as the lag between the time the client sends a player's command to the server and the time the generated video frame is decoded and presented on the screen. Interaction delay mainly contains three components: 1) *network delay* (at the network side); 2) *processing delay* (at the server side); and 3) *playout delay* (at the client side). Network delay is usually referred to as the network round-trip time, which can be measured by tools such as Ping and King [27]. Processing delay is the difference between the time the server receives a user's command and the time the server responds with a corresponding rendered frame. As for playout delay, it is the time required for the client to decode and display the encoded frame on the screen of the local machine. Because playout delay is usually constant and not affected by the cloud side, we do not include it in our model for brevity. Thus, we redefine the interaction delay of a user i as the sum of processing delay $D_p^t(i)$ and network delay $D_n^t(i)$, which can be represented as below

$$D^t(i) = D_p^t(i) + D_n^t(i). \tag{2}$$

Processing delay is determined jointly by the game genre, the VM configuration and the allocated bitrate, while network delay is mainly determined by the network condition between the user and the data center. To ease our presentation, we define $Y^t(i,k)$ to indicate whether a user i is redirected to the kth data center and $Z^t(i,d)$ to indicate whether a user i is allocated with the dth type VM at time slot t, respectively

$$Y^t(i,k) = \mathbf{1}\{\text{user } i \text{ is redirected to the } k\text{th data center}\}$$
$$Z^t(i,d) = \mathbf{1}\{\text{user } i \text{ is allocated with the } d\text{th type VM}\}.$$

Then, we can derive the processing delay and the network delay of a user i as follows:

$$D_p^t(i) = \sum_{j=1}^{M}\sum_{d=1}^{D} \tilde{D}_p^t(G(i), B(j), d) \cdot X^t(i,j) \cdot Z^t(i,d)$$
$$D_n^t(i) = \sum_{k=1}^{K} \tilde{D}_n^t(i,k) \cdot Y^t(i,k)$$

where $\tilde{D}_p^t(G(i), B(j), d)$ is the average processing delay for a user i when the assigned bitrate is $B(j)$ and the allocated VM is d-type, while $\tilde{D}_n^t(i,k)$ is the average network delay between a user i and a data center k at time slot t.

C. User Utility Model

For users who are playing games in the same or similar categories, we assume that they have a similar utility function. In cloud gaming, if the delay constraint can be satisfied, the utility of a game player is largely determined by the game genre and the allocated video bitrate. Therefore, suppose that the delay constraint can be met, we define the utility function of a user i allocated with a bitrate $B(j)$ as $\phi(G(i), B(j))$, where $G(i)$ is the index of the group that a user i belongs to. $\phi(\cdot)$ is usually a concave function of the allocated bitrate $B(j)$. Therefore, in each time slot t, the utility of a user i can be defined as

$$\Phi^t(i) = \sum_{j=1}^{M} X^t(i,j) \cdot \phi(G(i), B(j)). \tag{3}$$

Let U^t be the set of users in the system at the beginning of time slot t. The overall utility of all users in the system at time slot t can be derived as

$$\Phi^t = \sum_{i \in U^t} \Phi^t(i). \tag{4}$$

D. Service Cost Model

For geo-distributed data centers, the bandwidth price varies across different regions. In time slot t, let v_k^t and w_k^t be the unit bandwidth price and the amount of export bandwidth usage in the kth data center, respectively. Assume that the bandwidth price keeps constant within a time slot. Consider all data centers, the total bandwidth cost at time slot t can be given by

$$C_b^t = \sum_{k=1}^{K} v_k^t w_k^t$$

where $w_k^t = \sum_{i \in U^t} \sum_{j=1}^{M} X^t(i,j) \cdot Y^t(i,k) \cdot B(j)$. The above expression can also be rewritten as

$$C_b^t = \sum_{i \in U^t} C_b^t(i)$$

where $C_b^t(i)$ is the bandwidth cost incurred by a user i and

$$C_b^t(i) = \sum_{j=1}^{M}\sum_{k=1}^{K} X^t(i,j) Y^t(i,k) B(j) v_k^t.$$

In addition to the bandwidth cost, there also exists rental cost for each VM allocated to users. The CGSP should pay for the VMs rented from the cloud infrastructure provider. The VM rental fee is used to cover the cost incurred by electricity usage, machine maintenance and depreciation, and so on. Note that, in our system model, the CGSPs rent VMs from the cloud service providers (CSPs) to deliver cloud gaming services. According to the current pricing model (e.g., Amazon EC2), as a customer of CSPs, the CGSP only needs to pay for the VM rental cost. The additional cost incurred by VM reconfiguration should not be paid by the CGSP. In addition, it is also not necessary for a CGSP to take care of implementation details of the underlying physical cloud infrastructure (e.g., VM migration and VM consolidation), which are the responsibilities of the CSP. With the advance of virtualization techniques [28]–[30], seamless VM migration can be achieved for the cloud gaming service and the impact of VM migration to user QoE will not be a serious problem. In our system model, we consider a scenario that the CSP can support seamless or near-seamless VM migration.

As the rental cost is closely related to the VM configuration, we define the VM rental cost $C_o^t(i)$ incurred by a user i in time slot t as

$$C_o^t(i) = \sum_{k=1}^{K} \sum_{d=1}^{D} c_{dk} \cdot Y^t(i,k) \cdot Z^t(i,d)$$

where c_{dk} is the rental cost of the d-type VM in the kth data center in one time slot. Thus, the overall rental cost of all users in time slot t can be represented as

$$C_o^t = \sum_{i \in U^t} C_o^t(i).$$

For a CGSP, the service cost includes both bandwidth cost and VM rental cost. Thus, we can derive the total service cost in time slot t as

$$C^t = C_b^t + C_o^t.$$

E. Problem Formulation

For a CGSP, it is critical to minimize the cost of service delivery and increase the revenue in the meanwhile. Therefore, our objective in this paper is to design cost-efficient cloud resource allocation and bitrate adaptation algorithms while still respecting the QoE requirement of each user. To this purpose, we formulate the problem into the following stochastic optimization problem:

$$\mathbf{P1}.\ \min \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} (C^t - \alpha \cdot \Phi^t)$$

$$\text{s.t.} \sum_{j=1}^{M} X^t(i,j) = 1, X^t(i,j) \in \{0,1\} \quad \forall i \qquad (a)$$

$$\sum_{k=1}^{K} Y^t(i,k) = 1, Y^t(i,k) \in \{0,1\} \quad \forall i \qquad (b)$$

$$\sum_{d=1}^{D} Z^t(i,d) = 1, Z^t(i,d) \in \{0,1\} \quad \forall i \qquad (c)$$

$$\sum_{j=1}^{M} B(j) \cdot X^t(i,j) \geq A(G(i)) \quad \forall i \qquad (d)$$

$$\lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \frac{1}{N_l^t} \sum_{i \in U_l^t} D^t(i) \leq \epsilon_l \quad \forall l. \qquad (e)$$

In the above problem, the objective function contains two components. One is the long-term time-average service cost incurred by bandwidth usage and VM rental, which should be minimized as much as possible. The other is the total utility of all users, which should be maximized to improve user engagement on game playing. These two objective components are conflicting with each other. To optimize two conflicting objective components simultaneously, it is a common optimization technique to combine them into one weighted objective function. The tradeoff between two objective components can be tuned by the weight α.

The first three constraints [i.e., constraints (a), (b), and (c)] ensure that a unique bitrate, data center, and VM configuration must be selected for a particular user at the beginning of each time slot. The constraint (d) is used to ensure that the allocated bitrate should be no less than the basic bitrate for each user, so that users can enjoy a fluent game experience. The last constraint (e) guarantees that the time-average interaction delay of users in each group should be less than a predefined threshold (e.g., 100 ms). Intuitively, different game categories have different delay and QoE requirements. For each group of users, we define the threshold as ϵ_l, where l is the group index, and the group size is N_l^t at time slot t. When the interaction delay of a user is under a certain threshold, which is enough to ensure good user experience, a further reduction of interaction delay brings marginal benefit to a user. In this case, the CGSP cares more about the minimization of service cost in running the service.

The value of α represents the relative importance between service cost and user utility. The service provider can configure the value of α based on its bias on these two objective components. However, even if we put too much weight on the service cost, the user utility will only be moderately impacted, instead of being too lousy. In our problem formulation, we need to ensure the constraints (d) and (e) when optimizing the objective function. By satisfying the constraints during optimization, we can guarantee to meet a user's basic requirements on bandwidth and latency, which is essential for a user to have a basic gaming experience. When we increase the weight on the user utility, a much better gaming experience can be achieved. The decision of the optimal α value depends on the bias of a CGSP, which is pretty subjective and beyond the scope of this paper.

IV. Design of Online Algorithm

To solve the constrained stochastic optimization problem **P1**, we exploit the Lyapunov optimization theory [8] to design online control strategies for cloud resource allocation and video bitrate adaptation. A major benefit of Lyapunov optimization is that it does not require any priori knowledge about user behaviors and network conditions. By taking actions to greedily minimize the drift-plus-penalty in each time slot, it can provide provable performance with explicit bounds. Other approaches (e.g., Markov processes) can also be used to solve this problem, but these approaches need some priori knowledge about user behaviors and network conditions.

In the framework of Lyapunov optimization, the original stochastic optimization problem can be transformed into an optimization problem of minimizing the Lyapunov drift-plus-penalty. Using Lyapunov optimization, the time average constraints in Problem $P1$ can be transformed into a set of queue stability constraints [8]. In this paper, we will redefine the virtual queues in the context of adaptive cloud gaming and prove that delay constraints in the original optimization problem can be transformed into a set of queue stability problems.

A set of virtual queues $\mathbf{H} = \{H_1, H_2, \ldots, H_L\}$ are introduced for problem solving. H_l denotes the virtual

QoE queue of the lth user group and $H_l(t)$ denotes the queue backlogs in the lth user group at time slot t. The update of the queue H_l is described as

$$H_l(t+1) = \max\left\{H_l(t) + \frac{1}{N_l^t}\sum_{i\in U_l^t} D^t(i) - \epsilon_l, 0\right\}.$$

In Lemma 1, we can prove that the delay constraint (e) can be satisfied if the virtual queue H_l is stable.

Lemma 1: If the virtual queue H_l is stable, then the delay constraint (e) can be satisfied

$$\lim_{T\to\infty}\frac{H_l(t)}{t} = 0 \Rightarrow \lim_{T\to\infty}\frac{1}{T}\sum_{t=0}^{T-1}\frac{1}{N_l^t}\sum_{i\in U_l^t} D^t(i) \le \epsilon_l.$$

Proof: See [9, Appendix A] for the proof details. ■

Define the Lyapunov function as $L(t) = (1/2)\sum_{l=1}^{L} H_l^2(t)$ and Lyapunov drift as $\Delta(\mathbf{H}(t)) = L(t+1) - L(t)$. According to the Lyapunov optimization framework, we can then obtain the drift-plus-penalty by adding the cost into the drift, namely

$$\Delta(\mathbf{H}(t)) + V\mathbb{E}\left\{C_b^t + C_o^t - \alpha\Phi^t|\mathbf{H}(t)\right\}$$

where $\mathbf{H}(t) = \{H_1(t), H_2(t), \ldots, H_L(t)\}$, and V is a tunable parameter that affects the performance of the online algorithm. Therefore, the solution of the original stochastic optimization problem can be approximately obtained by solving the problem of minimizing the drift-plus-penalty in each time slot. That is, **P1** can be transformed into the following optimization problem **P2**:

$$\begin{aligned}\textbf{P2}.\ \min\ & \Delta(\mathbf{H}(t)) + V\mathbb{E}\left\{C_b^t + C_o^t - \alpha\Phi^t|\mathbf{H}(t)\right\}\\ \text{s.t. }& (a)(b)(c)(d).\end{aligned}$$

To solve Problem **P2**, we should first derive the upper bound of the drift-plus-penalty, denoted by Θ_1. The upper bound Θ_1 can be derived in the following lemma.

Lemma 2: For any feasible X^t, Y^t, and Z^t, the drift-plus-penalty is upper bounded by Θ_1, namely

$$\Delta(\mathbf{H}(t)) + V\mathbb{E}\left\{C_b^t + C_o^t - \alpha\Phi^t|\mathbf{H}(t)\right\} \le \Theta_1$$

where $\Theta_1 = V\mathbb{E}\{C_b^t + C_o^t - \alpha\Phi^t|\mathbf{H}(t)\} + \mathbb{E}\{\sum_{l=1}^{L} H_l(t)(1/N_l^t)\sum_{i\in U_l^t} D^t(i)|\mathbf{H}(t)\} + (LD_{\max}^2 + \sum_{l=1}^{L}\epsilon_l^2/2)$. Here, we assume that $D^t(i)$ is an increasing and convex function, and upper bounded by $D_{\max}$, i.e., $D^t(i) \le D_{\max}, \forall i$.

Proof: See [9, Appendix B] for the proof details. ■

Rather than directly minimizing the drift-plus-penalty, our strategy actually seeks to minimize the upper bound Θ_1. Here, we combine Θ_1 with the definition of C_b^t, C_o^t, and Φ^t. Therefore, we get the optimization problem **P3**

$$\begin{aligned}\textbf{P3}.\ \min & \sum_{i\in U^t}\left(\frac{H_{G(i)}(t)}{N_{G(i)}^t}D^t(i) + VC_b^t(i) + VC_o^t(i) - V\alpha\Phi^t(i)\right)\\ \text{s.t. }& (a)(b)(c)(d).\end{aligned}$$

Note the decision vector associated with a user i at time slot t as $S_i(t) = (X^t(i,j), 1 \le j \le M, Y^t(i,k), 1 \le k \le K, Z^t(i,d), 1 \le d \le D)$, and the set of all decisions is represented by $\mathbf{S}_i(t) = (S_i(t)|(a)(b)(c)(d))$, where the constraints (a), (b), (c), and (d) are given in Problem **P1**. Each user can be assigned with only one decision $S_i(t)$ from the set $\mathbf{S}_i(t)$ at each time slot.

Algorithm 1 One-Shot Algorithm

Input:
The value of L, M, K, V, α
Decision set $\mathbf{S}_i(t)$
Number of users in each group N_l^t
Utility function for users in each group $\phi(l,j)$
Processing delay and network delay $\tilde{D}_p^t(G(i), B(j), d)$, $\tilde{D}_n^t(i,k)$
Delay that users in each group can tolerate ϵ_l
Bandwidth price of each data center v_k^t
Rental cost of each VM configuration c_{dk}

Output:
User decision $S_i^*(t), \forall i \in U^t$

1: Initialization step:
$x(S_i(t)) = 0, \forall i \in U^t, \forall S_i(t) \in \mathbf{S}_i(t)$,
2: **for** $\forall i \in U^t$ **do**
3: $S_i^*(t) = \arg\max_{S_i(t)\in\mathbf{S}_i(t)} \tilde{u}_i(S_i(t))$;
4: **end for**

Define $x(S_i(t))$ as an indicator function to indicate whether a user i is assigned with the decision $S_i(t)$, thus $x(S_i(t)) \in \{0,1\}$. Then, we can transform the constraints (a), (b), (c), and (d) in the optimization problem **P3** to a simple constraint as

$$\sum_{S_i(t)\in\mathbf{S}_i(t)} x(S_i(t)) = 1 \quad \forall i \in U^t.$$

Here, we denote $\tilde{u}_i(S_i(t)) = -((H_{G(i)}(t)/N_{G(i)}^t)D^t(i) + VC_b^t(i) + VC_o^t(i) - V\alpha\Phi^t(i))$. Therefore, we can further simplify the representation of Problem **P3** as follows:

$$\begin{aligned}\textbf{P4}.\ \max & \sum_{i\in U^t}\sum_{S_i(t)\in\mathbf{S}_i(t)} \tilde{u}_i(S_i(t))\cdot x(S_i(t))\\ \text{s.t. }& \sum_{S_i(t)\in\mathbf{S}_i(t)} x(S_i(t)) = 1 \quad \forall i \in U^t\\ & x(S_i(t)) \in \{0,1\} \quad \forall i \in U^t \quad \forall S_i(t) \in \mathbf{S}_i(t).\end{aligned}$$

Thus, the original complex optimization problem can be transformed into a simple optimization problem. We design an one-shot algorithm (as shown in Algorithm 1) to solve Problem **P4**, which obtains the optimal result for **P4** and significantly reduces the computational complexity.

In the second line of Algorithm 1, we can see that we need to iterate over the decision $S_i(t) \in \mathbf{S}_i(t)$ of each user i to find the best decision vector that maximizes the value of $\tilde{u}_i(S_i(t))$. From the definition of $S_i(t)$, it is easy to verify that the maximal size of decision set $\mathbf{S}_i(t)$ is at most MKD. Thus, the computational complexity for executing the iterations is $O(MKDN)$ by even using a brutal search method. It also indicates that our proposed algorithm is a polynomial-time algorithm and its computational complexity is irrelevant to the value of V.

After utilizing the Lyapunov optimization theory, the selection of user decisions can be realized by solving the

Algorithm 2 Online Resource Allocation Algorithm

Input:

The value of L, M, K, V, α
Decision set $\mathbf{S}_i(t)$
Number of users in each group N_l^t
Utility function for users in each group $\phi(l, j)$
Processing delay and network delay $\tilde{D}_p^t(G(i), B(j), d)$, $\tilde{D}_n^t(i, k)$
Delay that users in each group can tolerate ϵ_l
Bandwidth price of each data center υ_k^t
Rental cost of each VM configuration c_{dk}

Output:

User decision $S_i^*(t), \forall i \in U^t, t$
Bitrate selection, datacenter selection and VM configuration variables $X^t(i, j), Y^t(i, k), Z^t(i, d), \forall i \in U^t, t$, $1 \le j \le M, 1 \le k \le K, 1 \le d \le D$

1: Initialization step: Let $t = 0$, and set $H_l(0) = 0$, for $l = 1, 2, \ldots, L$
2: **while** the cloud gaming service is running **do**
3: At the beginning of time slot t, monitor the queue backlog $H_l(t)$ and the real-time information of bandwidth price υ_k^t;
4: Update information of the set of all game users U^t, users of each group U_l^t, and the amount of users in each group N_i^t. In addition, update network delay between each user and data center $\tilde{D}_n(i, k)$;
5: Calculate the user decisions $\{S_i^*(t), \forall i \in U^t\}$ according to Algorithm 1;
6: Derive the strategy of bitrate selection, data center selection and VM configuration selection $\{X^t(i, j), \forall i \in U^t, 1 \le j \le M\}, \{Y^t(i, k), \forall i \in U^t, 1 \le k \le K\}$, $\{Z^t(i, d), \forall i \in U^t, 1 \le d \le D\}$ according to user decisions $\{S_i^*(t), \forall i \in U^t\}$;
7: Update virtual queues $\mathbf{H}$ according to $H_l(t+1) = \max\{H_l(t) + \frac{1}{N_l^t}\sum_{i \in U_l^t} D^t(i) - \epsilon_l, 0\}, \forall 1 \le l \le L$;
8: Set $t + 1 \rightarrow t$F.
9: **end while**

one-shot problem **P4** at each time slot. The details of our online algorithm for cloud resource allocation are given in Algorithm 2. The working process of Algorithm 2 can also be illustrated in Fig. 2. Our online algorithm can approach the optimal solution of the origin optimization problem within infinitely small distance. The distance to the optimality is determined by the tuning parameter V. We can obtain the performance bound with the following theorem.

Theorem 1: The performance bound of the time-average weighted value of service cost and user utility induced by our online algorithm (by solving Problem **P2**) can be given by

$$\lim_{T \to \infty} \frac{1}{T} \sum_{t=0}^{T-1} \left(C_b^t + C_o^t - \Phi^t\right) \le \hat{P} + \frac{\Lambda}{V}$$

where $\hat{P}$ denotes the infimum value of objective function in Problem **P1** over all stable policies, and $\Lambda = (LD_{\max}^2 + \sum_{l=1}^{L} \epsilon_l^2/2)$.

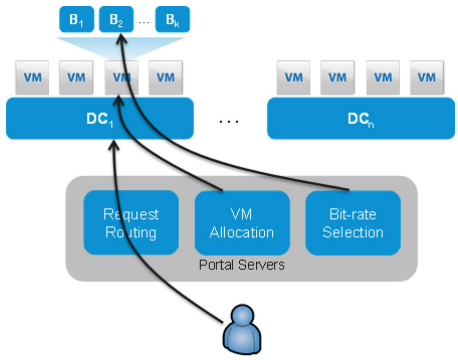


Fig. 2. Illustration of the working process of our online algorithm. The algorithm is running on the portal server and makes adjustments on datacenter selection, VM allocation, and bitrate selection periodically.

Proof: See [9, Appendix C] for the proof details. ■

The tunable parameter V determines the approximation extent of our algorithm to the optimality, and also the tradeoff between the overall cost and the interaction delay experienced by users. With a larger value of V, our algorithm performs close to the optimal value, but it is at the cost of a larger virtual queue length, which implies that game players will experience a larger interaction delay and results in a negative effect on the user experience.

V. Evaluations

In this section, we developed a discrete-event simulator to simulate the behavior of a CGSP under different decision strategies, and conducted a set of experiments to evaluate the effectiveness of our proposed online algorithm.

A. Simulation Setup

In our simulation, we consider a scenario in which a CGSP deploys its service in five geo-distributed data centers. To make our simulation more realistic, we use the real dataset on network delay among Internet nodes obtained from the Meridian Project [31], [32]. To take the dynamics of network delay into account, when using the Meridian real trace on network delay, we added a variance of 10% to network delay between nodes and varied the value in each time slot. In the simulation, we configure the bandwidth of each user higher than the basic bitrate requested by a user and also allow the variation of network bandwidth. In each time slot, we assume the arrival of game players follows the Poisson distribution with the mean value of 500.

Each data center can offer three different types of VM instances. The rental cost of different VM instances is set based on Amazon EC2's pricing model. We use the real VM pricing trace obtained from Amazon's Website [33]. According to the trace, we set the price of three types of VM instances as $0.07, 0.14$, and 0.28 (in units of dollars per hour), respectively. In Amazon's pricing model, the price of bandwidth usage varies with time. In our experiments, we assume that the bandwidth price takes values from the range $[0.05, 0.07]$ (in units of dollars per gigabyte) and changes over time. We also assume that the bandwidth price at each data center is independent and identically distributed.

The CGSP can scale up or down the number of VM instances and bandwidth provisioned in each data center.

Assume that there are totally five different game categories in the system for simplicity. According to the studies in [34]–[36], users who are playing games in different categories have different tolerance degrees on interaction delay. For example, it is observed that players of FPS games demand for less than 100 ms interaction delay, while 500 ms interaction delay is good enough for role-playing games [34]. Thus, we set the predefined threshold of interaction delay for the five game categories as 100, 200, 300, 400, 500 ms, respectively. Moreover, games in different categories have different requirements on the video bitrate to ensure a good game experience. Therefore, we assume that the basic bitrate of each game category follows a uniform distribution in [4, 5], [3, 4], [2, 3], [1, 2], and [0.5, 1] Mbits/s, respectively. In our simulation, each user randomly chooses a game from five different game categories, and the gaming session length of each user is simulated as a Weibull distribution according to the measurement work in [37] and [38]. The Weibull distribution contains two parameters: 1) the shape, $k > 0$ and 2) the scale, $\lambda > 0$. According to [37], we set the values of these two parameters as $\lambda = 137$ and $k = 0.93$. Note that, in real life, the derivation of game-specific parameters requires some additional effects of CGSPs. Most of parameters can be derived by performing direct measurements or analyzing the history data of system operation. For some QoE-related parameters, CGSPs can collect feedbacks from a sample set of game players at the end of each game session and use statistical techniques to derive the parameters.

The processing delay of each game in the cloud platform is set based on the measurement work in [26] and [39]. The processing delay is determined by three factors, namely, game category, video bitrate, and VM configuration. For games in the same category, the processing delay decreases if a more powerful VM is allocated to the user. Games with high interactivity and gorgeous graphic effects incur a higher processing delay for frame rendering. Moreover, a higher processing delay will be incurred if encoding game scenes into a higher bitrate video stream. In the trace, the processing delay is generally in the range of [30, 200] ms. In our paper, we consider interaction delay as the sum of network delay and processing delay, because playout delay is usually constant and will not be affected by the strategy we made. In our simulation, we set playout delay to 15 ms according to [39].

The user experience is affected by multiple system factors, such as the video bitrate, game category, and so on. Similar to the QoE model in [20], we define the utility function $\phi(\cdot)$ as a logarithmic function of the allocated video bitrate and the game category, namely

$$\phi(G(i), B(j)) = \log\left(1 + \frac{\sum_{j=1}^{M} B(j)X^t(i,j)}{A(G(i))}\right)$$

where $A(G(i)))$ is the basic bitrate required by a user i, and $\sum_{j=1}^{M} B(j)X^t(i,j)$ is the actual bitrate assigned for a user i. In the above definition, the utility received by the user increases concavely with the increase of the ratio between the actual streaming bitrate and the basic bitrate. The intuition behind such a definition is to ensure that the marginal benefit brought by increasing the video bitrate will be diminished when the video bitrate is high. The further increase of video bitrate will not generate significant improvement of user QoE.

Our proposed CACG algorithm can help the service provider adjust the selection of data center, video bitrate and the VM allocation for each user periodically according to his game genre and delay requirements. In the simulation, our algorithm adjusts the decision vector for each game player every 20 min. Note that our algorithm is only intended to optimize the overall resource provisioning periodically. For newly arrived users, we use the DDCS + MBS method (described later) to handle the incoming user requests timely.

We compare our proposed CACG algorithm with four other alternatives, each of which is a combination of a data center selection strategy and a bitrate selection strategy. For four other alternatives (excluding CACG), we assume that the system always assigns the user with the lowest VM configuration that is powerful enough to ensure the user's latency requirement. If the delay constraint is violated even with the best VM configuration, the system will assign the user with the best VM configuration.

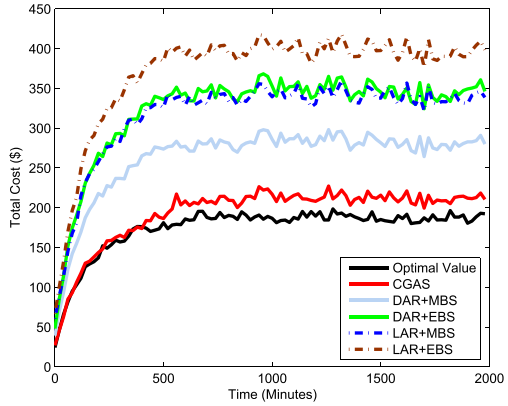
In our simulation, we consider the following two typical data center selection strategies.

1) *Delay-Sensitive Data Center Selection (DDCS):* In which user requests are always routed to a data center with the lowest network delay (for example, a data center is in a nearby region). It can reduce the interaction delay for a user.
2) *Load-Sensitive Data Center Selection (LDCS):* In which users are always routed to a data center with the lowest workload level. It can help to achieve load balancing among multiple data centers.

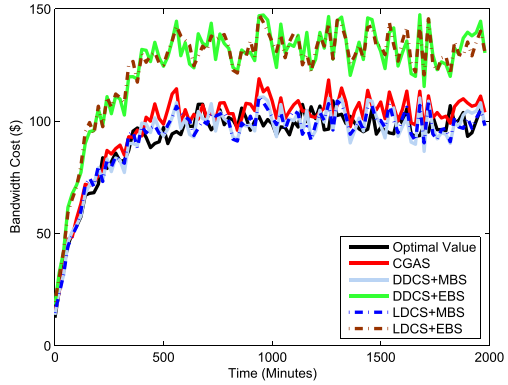
Two bitrate selection strategies are used for comparison in our experiments.

1) *Minimum Bitrate Selection (MBS):* In which users are always assigned with a bitrate that is equal to or just above their basic bitrate according to the game they play. In this case, the CGSP can satisfy the user's basic QoE requirement and minimize their bandwidth cost to support the cloud gaming service.
2) *Enhanced Bitrate Selection (EBS):* In which users care more about their gaming experience and wish to get a higher bitrate to guarantee their QoE. In our simulation, the CGSP will allocate a bitrate that is one level higher than the bitrate set by the MBS method. Thus, the service provider can offer a better gaming experience to users and attract more new game players to join. However, it is at the expense of higher bandwidth and rental cost.

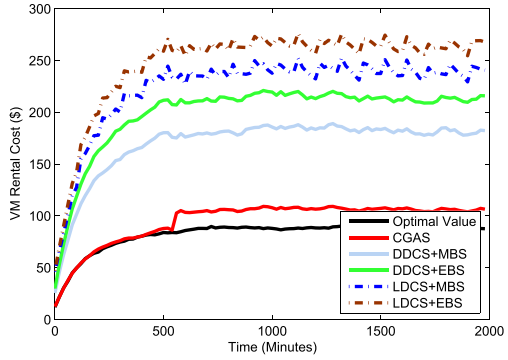
Therefore, we have five methods in comparison, namely: 1) DDCS + MBS; 2) DDCS + EBS; 3) LDCS + MBS; 4) LDCS + EBS; and 5) our proposed CACG algorithm.



(a)



(b)



(c)

Fig. 3. Comparison of service cost. (a) Total service cost. (b) Bandwidth cost. (c) VM rental cost.

B. Comparison of Service Cost

Fig. 3(a) shows the total service cost incurred by different methods during the simulation periods. Our simulation lasts for 2000 min. From the figure, we can find that LDCS + EBS incurs the highest service cost, and our proposed CACG has the lowest service cost. The curve of CACG is close to the optimal value with very small distance. More accurately, our method can reduce 48% of service cost compared with LDCS + EBS, 40% of service cost compared with DDCS + EBS and LDCS + MBS, and 25% of service cost compared with DDCS + MBS.

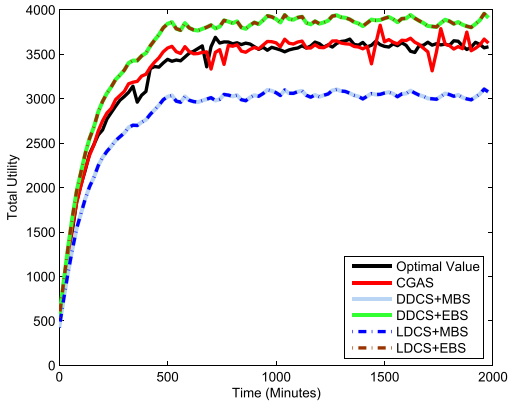


Fig. 4. Comparison of user utility.

In our paper, service cost includes two parts, bandwidth cost and VM rental cost. The comparison of bandwidth cost is shown in Fig. 3(b). We can observe that the bandwidth cost of DDCS + MBS and LDCS + MBS is the same and also the lowest among different methods. Bandwidth cost is determined by the bandwidth price and bandwidth consumption in each data center. As DDCS + MBS and LDCS + MBS always select the minimum bitrate that is just above the basic bitrate requirement, thus these two methods can achieve the lowest bandwidth cost. Our proposed CACG achieves a rather low bandwidth cost, which is close to the minimum value.

Fig. 3(c) further shows the VM rental cost incurred by different methods. Our proposed CACG can significantly reduce the VM rental cost compared with other methods. LDCS + EBS has the highest rental cost, as the selection of data centers is ignorant of latency, thus it results in higher network delay for each user. Besides, the method chooses an enhanced bitrate to encode the game scenes. To ensure the latency constraint, a more powerful VM should be allocated to reduce the processing delay.

C. Comparison of User Utility

Fig. 4 plots the comparison of user utility among different methods during our simulation. DDCS + MBS and LDCS + MBS have the lowest user utility, as both of them select the minimum bitrate for each user that is just above the basic bitrate requirement, which can only ensure a basic game experience. The methods using the EBS strategy achieve much better utility, because an enhanced bitrate will be chosen for each user so as to offer a much better game experience. Among all the methods, our proposed CACG achieves a much achieves a much higher utility value than that of DDCS + MBS and LDCS + MBS, and approaches the utility value of DDCS + EBS and LDCS + EBS with a small distance. However, the cost incurred by CACG is significantly lower than DDCS + EBS and LDCS + EBS as shown in the previous part.

Fig. 5 shows the average utility of users in different groups achieved by our CACG algorithm. In our simulation settings, different user groups have different QoE requirements, but users in the same group have similar QoE requirements. A user with a larger group index has a higher QoE requirement in terms of interaction delay and video bitrate. The results in the

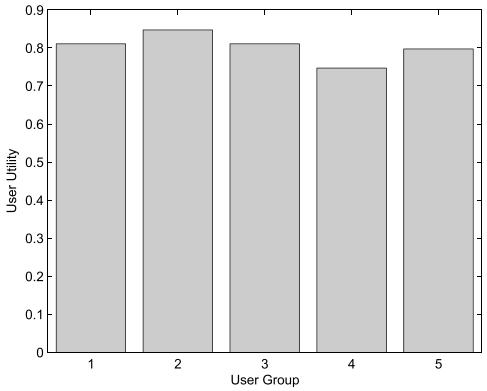
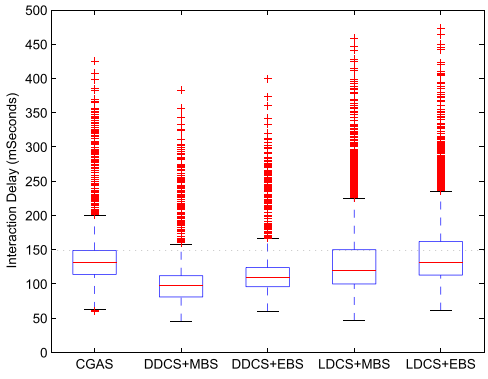
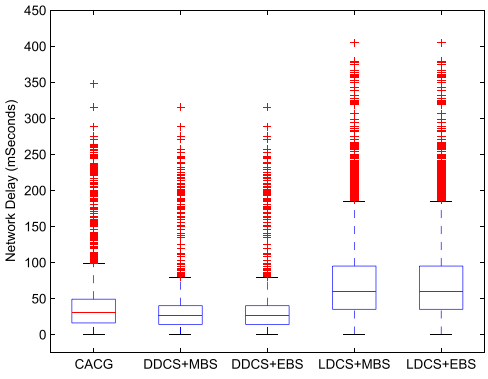


Fig. 5. Average utility of users in different groups achieved by CACG algorithm.



(a)



(b)

Fig. 6. Comparison of gaming latency. (a) Interaction delay. (b) Network delay.

figure show that users in different groups have very similar utility values under our CACG algorithm. It indicates that our proposed algorithm can guarantee a rather high level of fairness among users.

D. Comparison of Gaming Latency

Fig. 6 shows the comparison of gaming latency among different methods. From the results in Fig. 6(a), we can observe that our proposed CACG algorithm can reduce the interaction delay significantly. For 75% of users, their interaction delay is under 150 ms, and it is good enough

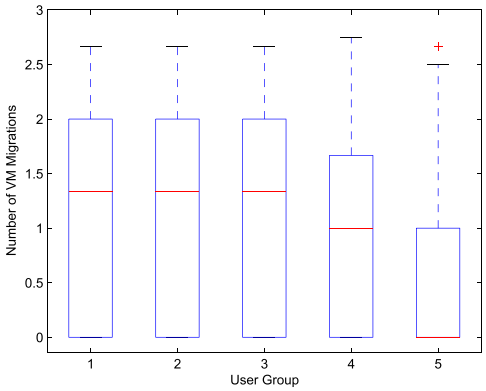


Fig. 7. Distribution of the number of VM migrations experienced by a user per hour.

to ensure a fluid game experience. In our formulation, we consider the interaction delay as a constraint that should be less than a threshold for each user. Obviously, when the delay constraint is satisfied and low enough to ensure good user experience, a further reduction of interaction delay brings marginal benefit to users. Thus, CACG does not try to minimize the interaction delay as much as possible.

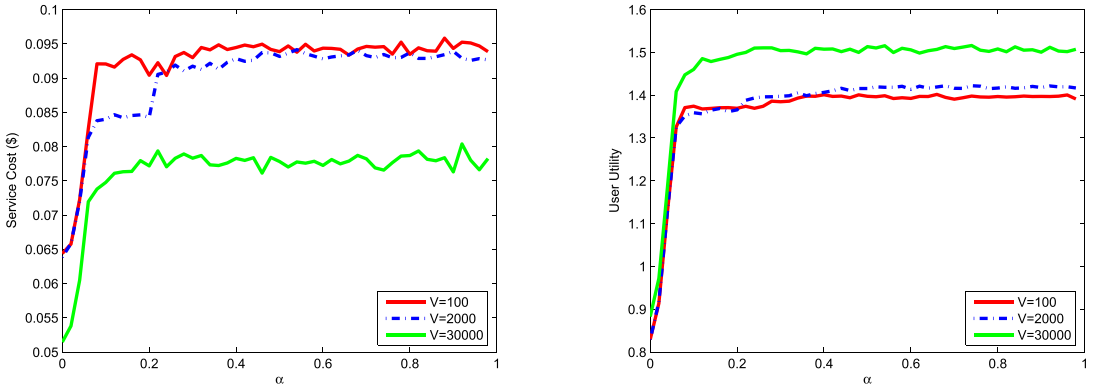
Among the five methods, DDCS + MBS performs the best due to the fact that it always routes user requests to the data center with the lowest network delay, and it also selects the bitrate that is just above the basic bitrate for each user, which helps to reduce the processing delay. The method DDCS + MBS has the worst performance, as it does not consider the network delay when redirecting each user to data centers. In addition, it chooses an enhanced bitrate for each user to ensure better QoE, which results in a much higher processing delay. With our CACG method, the variance of interaction delay can also be reduced significantly, and most outliers are caused by the nodes with bad network condition in the Meridian data set.

Fig. 6(b) plots the network delay incurred by different methods. For each user, the network delay is only related to the assigned data center. Thus, DDCS + MBS and DDCS + EBS perform the best, as both of them route users to the data center with the lowest network delay. In comparison, LDCS + MBS and LDCS + EBS have a much higher network delay. Our proposed CACG method can achieve a very low network delay, which approaches DDCS + MBS and DDCS + EBS with very small distance (normally with an increase of only 0–10 ms).

E. Impact of VM Migrations

As we all know, migration between data centers and VMs will possibly cause QoE degradation of game players. However, such migration cannot be completely avoided due to link congestion or system overload. In such cases, users will prefer to be migrated to another data center or VM with tolerable QoE degradation. To examine the impact of VM migrations incurred by our algorithm, we conduct some extra experiments to study the impact of VM migrations.

Fig. 7 depicts the distribution of the number of VM migrations experienced by a user per hour. Users with a larger group



(a) (b)

Fig. 8. Impact of the parameter α. (a) Mean service cost under various values of the parameter α. (b) Mean user utility under various values of the parameter α.

index have a higher QoE requirements on gaming latency and streaming bitrate. From Fig. 7, we can observe that users with a higher group index experience less VM migrations per hour, which means users who have a higher QoE requirement experience less VM migrations to guarantee their gaming quality. For users in Group 1, 2, and 3, the median number of VM migrations experienced per hour is about 1.3, while the median number of VM migrations experienced by users in Group 5 is close to zero. The reason why users with a smaller group index experience a little larger number of VM migrations is to cut down the service cost with tolerable QoE degradation.

F. Impact of Tunable Parameters

The value of α determines the tradeoff between user utility and service cost. To examine the impacts of different values of α, we conducted further experiments with various values of α. From Fig. 8, we can observe that, when the value of α increases from 0 to 1, the mean user utility and service cost both increase. With a higher value of α, the gaming experience of users can be improved. In the meanwhile, as a penalty, the service cost also increases due to the increasing resource requirements of QoE. When the value of α is higher than 0.3, the impact of α trends to be stable. Besides, we also observed that, with the increase of the parameter V, we can get a higher mean user utility and a lower mean service cost. However, it is at the cost of a much higher interaction delay for users.

Fig. 9 shows the scatter plot for the mean user utility and service cost under various values of the parameter α. Each point in the figure indicates a pair of the mean user utility and the mean service cost under a particular value of α. We can observe that the points mainly cluster in a few small regions, as the impact of α trends to be stable when the value of α is higher than 0.3. And it is easy to see that, by selecting a larger value of V, we can achieve a much smaller service cost and a much larger user utility, but a larger value of V may result in a higher interaction delay.

Fig. 10 depicts the impacts of the parameter V on the mean service cost and user utility. V is a tunable parameter

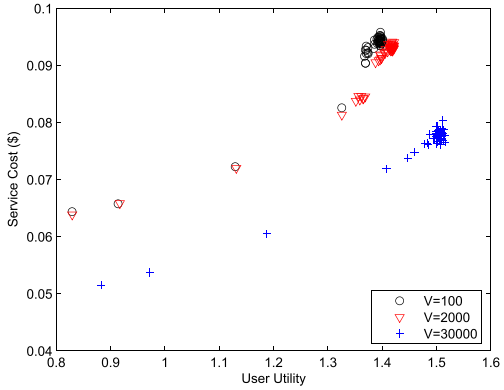
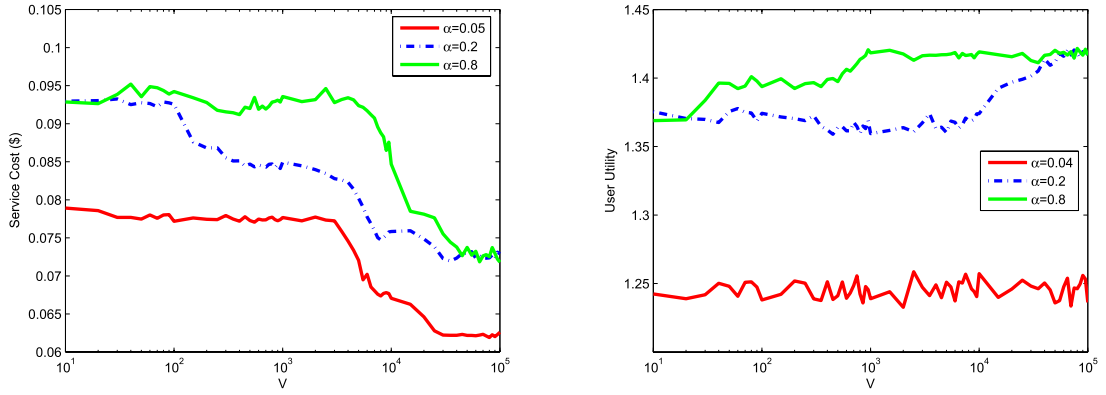


Fig. 9. Scatter plot of mean user utility and service cost under various values of the parameter α.

that determines the tradeoff between interaction delay and penalty cost. Penalty cost is defined as the subtraction of the service cost and the weighted user utility. With a larger value of V, we can further approach the infimum of penalty cost, which means a much lower service cost and a higher user utility. In Fig. 10, when the value of V increases from 100 to 10^5, the mean service cost decreases a lot with a small increase of the mean user utility. However, we should not choose a very large value of V radically, as a very large value of V will significantly increase the interaction delay. Besides, a larger value of α indicates better gaming experience with a larger mean user utility. In the meanwhile, as a penalty, the service cost also increases to ensure the improvement of QoE.

In Fig. 11, we present the scatter plot for the mean user utility and the mean service cost under various values of the parameter V. Each point in the figure corresponds to a pair of the mean user utility and the mean service cost under a particular value of V. We can observe that the mean service cost spreads over a large region, while the mean user utility does not change much under different values of V. We can achieve better gaming quality with a larger value of the parameter α, which is at the cost of a larger mean service cost.



(a) (b)

Fig. 10. Impact of the parameter V. (a) Mean service cost under various values of the parameter V. (b) Mean user utility under various values of the parameter V.

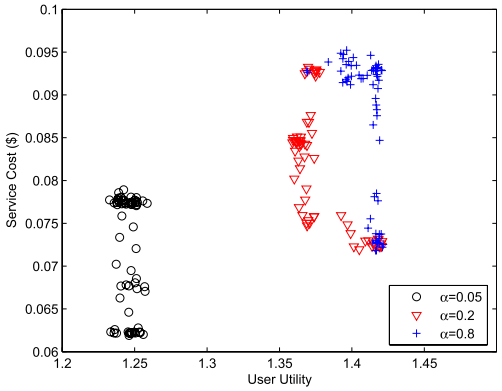


Fig. 11. Scatter plot of mean user utility and service cost under various values of the parameter V.

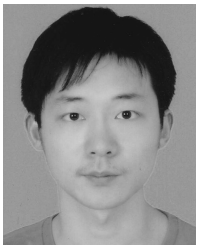
VI. Conclusion

In this paper, we investigated the problem of delivering CACG services. We designed an intelligent online allocation strategy called CACG to achieve adaptive cloud resource allocation in cloud gaming. Our objective is to minimize the service cost for cloud game service providers, while still ensuring good-enough QoE for game players. We formulated the problem as a constrained stochastic optimization problem, and exploited the Lyapunov optimization theory to derive the online strategy. Our proposed algorithm approaches the optimality with an explicitly provable performance upper bound. We also conducted extensive simulations with real traces, and our results show that the proposed algorithm can cut down at least 25% of service cost, while ensuring even better QoE for game players. In the next step, we plan to integrate our algorithm with the real cloud gaming system (e.g., GamingAnywhere) and further explore the impact of other system factors (e.g., gamer behavior) on resource provisioning.

References

[1] (Jul. 2012). *Distribution and Monetization Strategies to Increase Revenues From Cloud Gaming*. [Online]. Available: http://www.cgconfusa.com/report/documents/Content-5minCloudGamingReportHighlights.pdf
[2] (2014). *OnLive*. [Online]. Available: https://www.onlive.com/
[3] (2014). *Gaikai*. [Online]. Available: https://www.gaikai.com/
[4] (2014). *G-Cluster*. [Online]. Available: http://www.g-cluster.com/eng/
[5] StreamMyGame. (2014). [Online]. Available: https://streammygame.com/smg/index.php
[6] C. Chambers, W.-C. Feng, S. Sahu, D. Saha, and D. Brandt, "Characterizing online games," *IEEE/ACM Trans. Netw.*, vol. 18, no. 3, pp. 899–910, Jun. 2010.
[7] P. Quax, A. Beznosyk, W. Vanmontfort, R. Marx, and W. Lamotte, "An evaluation of the impact of game genre on user experience in cloud gaming," in *Proc. IEEE Int. Games Innov. Conf. (IGIC)*, Sep. 2013, pp. 216–221.
[8] M. J. Neely, "Stochastic network optimization with application to communication and queueing systems," *Synthesis Lectures Commun. Netw.*, vol. 3, no. 1, pp. 1–211, 2010.
[9] H. Tian, D. Wu, J. He, Y. Xu, and M. Chen. (2014). "On achieving cost-effective adaptive cloud gaming in geo-distributed data centers," Dept. Comput. Sci., Sun Yat-sen Univ., Guangdong, China, Tech. Rep. TR-CS10. [Online]. Available: http://sist.sysu.edu.cn/~dwu/CACG-TR.pdf
[10] K.-T. Chen, Y.-C. Chang, P.-H. Tseng, C.-Y. Huang, and C.-L. Lei, "Measuring the latency of cloud gaming systems," in *Proc. 19th ACM Int. Conf. Multimedia*, 2011, pp. 1269–1272.
[11] S. Choy, B. Wong, G. Simon, and C. Rosenberg, "The brewing storm in cloud gaming: A measurement study on cloud to end-user latency," in *Proc. 11th Annu. Workshop Netw. Syst. Support Games*, Nov. 2012, pp. 1–6.
[12] M. Manzano, J. A. Hernandez, M. Urueña, and E. Calle, "An empirical study of cloud gaming," in *Proc. 11th Annu. Workshop Netw. Syst. Support Games*, Nov. 2012, pp. 1–2.
[13] M. Manzano, M. Urueña, M. Sužnjević, E. Calle, J. A. Hernández, and M. Matijasevic, "Dissecting the protocol and network traffic of the OnLive cloud gaming platform," *Multimedia Syst.*, vol. 20, no. 5, pp. 451–470, 2014.
[14] Y.-T. Lee, K.-T. Chen, H.-I. Su, and C.-L. Lei, "Are all games equally cloud-gaming-friendly? An electromyographic approach," in *Proc. 11th Annu. Workshop Netw. Syst. Support Games (NetGames)*, Nov. 2012, pp. 1–6.
[15] M. Suznjevic, J. Beyer, L. Skorin-Kapov, S. Moller, and N. Sorsa, "Towards understanding the relationship between game type and network traffic for cloud gaming," in *Proc. IEEE Int. Conf. Multimedia Expo Workshops (ICMEW)*, Jul. 2014, pp. 1–6.
[16] X. Qiu, H. Li, C. Wu, Z. Li, and F. C. M. Lau, "Dynamic scaling of VoD services into hybrid clouds with cost minimization and QoS guarantee," in *Proc. 19th Int. Packet Video Workshop (PV)*, May 2012, pp. 137–142.
[17] F. Wang, J. Liu, and M. Chen, "CALMS: Cloud-assisted live media streaming for globalized demands with time/region diversities," in *Proc. IEEE INFOCOM*, Mar. 2012, pp. 199–207.
[18] Y. Wu, C. Wu, B. Li, X. Qiu, and F. C. M. Lau, "CloudMedia: When cloud on demand meets video on demand," in *Proc. 31st Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jun. 2011, pp. 268–277.
[19] J. He, D. Wu, Y. Zeng, X. Hei, and Y. Wen, "Toward optimal deployment of cloud-assisted video distribution services," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 10, pp. 1717–1728, Oct. 2013.

[20] J. He, Y. Wen, J. Huang, and D. Wu, "On the cost–QoE tradeoff for cloud-based video streaming under Amazon EC2's pricing models," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 4, pp. 669–680, Apr. 2014.
[21] M. Marzolla, S. Ferretti, and G. D'Angelo, "Dynamic resource provisioning for cloud-based gaming infrastructures," *Comput. Entertainment*, vol. 10, no. 1, 2012, Art. ID 4.
[22] T. N. B. Duong, X. Li, R. S. M. Goh, X. Tang, and W. Cai, "QoS-aware revenue-cost optimization for latency-sensitive services in IaaS clouds," in *Proc. IEEE/ACM 16th Int. Symp. Distrib. Simulation Real Time Appl. (DS-RT)*, Oct. 2012, pp. 11–18.
[23] D. Wu, Z. Xue, and J. He, "iCloudAccess: Cost-effective streaming of video games from the cloud with low latency," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 8, pp. 1405–1416, Aug. 2014.
[24] D. Finkel, M. Claypool, S. Jaffe, T. Nguyen, and B. Stephen, "Assignment of games to servers in the OnLive cloud game system," in *Proc. 13th Annu. Workshop Netw. Syst. Support Games (NetGames)*, Dec. 2014, pp. 1–3.
[25] H.-J. Hong, D.-Y. Chen, C.-Y. Huang, K.-T. Chen, and C.-H. Hsu, "Placing virtual machines to optimize cloud gaming experience," *IEEE Trans. Cloud Comput.*, vol. 3, no. 1, pp. 42–53, Jan./Mar. 2014.
[26] (2014). *GamingAnywhere*. [Online]. Available: http://gaminganywhere.org/index.html
[27] K. P. Gummadi, S. Saroiu, and S. D. Gribble, "King: Estimating latency between arbitrary Internet end hosts," in *Proc. 2nd ACM SIGCOMM Workshop Internet Meas.*, 2002, pp. 5–18.
[28] F. Hao, T. V. Lakshman, S. Mukherjee, and H. Song, "Enhancing dynamic cloud-based services using network virtualization," in *Proc. 1st ACM Workshop Virt. Infrastruct. Syst. Archit.*, 2009, pp. 37–44.
[29] V. Jalaparti, M. Caesar, S. Lee, J. Pang, and J. Van der Merwe, "SMOG: A cloud platform for seamless wide area migration of online games," in *Proc. 11th Annu. Workshop Netw. Syst. Support Games*, 2012, pp. 1–6.
[30] T. Wood, K. K. Ramakrishnan, P. Shenoy, and J. Van der Merwe, "CloudNet: Dynamic pooling of cloud resources by live WAN migration of virtual machines," *ACM SIGPLAN Notices*, vol. 46, no. 7, pp. 121–132, 2011.
[31] B. Wong, A. Slivkins, and E. G. Sirer, "Meridian: A lightweight network location service without virtual coordinates," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 35, no. 4, pp. 85–96, 2005.
[32] (2014). *Meridian Project*. [Online]. Available: http://www.cs.cornell.edu/People/egs/meridian/
[33] (2014). *Amazon EC2 Pricing*. [Online]. Available: http://aws.amazon.com/cn/ec2/pricing/
[34] M. Claypool and K. Claypool, "Latency and player actions in online games," *Commun. ACM*, vol. 49, no. 11, pp. 40–45, 2006.
[35] T. N. H. Henderson, "The effects of relative delay in networked games," Ph.D. dissertation, Dept. Comput. Sci., Univ. College London, London, U.K., 2003.
[36] S. Zander, I. Leeder, and G. Armitage, "Achieving fairness in multiplayer network games through automated latency balancing," in *Proc. ACM SIGCHI Int. Conf. Adv. Comput. Entertainment Technol.*, 2005, pp. 117–124.
[37] M. Kihl, A. Aurelius, and C. Lagerstedt, "Analysis of world of warcraft traffic patterns and user behavior," in *Proc. Int. Congr. Ultra Modern Telecommun. Control Syst. Workshops (ICUMT)*, Oct. 2010, pp. 218–223.
[38] F. Chang and W.-C. Feng, "Modeling player session times of on-line games," in *Proc. 2nd Workshop Netw. Syst. Support Games*, 2003, pp. 23–26.
[39] C.-Y. Huang, K.-T. Chen, D.-Y. Chen, H.-J. Hsu, and C.-H. Hsu, "GamingAnywhere: The first open source cloud gaming system," *ACM Trans. Multimedia Comput., Commun., Appl.*, vol. 10, no. 1s, 2014, Art. ID 10.



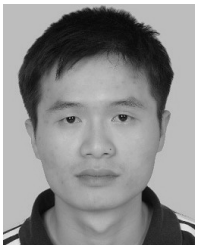
Hao Tian received the B.S. degree from Sun Yat-sen University, Guangzhou, China, in 2013, where he is currently working toward the master's degree with the Department of Computer Science under the supervision of Prof. D. Wu.

His research interests include cloud computing, data center networking, content distribution, and software defined networking.

Di Wu (M'06) received the B.S. degree from University of Science and Technology of China, Hefei, China, in 2000; the M.S. degree from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, in 2003; and the Ph.D. degree in computer science and engineering from the Chinese University of Hong Kong, Hong Kong, in 2007.

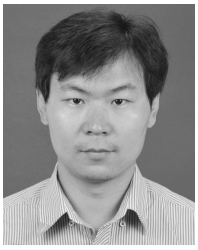
He was a Post-Doctoral Researcher with the Department of Computer Science and Engineering, New York University Polytechnic School of Engineering, Brooklyn, NY, USA, from 2007 to 2009, advised by Prof. K. W. Ross. He is currently an Associate Professor and Associate Department Head of the Department of Computer Science with Sun Yat-sen University, Guangzhou, China. His research interests include multimedia communication, cloud computing, peer-to-peer networking, Internet measurement, and network security.

Dr. Wu was a co-recipient of the IEEE INFOCOM Best Paper Award in 2009. He has served as an Editor of *Journal of Telecommunication Systems* (Springer), *Journal of Communications and Networks*, *Peer-to-Peer Networking and Applications* (Springer), *Security and Communication Networks* (Wiley), and *KSII Transactions on Internet and Information Systems*, and a Guest Editor of IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY of the Special Issue on Visual Computing in the Cloud: Mobile Computing. He serves as the MSIG Chair of the IEEE Communications Society Multimedia Communications Technical Committee from 2014 to 2016. He served as the TPC Co-Chair of the IEEE GLOBECOM Workshop on Cloud Computing Systems, Networks, and Applications in 2014, and the Chair of the CCF Young Computer Scientists and Engineers Forum, Guangzhou, from 2014 to 2015, and is a member of the Council of China Computer Federation.



Jian He received the B.S. and M.S. degrees from Sun Yat-sen University, Guangzhou, China, in 2011 and 2014, respectively. He is currently working toward the Ph.D. degree with the Department of Computer Science, The University of Texas at Austin, Austin, TX, USA.

His research interests include content distribution networks, data center networking, green networking, and network measurement.



Yuedong Xu received the B.S. degree from Anhui University, Hefei, China, the M.S. degree from the Huazhong University of Science and Technology, Wuhan, China, and the Ph.D. degree from The Chinese University of Hong Kong, Hong Kong.

He held a post-doctoral position with INRIA Sophia Antipolis, Valbonne, France, and Universite d'Avignon, Avignon, France, from 2009 to 2012. He is currently a Tenure-Track Associate Professor with the School of Information Science and Technology, Fudan University, Shanghai, China. His current research interests include performance evaluation, control, optimization, and economic analysis of communication networks.

Min Chen (M'08–SM'09) was a Research and Development Director of Confederal Network Inc., Renton, WA, USA, from 2008 to 2009. He was an Assistant Professor with the School of Computer Science and Engineering, Seoul National University (SNU), Seoul, Korea, from 2009 to 2012. He was a Post-Doctoral Fellow with SNU for one and a half years. He was also a Post-Doctoral Fellow with the Department of Electrical and Computer Engineering, University of British Columbia, Vancouver, BC, Canada, for three years. He is currently a Professor with the School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China. He has authored over 180 paper publications, including 85 SCI papers. His current research interests include Internet of Things, big data, machine-to-machine communications, body area networks, e-healthcare, mobile cloud computing, ad hoc cloudlet, cloud-assisted mobile computing, ubiquitous network and services, and multimedia transmission over wireless network.

Mr. Chen was a recipient of the Best Paper Award from the IEEE International Conference on Communications (ICC) in 2012, and the Best Paper Runner-Up Award from QShine in 2008. He serves as an Editor or Associate Editor of *Information Sciences*, *Wireless Communications and Mobile Computing*, *IET Communications*, *IET Networks*, the International Journal of Security and Communication Networks (Wiley), the *Journal of Internet Technology*, *KSII Transactions on Internet and Information Systems*, and the *International Journal of Sensor Networks*. He is a Managing Editor of the *International Journal of Autonomous and Adaptive Communication Systems* and the *International Journal of Advanced Research and Technology*. He is a Guest Editor of the *IEEE Network* and the *IEEE Wireless Communications Magazine*. He was the Co-Chair of the IEEE ICC Communications Theory Symposium in 2012, and the IEEE ICC Wireless Networks Symposium in 2013. He was the General Co-Chair of the IEEE International Conference on Computer and Information Technology in 2012. He is the General Co-Chair of Mobimedia in 2015. He was the General Vice Chair of Tridentcom in 2014. He was a Keynote Speaker for CyberC 2012 and Mobiquitous 2012. He was a TPC Member of the IEEE INFOCOM in 2013 and 2014. He is the Chair of the IEEE Computer Society Special Technical Communities on Big Data.