Virtual Machine Placement Optimization in Mobile Cloud Gaming through QoE-Oriented Resource Competition

Yiwen Han, *Student Member, IEEE,* Dongyu Guo, *Student Member, IEEE,* Wei Cai, *Member, IEEE,* Xiaofei Wang*, *Senior Member, IEEE* and Victor C. M. Leung, *Fellow, IEEE*

ABSTRACT—Cloud gaming is a novel service provisioning paradigm, which hosts video games in the cloud and transmits interactive game streams to game players via the Internet. In such cloud gaming scenarios, the cloud is required to consume tremendous resources for video rendering and streaming, especially when the number of concurrent players reaches a certain level. On the other hand, different game players may have distinct requirements on Quality-of-Experience, such as high video quality, low delay, etc. Under this circumstance, how to satisfy players of different interests by efficiently leveraging cloud resources becomes a major challenge to existing cloud gaming services. In order to meet the overall requirements of players in a cost-effective manner, this work applies game theory to cloud gaming scenarios. It proposes a distributed algorithm to optimize virtual machine (VM) placement in mobile cloud gaming through resource competition. Further, by constructing a potential function, we prove that the resource competition game is a potential game, and the proposed algorithm scales well as the player population increases. We prove theoretically and verify experimentally that, with the proposed distributed VM placement algorithm, players can achieve a mutually satisfying state within a finite number of iterations.

Index Terms—Mobile cloud gaming, game theory, resource optimization

1 Introduction

Mobile gaming is becoming increasingly popular, and it is reported that mobile revenues account for more than 50% of the global gaming market as it reaches $137.9 billion in 2018 [1]. However, for sophisticated games with advanced graphical effects and complicated scenes, state-of-the-art mobile terminals still lack support for the increased storage and computation requirements of contemporary games. To this end, cloud gaming [2] [3] is proposed to help relieve the burden on terminal devices by hosting the game engine in the cloud.

There are two main cloud gaming paradigms [4]: 1) Remote rendering, as shown in Fig. 1, i.e., executing source code on a cloud gaming platform and delivering video frames to devices of players; 2) Local rendering, i.e., the rendering module is implemented in the devices of players while the cloud gaming platform processes the gaming logics and sends a set of generated display instructions to these devices. Specifically, local rendering consumes more resources on the devices of players while remote rendering depends more on Internet performance.

Since the upcoming 5th Generation (5G) mobile networks are aimed to provide high bandwidth and low-latency communications [5], in this paper we focus on the more promising remote rendering paradigm of cloud gaming, rather than local rendering, due to the following reasons. First, local rendering means that the source codes of games themselves need to be modified, which is impractical due to various copyright and cost limitations. Second, local rendering is not feasible to support most of the AAA-level games from other platforms since the graphics capabilities of mobile devices are far from enough. Third, local rendering is designed to eliminate the high burden of real-time video transmission on the network. Nonetheless, the high-bandwidth transmission brought by 5G networks may make local rendering less meaningful in this regard [6].

With respect to the remote rendering paradigm of cloud gaming, commands of players are sent to the cloud while the cloud renders gaming scenes accordingly and streams the corresponding real-time video frames back to the players [7] via backbone and wireless networks. Such a paradigm is a promising solution for enabling players to experience resource-hungry games on light-weight mobile devices such as phones or tablets.

Commercial companies, e.g., OnLive[1], Gaikai[2], G-

- *A preliminary version of the paper appears as 'QoE-Oriented Resource Optimization for Mobile Cloud Gaming: A Potential Game Approach' in ICC 2019. This article has made a significant extension.*
- *Yiwen Han, Dongyu Guo and Xiaofei Wang are with Tianjin Key Laboratory of Advanced Networking, College of Intelligence and Computing, Tianjin University, Tianjin, China. E-mail: {hanyiwen, dongyuguo, xiaofeiwang}@tju.edu.cn.*
- *Wei Cai is with the School of Science and Engineering, The Chinese University of Hong Kong, Shenzhen, China and also with the Shenzhen Institute of Artificial Intelligence and Robotics for Society, Shenzhen, China. E-mail: caiwei@cuhk.edu.cn.*
- *V. C. M. Leung is with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China, and also with the Department of Electrical and Computer Engineering, the University of British Columbia, Vancouver, Canada. E-mail: vleung@ieee.org.*
- *Corresponding author: Xiaofei Wang (xiaofeiwang@tju.edu.cn).*

1. http://www.onlive.com/
2. http://www.gaikai.com/

cluster[3], and Ubitus[4], explored cloud gaming in late 2000's. Nvidia's cloud gaming project GeForce Now is currently in beta trial run in North America and Europe[5]. Moreover, Microsoft launched a dedicated cloud-based gaming environment and its Xbox gaming services with its powerful public cloud, Azure[6]. Before 5G networks are fully commercialized, by taking advantage of the nearest (relative to the player) server among its numerous data centers across the globe, Google unveils its cloud gaming solution, Stadia, which is capable of streaming video games with up to 4K resolution at 60 frames per second[7].

To support elastic cloud computing, hardware virtualization is widely employed to allows multiple virtual machines (VMs) to share one physical machine [8]. Nevertheless, since cloud gaming is resource-intensive and delay-sensitive, general solutions for VM management in cloud computing cannot be directly applied [9]. In particular, VM placement [9] in cloud gaming scenarios are not addressed well. Specifically, when a player requests a cloud game service from a service provider, its cloud data center needs to allocate hardware resources for this player, configure the VM, and select an appropriate physical machine to place the VM. However, for commercial scenes with massive players, if players with various experience requirements and networking conditions are not scheduled properly [10], the cloud resources cannot be sufficiently utilized, and the Quality-of-Experience (QoE) is downgraded as well. Therefore, it becomes a challenge to schedule the VM placement while guaranteeing the overall QoE of players with different interests.

In this paper, we employ game theory to solve this challenge and to help service providers reduce maintenance costs. The adoption of game theory is natural due to the decentralized [11] characteristic of the cloud gaming architecture: players are competing for the limited cloud resources and finally are self-organized into a mutually satisfactory state. Moreover, optimization based on game theory can help the cloud ease the burden of complex centralized management for resource allocation, such as collecting players' network information and dealing with a large scale of data. Besides, as players may want to play different games and get a better experience in terms of QoE, game theory can be utilized to analyze the resource competition among different players with various targeting games.

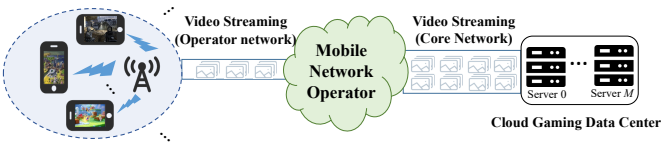


Fig. 1: Remote rendering for cloud gaming.

Hereinafter we formulate the multi-player mobile cloud gaming problem as a distributed optimization problem with respect to multi-player resource competition. The major contributions of our work are summarized below:

- We propose a mobile cloud gaming architecture along with a resource optimization problem, i.e., VM placement optimization, and show that it is NP-hard to find optimal solutions for this problem.
- By leveraging potential game theory, we study the intrinsic property of the resource competition game, and show that this competition game always possesses a Nash Equilibrium (NE). Further, to solve VM placement optimization, we design a distributed algorithm with a complexity of $\mathcal{O}(M \log M)$, where M is the number of physical servers in the cloud.
- Besides, we quantify the efficiency ratio of our solution compared to the optimal solution by both theoretical analyses and experiments. Experimental results corroborate that the proposed algorithm can achieve efficient performance, $60\%-75\%$ better compared with other strategies, and scales well with the number of players.

The remainder of this paper is organized as follows. Related works are discussed in Section 2 and Section 3 elaborates the proposed cloud gaming architecture. Then, a multi-player resource competition problem is introduced in Section 4. Performance analysis is derived in Section 5 along with experiment results given in Section 6. Finally, Section 7 concludes the paper.

2 Related Work

2.1 Cloud Gaming System

There have been several achievements for building cloud gaming systems, especially in academia. Consisting of a distributed service platform, a distributed rendering system, and an encoding/streaming system, a cloud-based gaming service platform [12] is presented to support isolated audio/video capturing and multiple types of clients. Based on the RemoteFX extension of Windows remote desktop protocol, Depasquale et al. [13] implements a cloud computing platform for cloud gaming to service a number of players through the use of virtualization. Later, C.-Y. Huang et al. [7] propose an open-source, extensible, portable, and reconfigurable cloud gaming system, i.e., GamingAnywhere. It is the first platform for researchers, game developers, service providers, and gaming players to set up their own cloud gaming testbeds. Based on GamingAnywhere, Q. Hou et al. [14] further improve the encoding and the multi-client concurrent access in the server of cloud gaming. They build concurrent servers based on NVIDIA GRID GPU to reduce the delay in bandwidth-limited scenarios and to improve the efficiency of multi-client concurrent access. The project AdaPtive HFR vIdeo Streaming (APHIS) [15] is a novel transmission scheduling framework for mobile cloud gaming applications. APHIS can dynamically optimize the quality of game video streaming by adjusting its video traffic load and forward error correction (FEC) coding.

2.2 Virtual Machine Placement

Using virtualization can reduce the cost of cloud service providers [8]. From the VM placement perspective, researchers of different interests are exploring how to efficiently utilize the cloud architecture to facilitate better cloud

3. http://www.gcluster.com/eng
4. http://www.ubitus.net/
5. https://www.nvidia.com/en-gb/geforce/products/geforce-now/mac-pc/
6. https://azure.microsoft.com/en-us/solutions/gaming/
7. https://store.google.com/magazine/stadia

gaming services. For example, the framework proposed in [16] optimizes the interference between VMs (and gaming players) by analyzing the limitation of application resource requirements. Meanwhile, the results in [17] show that cloud gaming can be optimized by achieving network awareness, and it is possible to improve cloud gaming performance while reducing costs through scheduling the most optimal wireless link and cloud server for each game session. On the other hand, H. Hong et al. propose a novel QoE-aware VM placement strategy for cloud gaming [18] [19]. Further, based on the prediction of the end times of game sessions, Y. Li et al. propose an efficient request dispatching algorithm to assign gaming requests to the cloud servers in a cloud gaming system [20]. Compared to the conventional first-fit and best-fit placement strategies, the strategy proposed in [20] can better help service providers reduce the resource waste of the cloud servers.

For scenarios involving multiple game players, M. Marzolla et al. [21] improves the efficiency of resource provisioning for massively multiplayer online games (MMOG), by employing greedy heuristics to allocate the minimum number of computing nodes required to meet the service needs. Besides, to save the cost of the cloud, D. Finkel et al. model the distribution of games to servers and players requesting games by examining actual OnLive server logs, and construct improved game distribution strategies based on a hill-climbing algorithm [22].

3 System Model

We list related symbols and their definitions in Table 1 for the readability of the rest of the paper.

3.1 Architecture of Mobile Cloud Gaming

The proposed mobile cloud gaming architecture is presented in Fig. 2. First, each player chooses a game of interest to play and sends the game requirements to the cloud. Second, the available gaming resolutions of each player is determined within the associated BS according to the dynamic wireless environment. Then, the dispatcher server at the cloud collects all game requests from all BSs at one time, including the available gaming resolutions of each player. For each player, based on the requirements, the dispatcher server allocates a specific physical server along with a customized VM on it. Next, the dispatcher server admits the request, and copies related files of the target game to the corresponding VM. Finally, the dispatcher server exposes the IP address of that VM to the player, while that VM is determined as their dedicate VM for the game session.

In the mobile cloud gaming scenario, we assume that there are N game players $\mathcal{N} = \{1, 2, ..., N\}$ and M physical servers $\mathcal{M} = \{1, 2, ..., M\}$ performing intensive computation in the cloud data center. The server $\psi = 1 \in \mathcal{M}$ is the initial server. As we state in Section 3.5, which means that the initial strategy of the dispatch server is first admitting all requests from all players and associating them with the initial server ψ. For mobile cloud gaming, the quality of wireless environments determines the actual gaming resolution players can choose. Thus, we take the competition model of radio resources, commonly adopted in mobile edge computing scenarios [23], [24]. These models are built for future 5G edge computing scenarios [25], [26] and they fit in with our vision about mobile cloud gaming. Based on these pioneer works, the available gaming resolutions for each player can be determined within the associated BS according to wireless environments.

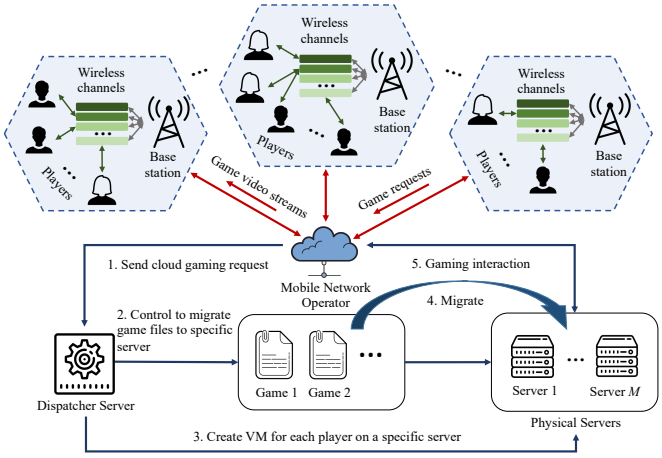


Fig. 2: The proposed architecture for mobile cloud gaming.

We consider a comprehensive scenario including B Base Stations (BSs) $\mathcal{B} = \{1, 2, ..., B\}$, within which each player can enjoy mobile gaming via a wireless link. The game players within the coverage area of BS b is denoted as $\mathcal{N}_b$ and the total number of players covered by all BSs is equal to the number of players (i.e., N) that need to be served by the cloud, namely $\mathcal{N} = \{\mathcal{N}_b, b \in \mathcal{B}\}$. In this case, players of mobile cloud gaming are contenders competing for finite wireless resources, and such a problem is well investigated in [23], [24]. Specifically, we consider that BS b provisions C interference-free wireless channels $\mathcal{C} = \{1, 2, ..., C\}$, on which the gaming video streams are transmitted, for $N_b = |\mathcal{N}_b|$ mobile devices (belonging to the players) and each channel is allocated with ω Hz bandwidth.

For the wireless resource competition game, we denote $c_{b,n} \in \mathcal{C}$ as the allocated channel for the player n within BS b. On the cloud side, $a_n \in \mathcal{M}$ is denoted as the associated physical server, on which the VM for hosting the gaming session of player n is created. Thus, given the decision profiles $\boldsymbol{c} = \{c_{b,n}, b \in \mathcal{B}, n \in \mathcal{N}\}$ for channel allocation and the decision profiles $\boldsymbol{a} = (a_1, a_2, ..., a_N)$ for physical server association of all players, the whole problem of optimizing gaming experiences for all players is essentially a combinatorial optimization problem.

3.2 Delay Model

Since game instances are created and destroyed dynamically on top of the base VMs, before the game starts, there exists a clone delay, which is a part of the service initialization delay [27], for each player n to wait for copying related files. Within the cloud data center, we denote the speed of the hard disk for writing game files as W for simplicity. Note that the clone delay introduced here can also be extended to the whole service delay by incorporating the computing delay and data transmission delay without violating the general model. Therefore, if one player chooses to play a

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2020.3002023, IEEE Transactions on Cloud Computing

TABLE 1: Table of Notations

Param.	Def.	Param.	Def.	Param.	Def.
$\mathcal{A}_n$	Available set of association strategies	$\mathcal{F}_n$	Gaming FPS of player n	$\mathcal{V}_n$	Expected experience level of player n with respect to gaming resolution
a_n	Associated physical server of player n	$I\{\cdot\}$	Indicates whether the event in $\{\cdot\}$ is true or not	W	Writing speed of hard disks
$\boldsymbol{a}$	Association decisions of all players	$\mathcal{L}_n$	Gaming experience loss of player n	Γ	Multi-player resource competition game
$\boldsymbol{a}_{-n}$	Association decisions of other players except player n	m^*	The server hosting the most players given by centralized optimum	γ	The set of NE of the multi-player resource competition game
$\boldsymbol{a}^*$, a_n^*	Centralized optimal solution that maximizes the number of beneficial offloading players	$\mathcal{M}$	Set of physical servers in the cloud	κ	Number of physical CPU cores
$\tilde{\boldsymbol{a}}$, $\tilde{a}_n$	Arbitrary NE of the game	M	Number of physical servers	μ	Memory size of RAM
$\overline{\boldsymbol{a}}$, $\overline{a}_n$	Centralized optimum that minimize the overall gaming experience loss	n	Player n	Φ	Constructed potential function
a_n'	Beneficial offloading server of player n	$\mathcal{N}$	Set of players	ψ	Initial associated server
$\mathcal{B}$	Set of BSs	$\mathcal{N}_b$	Set of Players within BS b	$\overline{\boldsymbol{\psi}}$	Servers except ψ
B	Number of BSs	P	VM initialization time	ω	Bandwidth of each channel
$\beta_{b,n}$	Transmission rate of the player n within BS b	$p_{b,n}$	Transmitting power of the player n communicating with BS b	ϖ_0	Noise power in wireless environments
$\mathcal{C}$	Set of interference-free wireless channels	$\mathcal{R}_n$	Number of resource units to support perfect gaming	ω_1, ω_2, ω_3	Parameters for modeling the gaming FPS
C	Number of wireless channels	S_n	File size of the game player n requests	Δ_n	Beneficial server set of player n
$c_{b,n}$	Allocated channel for the player n within BS b	$\mathcal{T}_n$	Threshold for player n choosing beneficial offloading	λ_1, λ_2, λ_3	Impact factors effecting the gaming experience
$\boldsymbol{c}$	Decision profiles of all players	$\mathcal{V}_n'$	Actual experience level of player n with respect to gaming resolution	ρ_m	Workload on server m
$\mathcal{D}_n$	Total delay of initialization				

game with file size S_n, considering the VM initialization time P in [28], the total delay can be represented by

$$\mathcal{D}_n = \frac{S_n}{W} + P. \tag{1}$$

3.3 Gaming Resolution Model

As shown in Fig. 1, game video streams generated by the cloud is first transmitted to mobile network operators via the core network, and then delivered to mobile devices of players via respective mobile networks. In addition, each BS admits the request of each player and determines available gaming resolutions, according to the results of the wireless channel assignment. We denote $\mathcal{V}_n'$ and $\mathcal{V}_n$ as the quantified experience level of the actual and the requested (expected) gaming resolution of player n, respectively. In general, affected by imperfect wireless environments, the requested gaming resolution cannot always be satisfied due to the fluctuating transmission rate $\beta_{b,n}$ in the wireless network [29] given by

$$\beta_{b,n}(\boldsymbol{c}) = \omega \log_2\left(1 + \frac{p_{b,n}}{\varpi_0 + \sum\limits_{i \in \mathcal{N}_b \setminus \{n\}: c_{b,i} = c_{b,n}} p_{b,i}}\right), \tag{2}$$

where $p_{b,n}$ is the transmitting power of player n communicating with BS b and ϖ_0 is the background noise. Besides, the actual gaming resolution, indicated by the experience level $\mathcal{V}_n'$ with respect to gaming resolution, also relates to the gaming FPS (Frame Per Second). Since 30 FPS is the minimum requirement for smooth gaming, we set 30 as the base FPS requirement, while the actual gaming FPS may be higher to deliver a better gaming experience. In addition, the actual gaming resolution of player n is bounded by the available wireless transmission rate, as given in Table 2. For example, if a player requests 1080p gaming while the wireless network environment can only provide them with a $5,000$ Kbps transmission rate, the actual gaming resolution for them is 720p. Note that the data presented in Table 2 were obtained from our testbed using H.264 encoding, while the data evaluated with other customized platforms and settings may differ. Nonetheless, this does not affect the applicability of our proposed algorithm in Section 4.4.

TABLE 2: Minimum Transmission Rates for Different Gaming Resolutions at 30 FPS

Resolutions	720p	1080p	1440p	2160p
Experience Level	1	2	3	4
Bitrates/Kbps	≥4,500	≥9,500	≥17,300	≥33,000

Considering that the transmission rate on backbone networks is much better than the wireless transmission rate, the available resolution of mobile cloud gaming is mainly affected by the latter one. Therefore, at each BS (e.g., BS b), player n is always willing to compete for a higher transmission rate $\beta_{b,n}$. Under the circumstances, such a competition game for transmission rate can be solved by the proposed Gauss-Seidel-like method in [23], and it is proved to have a pure NE. After that, by considering both the expectation of players and the transmission rate in practical

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2020.3002023, IEEE Transactions on Cloud Computing

wireless environments, the gaming resolution of each player is determined and then sent to the cloud as the actual gaming resolution.

3.4 Experienced FPS Model

The gaming FPS experienced by players is also a pivotal experience metric. In the proposed architecture, all physical servers in the cloud are equipped with the same configuration, i.e., CPU with κ cores, graphics cards with μ GB memory in total, and φ GB RAM. The gaming FPS is determined mainly by the comprehensive CPU, GPU, and RAM usages in each physical server. When the server is dedicated to only one player, this player can certainly acquire powerful capabilities of game rendering and hence experience fluent gaming. However, in general, each physical server hosting multiple VMs is serving multiple players at the same time, which may cause resource contentions. In particular, if there are too many players associated with the same physical server, the gaming experiences of all these players will drop dramatically. Hereinafter, we take the CPU usage as a common impact factor for illustrating the resource competition within one physical server.

For a physical server, specific interference and performance degradation occur when the number of CPU cores allocated to all VMs exceeds its resource limit [19]. It is similar to the concept of parallel execution of multiple threads in the operating system. Specifically, when the number of created threads is far more than the number of threads that the CPU can offer, the competition for resources and the switching of CPU contexts may lead to worse overall efficiency. Back to cloud gaming, in [19], the relationship between the number of VMs and the gaming FPS is investigated. The experimental results show that such a relationship can be approximated by a Sigmoid function (having domain of all real numbers, with return value monotonically increasing) [30] in the form of $s\left(x\right)=\delta_1/\left(1+\mathrm{e}^{-\delta_2 x+\delta_3}\right)$, where δ_1, δ_2 and δ_3 are parameters to model the relationship between the variable x and the dependent variable s.

Similarly, if we deem each VM as a resource unit allocated with only one virtual CPU core, the relationship between the available resource units and the gaming FPS can also be naturally established as done in [19]. Therefore, with denoting $\mathcal{R}_n$ as the number of CPU cores required for player n to experience perfect gaming, the gaming FPS can be quantified as

$$\mathcal{F}_n\left(\boldsymbol{a}\right)=\frac{\omega_1}{1+\mathrm{e}^{\omega_2\left(\sum\limits_{i\in\mathcal{N}\setminus\{n\}:a_i=a_n}\mathcal{R}_i+\mathcal{R}_n\right)/\kappa+\omega_3}}\mathcal{R}_n, \quad (3)$$

where ω_1, ω_2 and ω_3 are parameters for modeling the relationship between the gaming FPS and the resource requirements of all players within the same physical server. For each game used to evaluate the performance of our method, we run it 30 times for each VM configuration, record all data of gaming FPS, and based on (3), perform regression analysis on these data to acquire the specific values of ω_1, ω_2 and ω_3.

3.5 Quantifying Gaming Experience Loss

For clear description, the initial associated server and the other available servers are represented by $\psi=1$ and $\overline{\psi}=\{2,...,M\}$, respectively. The initial strategy is first admitting all requests from all players and associating them with an initial server $\psi=1$. Notice that associating with ψ is only for quantifying the gaming experience of players rather than to directly start gaming sessions on it.

TABLE 3: Default Impact Factors for Gaming Experience

Game (File Size)	Type	λ_1	λ_2	λ_3
Need For Speed: Rivals (14.2 GB)	RAC	0.3	0.5	0.2
Fallout 4 (25.8 GB)	ARPG	0.2	0.5	0.3
The Witcher 3: Wild Hunt (38.9 GB)	ARPG	0.2	0.3	0.5
Dragon Quest XI (13.9 GB)	RPG	0.5	0.2	0.3
World of Warcraft (79 GB)	MMORPG	0.5	0.3	0.2
Civilization VI (12.6 GB)	SLG	0.1	0.1	0.8
Total War: Rome II (23.9 GB)	SLG/RTS	0.1	0.4	0.5
StarCraft II (23.2 GB)	RTS	0.4	0.3	0.3
The Sims 4 (15.3 GB)	SIM	0.4	0.4	0.2
Minecraft (1.89 GB)	SIM	0.3	0.5	0.2
COD 14: WWII (68.8 GB)	FPS	0.3	0.4	0.3
NBA 2K19 (67.56 GB)	SPG	0.5	0.3	0.2
FIFA 19 (32.83 GB)	SPG	0.5	0.2	0.3
Street Fighter 5 (17.2 GB)	FTG	0.4	0.4	0.2
GTA 5 (74.2 GB)	RPG/FPS/RAC	0.3	0.4	0.3

By combining the delay in (1), the actual gaming resolution $\mathcal{V}_n'$ and the FPS in (3), we evaluate the overall gaming experience of player n in terms of the gaming experience loss $\mathcal{L}_n\left(\boldsymbol{a}\right)$. Each player's objective is to minimize their gaming experience loss $\{\mathcal{L}_n\left(\boldsymbol{a}\right), n\in\mathcal{N}\}$.

$$\mathcal{L}_n(\boldsymbol{a})=\lambda_1\mathcal{D}_n-\lambda_2\mathcal{F}_n(\boldsymbol{a})-\lambda_3\mathcal{V}_n' \quad (4)$$

Herein, $\{\lambda_i, i=1,2,3\}$, satisfying $0\leq\lambda_i\leq 1$ and $\sum_{i=1}^{3}\lambda_i=1$, are impact factors affecting the comprehensive gaming experience of players and are mainly depended on the game type. For our game library, as presented in Table 3, we determine default values, inspired by [28], for these impact factors. For instance, with respect to Civilization VI, players usually do not need very high gaming FPS and ultra-low delays but want a finer gaming resolution to improve the sense of substitution. Accordingly, we set $\lambda_1=0.1$, $\lambda_2=0.1$ and $\lambda_3=0.8$. Note that these settings are for general use, while practical values can be customized for each player. It can be inferred that the experience loss $\mathcal{L}_n\left(\boldsymbol{a}\right)$, conceived by player n, is affected by factors including the hardware requirement of the chosen game, the hardware burden of the associated physical server, the wireless environments and the features of this game.

4 Multi-player Competition Game

4.1 Solving Optimization Problems is NP-Hard

We first consider an optimization problem in terms of the number of beneficial offloading players. Another important metric, the overall experience loss of players, will be discussed later. Maximizing the number of beneficial offloading players is the optimization objective of the dispatch server, i.e., achieving better load balancing for physical servers

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2020.3002023, IEEE Transactions on Cloud Computing

in the cloud. Therefore, the problem can be formulated mathematically as follows.

$$\begin{aligned} &\max_{\boldsymbol{a}} \sum_{n\in\mathcal{N}} I_{\{a_n\in\overline{\psi}\}} \\ &\text{s.t. } \mathcal{L}_n\left(\psi,\boldsymbol{a}_{-n}\right) \geq \mathcal{L}_n\left(\overline{\psi},\boldsymbol{a}_{-n}\right), \forall n\in\mathcal{N}, \\ &\quad a_n\in\{1,\ldots,M\}, \forall n\in\mathcal{N}. \end{aligned} \tag{5}$$

In (5), $\boldsymbol{a}_{-n} = (a_1,\ldots,a_{n-1},a_{n+1},\ldots,a_N)$ represents the association decisions by all other players except player n. $\mathcal{L}_n\left(\psi,\boldsymbol{a}_{-n}\right)$ or $\mathcal{L}_n\left(\overline{\psi},\boldsymbol{a}_{-n}\right)$ correspond to the gaming experience loss when the player n is associated with the initial server ψ or associated with one of the other servers $\overline{\psi}$ (i.e., $a_n\in\overline{\psi}$), respectively. $I_{\{\cdot\}}\in\{0,1\}$ indicates whether the event in $\{\cdot\}$ is true ($I=1$) or not ($I=0$). Unfortunately, solving the problem of maximizing the number of beneficial offloading players can be extremely challenging.

Theorem 1. *The problem in (5) that finds the solution, i.e.,* $\arg\max_{\boldsymbol{a}} \sum_{n\in\mathcal{N}} I_{\{a_n\in\overline{\psi}\}}$*, to maximize the number of beneficial offloading players is NP-hard.*

Proof. First, we introduce the maximum cardinality bin packing problem [31] as formulated in (6): given N items with each b_i is one size for $i\in\mathcal{N}$ and M bins of identical capacity $\mathcal{T}$, the objective is to assign a maximum number of items to the fixed number of bins without violating the capacity constraint $\mathcal{T}$.

$$\begin{aligned} &\max \sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij} \\ &\text{s.t. } \sum_{i=1}^{N} b_i y_{ij} \leq \mathcal{T}, \forall j\in\mathcal{M}, \\ &\quad \sum_{j=1}^{M} y_{nj}\in\{0,1\}, \forall i\in\mathcal{N}, \\ &\quad y_{ij}\in\{0,1\}, \forall i\in\mathcal{N}, j\in\mathcal{M}. \end{aligned} \tag{6}$$

For our problem (5), we can infer that a player can achieve beneficial offloading only if their associated physical server is not overloaded, i.e., $\sum_{i\in\mathcal{N}\setminus\{n\}:\ a_i=a_n}\mathcal{R}_i \leq \mathcal{T}_n$, where $\mathcal{T}_n$ is the overload threshold with respect to CPU cores. In view of this fact, a special case of problem (5) can be transformed to the maximum cardinality bin packing problem as follows. Specifically, the game players and the available physical servers in our problem can be regarded as the items and the bins in the maximum cardinality bin packing problem, respectively. After that, the size of an item n and the capacity constraint of each bin can be given as $b_n=\mathcal{R}_n$ and $\mathcal{T}=\mathcal{T}_n+\mathcal{R}_n$, respectively. By this means, as long as a player n on their associated physical server a_n achieves the beneficial offloading, for an item n, the total sizes of the items on its assigned bin a_n shall not violate the constraint $\mathcal{T}$, since $\sum_{i\in\mathcal{N}\setminus\{n\}:\ a_i=a_n}\mathcal{R}_i \leq \mathcal{T}_n$ implies $\sum_{i=1}^{N} b_i x_{i,a_n} = \sum_{i\in\mathcal{N}\setminus\{n\}:\ a_i=a_n}\mathcal{R}_i + \mathcal{R}_n \leq \mathcal{T}$.

Therefore, the algorithm capable of finding the maximum number of beneficial offloading players can address the maximum cardinality bin packing problem as well. Since the maximum cardinality bin packing problem is proved NP-hard [31], the problem in (5) is also NP-hard. □

Besides the number of beneficial offloading players, another important metric in multi-player cloud gaming is the overall experience loss of all players, aimed to be minimized, i.e.,

$$\begin{aligned} &\min_{\boldsymbol{a}} \sum_{n=1}^{N} \mathcal{L}_n(\boldsymbol{a}) \\ &\text{s.t. } a_n\in\{1,...,M\}, \forall n\in\mathcal{N}. \end{aligned} \tag{7}$$

Theorem 2. *The problem in (7) that finds the optimum, i.e.,* $\arg\min_{\boldsymbol{a}} \sum_{n=1}^{N} \mathcal{L}_n(\boldsymbol{a})$*, to minimize the overall experience loss of players is NP-hard.*

Proof. Consider a special case where game players only care about the gaming delay and FPS, the Uncapacitated Facility Location (UFL) problem [32] can be transformed to the problem in (7). Specifically, N players and available M servers are mapped to clients and potential sites for locating facilities, respectively. The UFL problem considers the placement of multiple facilities to minimize opening and transportation costs for a set of sites. The UFL problem can be reduced to the set cover problem, which is NP-complete, and is hence NP-hard [33]. Thus, as an extension of the UFL problem, the player-server association problem (7) in our case is also NP-hard. □

Note that the centralized optimization problem of minimizing the overall experience loss of players involves a combinatorial optimization over the multi-dimensional discrete space $\{1,\ldots,M\}^N$. Particularly, for large-scale cloud gaming scenarios, if the optimization algorithm possesses high time complexity, it is impractical to be employed. Therefore, in the following, a potential game-based algorithm is proposed for optimizing the two objectives mentioned above.

4.2 Game Formulation

When acknowledging load conditions of all physical servers, player n starts to find the best association server. In other words, in order to ease the burden on the initial server while improving the cloud gaming experience, player n can keep associating with the initial server ψ, or choose to offload to another server out of $\overline{\psi}$. Note that the initialization server ψ can be dynamically changed instead of fixed. In our experiments, we periodically select the server with the heaviest load as the initial server.

According to the experience loss model in Section 3, it can be found that decisions on associating appropriate servers, taken by players, are tightly coupled. If too many players choose to associate with the same physical server at one time, the interference among VMs on it will bring about significant experience losses $\sum_{i=1}^{N}\mathcal{L}_i\left(\boldsymbol{a}\right)$ in total.

The dispatch server should be capable of offloading the burden from the initial server ψ to other servers $\overline{\psi}$. Therefore, the concept of beneficial offloading, indicating that a player chooses a server out of $\overline{\psi}$ and hence experiences better gaming than associating with ψ, is introduced as

$$\mathcal{L}_n\left(\psi,\boldsymbol{a}_{-n}\right) \geq \mathcal{L}_n\left(\overline{\psi},\boldsymbol{a}_{-n}\right). \tag{8}$$

Consequently, each player would like to minimize the experience loss, i.e.,

$$\min_{a_n\in\mathcal{A}_n} \mathcal{L}_n\left(a_n,\boldsymbol{a}_{-n}\right), \forall n\in\mathcal{N}, \tag{9}$$

where $\mathcal{A}_n \subset \mathcal{M}$ is the available set of association strategies for player n. Hence, the competition game above can be formulated as

$$\Gamma = \left(\mathcal{N}, \{\mathcal{A}_n\}_{n\in\mathcal{N}}, \{\mathcal{L}_n\}_{n\in\mathcal{N}}\right), \tag{10}$$

where $\mathcal{N}$ is the set of players, $\mathcal{A}_n$ is the set of strategies for player n, experience loss $\mathcal{L}_n$ is the target to be minimized by player n, and Γ represents the strategic game, called multi-player resource competition game in the sequel. Next, we introduce the concept of NE to address this resource competition problem.

Definition 1. *A strategy profile $\boldsymbol{a}^* = (a_1^*, ..., a_n^*)$ for the multi-player resource competition game is a NE if at the equilibrium $\boldsymbol{a}^*$, no players can further reduce their experience loss by changing their associated server unilaterally, i.e.,*

$$\mathcal{L}_n\left(a_n^*, \boldsymbol{a}_{-n}^*\right) \leq \mathcal{L}_n\left(a_n, \boldsymbol{a}_{-n}^*\right),\ \forall a_n \in \mathcal{A}_n,\ n \in \mathcal{N}\ . \tag{11}$$

With the concept of NE, a corollary can be deduced:

Corollary 1. *If player n at a NE chooses to associate with a server in $\overline{\psi}$ (i.e., $a_n^* \in \{2, ..., M\}$), then player n must be a beneficial offloading player.*

Proof. It can be proved by contradiction as follows. If player n at a NE chooses a server in $\overline{\psi}$ and this server is not a beneficial offloading server, according to the definition of beneficial offloading, player n can easily keep associating with ψ to reduce their experience loss and reach a new NE. Obviously, such new equilibrium strategy $a_n^* = 1$ contradicts with the fact that $a_n^* \in \{2, ..., M\}$. □

4.3 Discussions on Potential Game-based Optimization

To proceed, we introduce potential game [34] to study and to prove the presence of NE in the multi-player resource competition game.

Definition 2. *If there exists a potential function $\Phi(\boldsymbol{a})$, for every $n \in \mathcal{N}$, $\boldsymbol{a}_{-n} \in \prod_{i\neq n}\mathcal{A}_i$, and $a_n, a_n' \in \mathcal{A}_n$, when*

$$\mathcal{L}_n(a_n', \boldsymbol{a}_{-n}) \leq \mathcal{L}_n(a_n, \boldsymbol{a}_{-n}) \tag{12}$$

holds, we can always have

$$\Phi(a_n', \boldsymbol{a}_{-n}) \leq \Phi(a_n, \boldsymbol{a}_{-n}). \tag{13}$$

Then, the corresponding game is called a potential game and can always reach a NE. Moreover, it possesses the finite improvement property, i.e., any asynchronous process towards a better strategy must be finite and leads to a NE [34].

That is, in the process of the potential game, no more than one player updates the strategy of server association to reduce experience loss $\mathcal{L}_n$ at any given time, until a NE is reached. We next show the multi-player competition game is a potential game.

Lemma 1. *Given a server association profile $\boldsymbol{a}$, if the load on a server (in $\overline{\psi}$) selected by player n satisfies $\theta_n(a_n \in \overline{\psi}, \boldsymbol{a}_{-n}) = \sum_{i\in\mathcal{N}\setminus\{n\}:a_i=a_n}\mathcal{R}_i \leq \mathcal{T}_n$, player n achieves beneficial offloading, with the threshold*

$$\mathcal{T}_n = \sum_{i\in\mathcal{N}\setminus\{n\}:a_i=\psi}\mathcal{R}_i\ . \tag{14}$$

Proof. For player n, the file size S_n and the hardware requirement $\mathcal{R}_n$ of the chosen game will not change within a certain game session. According to the definition of beneficial offloading, we have $\mathcal{L}_n\left(\psi, \boldsymbol{a}_{-n}\right) \geq \mathcal{L}_n\left(\overline{\psi}, \boldsymbol{a}_{-n}\right)$, which means that

$$\sum_{i\in\mathcal{N}\setminus\{n\}:a_i=a_n}\mathcal{R}_i + \mathcal{R}_n \leq \sum_{i\in\mathcal{N}\setminus\{n\}:a_i=\psi}\mathcal{R}_i + \mathcal{R}_n, \forall a_n \in \overline{\psi}. \tag{15}$$

This implies that the threshold $\mathcal{T}_n$ for player n choosing beneficial offloading from ψ to $\overline{\psi}$ should meet

$$\sum_{i\in\mathcal{N}\setminus\{n\}:a_i=a_n}\mathcal{R}_i \leq \sum_{i\in\mathcal{N}\setminus\{n\}:a_i=\psi}\mathcal{R}_i = \mathcal{T}_n, \forall a_n \in \overline{\psi}. \tag{16}$$

□

According to Lemma 1, it is beneficial for a player to offload the resource burden from server ψ to a server in $\overline{\psi}$, when the workloads of some servers $\overline{\psi}$, represented by $\mathcal{T}_n$, are not as high as that of the initial ψ. Specifically, if there are already numerous players occupying the initial server ψ, the player n can seek for better gaming experience and offload the gaming request to another server in $\overline{\psi}$. In this way, the burden on the initial server ψ is alleviated while the good gaming experiences of related players are maintained. Based on Lemma 1, we show that the multi-player resource game is a potential game by constructing a potential function as

$$\begin{aligned}\Phi\left(\boldsymbol{a}\right) =& \frac{1}{2}\sum_{i=1}^{N}\sum_{j\neq i}\mathcal{R}_i\mathcal{R}_j I_{\{a_i=a_j\}}I_{\{a_i\in\overline{\psi}\}} \\ &+ \sum_{i=1}^{N}\mathcal{R}_i\mathcal{T}_i I_{\{a_i=\psi\}}\ .\end{aligned} \tag{17}$$

Theorem 3. *The competition game $\Gamma = \left(\mathcal{N}, \{\mathcal{A}_n\}_{n\in\mathcal{N}}, \{\mathcal{L}_n\}_{n\in\mathcal{N}}\right)$ with the potential function Φ given in (17) in cloud gaming is indeed a potential game possessing the finite improvement property, and it can always converge to a NE in a limited number of iterations.*

Proof. We assume that the game is a potential game and its potential function is $\Phi(\boldsymbol{a})$. Player $n \in \mathcal{N}$ updates their decision from a_n to a_n' and obtains less experience loss, i.e., $\mathcal{L}_n(a_n, \boldsymbol{a}_{-n}) > \mathcal{L}_n(a_n', \boldsymbol{a}_{-n})$. According to definition 2, we will show that this will also lead to a decrease in potential function $\Phi(\boldsymbol{a})$, i.e., $\Phi(a_n, \boldsymbol{a}_{-n}) > \Phi(a_n', \boldsymbol{a}_{-n})$. There are four cases in total: 1) $a_n \in \overline{\psi}$, $a_n' \in \overline{\psi}$ and $a_n \neq a_n'$; 2) $a_n \in \overline{\psi}$ and $a_n' = \psi$; 3) $a_n = \psi$ and $a_n' \in \overline{\psi}$; 4) $a_n = \psi$ and $a_n' = \psi$.

1) $a_n \in \overline{\psi}$, $a_n' \in \overline{\psi}$ and $a_n \neq a_n'$ means that player n has changed the association strategy from one beneficial server to another. According to (4), because the hardware requirement of each player is fixed within one game session and the function $\beta_1/\left(1+\mathrm{e}^{-\beta_2 x+\beta_3}\right)$ increases monotonically with x, we know that the condition $\mathcal{L}_n(a_n, \boldsymbol{a}_{-n}) > \mathcal{L}_n(a_n', \boldsymbol{a}_{-n})$ implies that

$$\sum_{i\in\mathcal{N}\setminus\{k\}:a_i=a_n}\mathcal{R}_i > \sum_{i\in\mathcal{N}\setminus\{n\}:a_i=a_n'}\mathcal{R}_i\ . \tag{18}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2020.3002023, IEEE Transactions on Cloud Computing

Since $a_n \neq \psi$ and $a'_n \neq \psi$, according (17) and (18), there exists

$$\begin{aligned}&\Phi\left(a_n, \boldsymbol{a}_{-n}\right) - \Phi\left(a'_n, \boldsymbol{a}_{-n}\right)\\ =&\frac{1}{2}\mathcal{R}_n \sum_{i\neq n} \mathcal{R}_i I_{\{a_i=a_n\}} + \frac{1}{2}\sum_{n\neq i} \mathcal{R}_i I_{\{a_n=a_i\}}\mathcal{R}_n\\ &- \frac{1}{2}\mathcal{R}_n \sum_{i\neq n} \mathcal{R}_i I_{\{a_i=a'_n\}} - \frac{1}{2}\sum_{n\neq i} \mathcal{R}_i I_{\{a'_n=a_i\}}\mathcal{R}_n\\ =&\mathcal{R}_n \sum_{i\neq n} \mathcal{R}_i I_{\{a_i=a_n\}} - \mathcal{R}_n \sum_{i\neq n} \mathcal{R}_i I_{\{a_i=a'_n\}} > 0\ .\end{aligned} \tag{19}$$

2) $a_n \in \overline{\boldsymbol{\psi}}$ and $a'_n = \psi$ means that player n associates with the initial server ψ once again. After a number of iterations in the competition game, the load of server ψ has been offloaded to other servers $\overline{\boldsymbol{\psi}}$, which results in that its own load is lower than that of others. Under this circumstance, it might be beneficial for player n to improve the gaming experience if associating back with the initial server ψ. The reason player n deciding to offload is that the workload on each server of $\overline{\boldsymbol{\psi}}$ is over its limit $\mathcal{T}_n$, i.e., $\sum_{i\in\mathcal{N}\setminus\{n\}:a_i=a_n} \mathcal{R}_i > \mathcal{T}_n$. This implies that

$$\begin{aligned}&\Phi\left(a_n, \boldsymbol{a}_{-n}\right) - \Phi\left(a'_n, \boldsymbol{a}_{-n}\right)\\ =&\frac{1}{2}\mathcal{R}_n \sum_{i\neq n} \mathcal{R}_i I_{\{a_i=a_n\}}\\ &\qquad + \frac{1}{2}\sum_{n\neq i} \mathcal{R}_i I_{\{a_n=a_i\}}\mathcal{R}_n - \mathcal{R}_n\mathcal{T}_{n'}\\ =&\mathcal{R}_n \sum_{i\neq n} \mathcal{R}_i I_{\{a_i=a_n\}} - \mathcal{R}_n\mathcal{T}_n > 0.\end{aligned} \tag{20}$$

3) $a_n = \psi$ and $a'_n \in \overline{\boldsymbol{\psi}}$ means that player n may encounter more gaming experience loss if they keep associating with the initial server ψ. Instead, player n can choose a beneficial offloading server in $\overline{\boldsymbol{\psi}}$ to achieve a better gaming experience. According to the definition of beneficial offloading server, we know that $\sum_{i\in\mathcal{N}\setminus\{n\}:a_i=a'_n} \mathcal{R}_i < \mathcal{T}_n$. In this case,

$$\begin{aligned}&\Phi\left(a_n, \boldsymbol{a}_{-n}\right) - \Phi\left(a'_n, \boldsymbol{a}_{-n}\right)\\ =&\mathcal{R}_n\mathcal{T}_n - \frac{1}{2}\mathcal{R}_i \sum_{i\neq n} \mathcal{R}_i I_{\{a_i=a'_n\}}\\ &\qquad - \frac{1}{2}\sum_{n\neq i} \mathcal{R}_i I_{\left\{a'_k=a_i\right\}}\mathcal{R}_n\\ =&\mathcal{R}_n\mathcal{T}_n - \mathcal{R}_n \sum_{i\neq n} \mathcal{R}_i I_{\{a_i=a'_n\}} > 0.\end{aligned} \tag{21}$$

4) $a_n = \psi$ and $a'_n = \psi$ means that player n remains associated with the initial server ψ. Since player n does not change their decision, hence the NE is not affected.

Combining the results in the four cases above, we can conclude that the formulated multi-player competition game is a potential game with potential function (17) and can always reach a NE after a finite number of iterations. □

The key idea of the proof is to show that when the player updates the current decision to a better decision, the reduction of the experience loss function results in a reduction in the potential function of the multi-player competition game as well. Theorem 3 implies that any asynchronous process towards a better strategy must be finite and leads to a NE. In the next section, we take advantage of the finite improvement property to design a distributed algorithm.

4.4 Algorithm Design

In this section, for the multi-player resource competition game, we present a distributed VM placement algorithm to lead it to reach a NE. The motivation for designing a distributed algorithm is enabling players to make mutually satisfactory decisions before associating with servers. The key idea of this algorithm is to improve the VM placement strategy by using the finite improvement property of the potential game and to let one player improve their server association decision at a time. We consider a slotted time structure for the decision updates of players. In that manner, each decision slot t includes the following two phases:

Algorithm 1 Distributed VM Placement Algorithm

Initialization:

Each player associates with ψ, i.e., $a_n(0) = \psi, \forall n \in \mathcal{N}$.

Iteration:

repeat
  The cloud pushes $\{\rho_m(\boldsymbol{a}(t)), m \in \mathcal{M}\})$, i.e., the load information of all servers, to each player.
  Each player computes their beneficial server set $\Delta_n(t)$ based on (23).
  if $\Delta_n(t) \neq \varnothing$ **then**
    Each player sends their request for a target physical server to the cloud to contend for the decision update opportunity.
    if player n receives the update acknowledgement from the dispatch server **then**
      Player n chooses $a'_n \in \Delta_n(t)$ and the cloud records the server association $a_n(t+1) = a'_n$.
    else
      Keep original decision $a_n(t+1) = a_n(t)$.
    end if
  else
    Keep original decision $a_n(t+1) = a_n(t)$.
  end if
until END message is received from the cloud

1) *Server Workload Perception*: At this phase, we measure the workload on all physical servers $\mathcal{M} = \{1, ..., M\}$ in terms of CPU, GPU, RAM, etc. Taking CPU cores for illustration, the workload on each server $m \in \mathcal{M}$ can be denoted as $\rho_m\left(\boldsymbol{a}(t)\right) \triangleq \sum_{i\in\mathcal{N}:a_i(t)=m} \mathcal{R}_i$. The cloud periodically pushes the load information $\{\rho_m(\boldsymbol{a}(t)), m \in \mathcal{M}\})$ of servers to the players. By this means, each player n can obtain the resource occupancy $\{\theta_n\left(m, \boldsymbol{a}_{-n}(t)\right), m \in \mathcal{M}\}$ of other players on each server $m \in \mathcal{M}$, where

$$\theta_n\left(m, \boldsymbol{a}_{-n}(t)\right) = \begin{cases} \rho_m(\boldsymbol{a}(t)) - \mathcal{R}_n, & \text{if } a_n(t) = m,\\ \rho_m(\boldsymbol{a}(t)), & \text{otherwise.}\end{cases} \tag{22}$$

2) *Association Decision Update*: At this stage, we exploit the finite improvement property of the multi-player

competition game by having one player process a decision update. Based on the resource occupancy $\{\theta_n(m, \boldsymbol{a}_{-n}(t)), m \in \mathcal{M}\}$ of other players on each server, each player first computes their set of best response update as

$$\Delta_n(t) \triangleq \{\tilde{a}: \ \tilde{a} = \arg\min_{a \in \mathcal{A}_n} \mathcal{L}_n(a, \boldsymbol{a}_{-n}(t)) \text{ and } \mathcal{L}_n(\tilde{a}, \boldsymbol{a}_{-n}(t)) < \mathcal{L}_n(a_n(t), \boldsymbol{a}_{-n}(t))\}. \quad (23)$$

Consequently, if player n can find a better association decision at decision slot t, i.e., $\Delta_n(t) \neq \varnothing$, player n will send a decision update request to the cloud. In turn, upon receiving this request, the cloud will deem the player n willing to contend for the decision update opportunity. Otherwise, player n maintains the current decision. For multiple players who request for decision updating, the cloud randomly selects one player out of them and acknowledges their update at the next decision slot, i.e., $a_n(t+1) \in \Delta_n(t)$. When no more player has the incentive to deviate from the achieved decisions, the cloud broadcasts an END message to all players and proceeds with the VM placement strategy according to the decisions of all players at the final decision slot. Algorithm 1 illustrates the flow of the entire competition game.

We next analyze the computational complexity of this algorithm. After the initialization, all players in parallel proceed with the operations in Lines 3-14 of Algorithm 1 in each decision slot t. Since most of the operations in the algorithm are basic arithmetical calculations, the dominating part is the process of each player computing their beneficial server set $\Delta_n(t)$, which involves the sorting operation over M servers with a typical complexity of $\mathcal{O}(M \log M)$. Hence, the computational complexity in each decision slot t is $\mathcal{O}(M \log M)$. We have proved that the formulated game possesses the finite improvement property in Definition 3, i.e., it can converge to a NE within a finite number of decision slots. Thus, for a certain number of players, if we suppose that the algorithm can terminate within a bounded number of decision slots, we can deduce that the total computational complexity of the distributed VM placement algorithm is $\mathcal{O}(M \log M)$ for the cloud, where M is the number of physical servers. To derive the number of decision slots for convergence, viz., K, we define that

$$\mathcal{T}_{\max} \triangleq \max_{n \in \mathcal{N}} \{\mathcal{T}_n\}, \qquad \mathcal{T}_{\min} \triangleq \min_{n \in \mathcal{N}} \{\mathcal{T}_n\}, \quad (24)$$

$$\mathcal{R}_{\max} \triangleq \max_{n \in \mathcal{N}} \{\mathcal{R}_n\}, \qquad \mathcal{R}_{\min} \triangleq \min_{n \in \mathcal{N}} \{\mathcal{R}_n\}. \quad (25)$$

Theorem 4. *When $\mathcal{T}_n$ and $\mathcal{R}_n$ are non-negative integers for any $n \in \mathcal{N}$, the distributed VM placement algorithm terminates within at most $\mathcal{R}_{\max}^2 N^2 / 2\mathcal{R}_{\min} + \mathcal{T}_{\max}\mathcal{R}_{\max} N / \mathcal{R}_{\min}$ decision slots, i.e., $K \leq \mathcal{R}_{\max}^2 N^2 / 2\mathcal{R}_{\min} + \mathcal{T}_{\max}\mathcal{R}_{\max} N / \mathcal{R}_{\min}$.*

Proof. First, according to (17), we can directly obtain that

$$0 \leq \Phi(\boldsymbol{a}) \leq \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\mathcal{R}_{\max}^2 + \sum_{i=1}^{N}\mathcal{R}_{\max}\mathcal{T}_{\max} = \frac{1}{2}\mathcal{R}_{\max}^2 N^2 + \mathcal{R}_{\max}\mathcal{T}_{\max} N. \quad (26)$$

During a decision slot, suppose that a player $n \in \mathcal{N}$ updates their current decision a_n to the decision a_n' and can obtain better gaming experience, i.e., $\mathcal{L}_n(a_n, \boldsymbol{a}_{-n}) > \mathcal{L}_n(a_n', \boldsymbol{a}_{-n})$, we will show that this also leads to the decrement of the potential function Φ by at least $\mathcal{R}_{\min}$, i.e.,

$$\Phi(a_n, \boldsymbol{a}_{-n}) \geq \Phi(a_n', \boldsymbol{a}_{-n}) + \mathcal{R}_{\min}. \quad (27)$$

We consider three cases in the following: 1) $a_n \in \overline{\boldsymbol{\psi}}$, $a_n' \in \overline{\boldsymbol{\psi}}$ and $a_n \neq a_n'$; 2) $a_n = \psi$ and $a_n' \in \overline{\boldsymbol{\psi}}$; 3) $a_n \in \overline{\boldsymbol{\psi}}$ and $a_n' = \psi$.

For case 1, according to (19) in the proof of Theorem 3, we know that

$$\Phi(a_n, \boldsymbol{a}_{-n}) - \Phi(a_n', \boldsymbol{a}_{-n}) = \mathcal{R}_n\Big(\sum_{i \neq n}\mathcal{R}_i I_{\{a_i = a_n\}} - \sum_{i \neq n}\mathcal{R}_i I_{\{a_i = a_n'\}}\Big) > 0. \quad (28)$$

Since $\mathcal{R}_i$ are integers for any $i \in \mathcal{N}$, we know that

$$\sum_{i \neq n}\mathcal{R}_i I_{\{a_i = a_n\}} \geq \sum_{i \neq n}\mathcal{R}_i I_{\{a_i = a_n'\}} + 1. \quad (29)$$

Thus, according to (28) and (29), we have that

$$\Phi(a_n, \boldsymbol{a}_{-n}) - \Phi(a_n', \boldsymbol{a}_{-n}) \geq \mathcal{R}_n \geq \mathcal{R}_{\min}. \quad (30)$$

For case 2, according to (21) in the proof of Theorem 2, we know that

$$\Phi(a_n, \boldsymbol{a}_{-n}) - \Phi(a_n', \boldsymbol{a}_{-n}) = \mathcal{R}_n\Big(T_n - \sum_{i \neq n}\mathcal{R}_i I_{\{a_i = a_n'\}}\Big) > 0. \quad (31)$$

By the similar argument in case 1, we can have (30) as well.

For case 3, $\Phi(a_n, \boldsymbol{a}_{-n}) \geq \Phi(a_n', \boldsymbol{a}_{-n}) + \mathcal{R}_{\min}$ can be deduced similar to the proof of case 2.

Therefore, according to (26) and (27), along with driving the potential function $\Phi(\boldsymbol{a})$ to the minimum, we know that the algorithm 1 can terminate within at most $\mathcal{R}_{\max}^2 N^2 / 2\mathcal{R}_{\min} + \mathcal{T}_{\max}\mathcal{R}_{\max} N / \mathcal{R}_{\min}$ decision slots.

□

5 Performance Analysis

There are two important metrics: the number of beneficial offloading players and the overall gaming experience loss. To analyze the performance of the distributed algorithm, we follow the definition of Price of Anarchy (PoA) [35] in game theory, and then quantify the efficiency ratio of the worst NE in terms of the above two metrics over the centralized optimal solutions.

5.1 Number of Beneficial Offloading Players

The number of beneficial offloading players reflects the efficiency of VM placement since too many players associating with the initial server ψ implies an imbalance workload among other servers $\overline{\boldsymbol{\psi}}$. We denote γ as the set of NE of the multi-player resource competition game and $\boldsymbol{a}^* = (a_1^*, a_2^*, ..., a_N^*)$ as the centralized optimal solution that maximizes the number of beneficial offloading players. The PoA for the number of beneficial offloading players is defined as

$$\text{PoA} = \frac{\min_{\boldsymbol{a} \in \gamma}\sum_{n \in \mathcal{N}} I_{\{a_n \in \overline{\boldsymbol{\psi}}\}}}{\sum_{n \in \mathcal{N}} I_{\{a_n^* \in \overline{\boldsymbol{\psi}}\}}}. \quad (32)$$

As this PoA quantifies the efficiency of beneficial offloading, a larger PoA implies a better performance of the multi-player resource competition game solution.

Theorem 5. *Consider the multi-player resource competition game Γ, where $\mathcal{T}_n \geq 0$ for each player $n \in \mathcal{N}$. The PoA for the metric of the number of beneficial offloading players in (32) satisfies*

$$\frac{\left\lfloor \frac{\mathcal{T}_{\min}}{\mathcal{R}_{\max}} \right\rfloor}{\left\lfloor \frac{\mathcal{T}_{\max}}{\mathcal{R}_{\min}} \right\rfloor + 1} \leq \text{PoA} \leq 1. \tag{33}$$

Proof. Let $\tilde{\boldsymbol{a}} \in \gamma$ be an arbitrary NE of the game. Since the centralized optimum $\boldsymbol{a}^*$ maximizes the number of beneficial offloading players, we have $\sum_{n\in\mathcal{N}} I_{\{\tilde{a}_n \in \overline{\boldsymbol{\psi}}\}} \leq \sum_{n\in\mathcal{N}} I_{\{a_n^* \in \overline{\boldsymbol{\psi}}\}}$. When all players choose to associate with servers of $\overline{\boldsymbol{\psi}}$, we can get the maximum PoA as (34). In the following, the case that $\sum_{n\in\mathcal{N}} I_{\{\tilde{a}_n \in \overline{\boldsymbol{\psi}}\}} < N$ is addressed.

$$\text{PoA} = \frac{\min\limits_{\boldsymbol{a}\in\gamma} \sum\limits_{n\in\mathcal{N}} I_{\{\tilde{a}_n \in \overline{\boldsymbol{\psi}}\}}}{\sum\limits_{n\in\mathcal{N}} I_{\{a_n^* \in \overline{\boldsymbol{\psi}}\}}} \leq \frac{\sum\limits_{n\in\mathcal{N}} I_{\{\tilde{a}_n \in \overline{\boldsymbol{\psi}}\}}}{\sum\limits_{n\in\mathcal{N}} I_{\{a_n^* \in \overline{\boldsymbol{\psi}}\}}} \leq \frac{N}{N} = 1 \tag{34}$$

To proceed, we define the number of players associating with server m as $N_m(\boldsymbol{a}) \triangleq \sum_{i=1}^{N} I_{\{a_i=m\}}$ for a given decision profile $\boldsymbol{a}$. Because $\mathcal{T}_n \geq 0$, we have the case $\mathcal{L}_n(\psi, \boldsymbol{a}_{-n}) \leq \mathcal{L}_n\left(a_n \in \overline{\boldsymbol{\psi}}, \boldsymbol{a}_{-n}\right)$ $(\forall i \neq n, a_i = \psi)$, i.e., at least one player can achieve beneficial offloading by associating with a server in $\overline{\boldsymbol{\psi}}$ while the others associates with the initial server ψ. This means that for the centralized optimum $\boldsymbol{a}^*$, we have $\sum_{n\in\mathcal{N}} I_{\{a_n^* \in \overline{\boldsymbol{\psi}}\}} \geq 1$. Let $N_{m^*}(\boldsymbol{a}^*) = \max_{m\in\mathcal{M}}\{N_m(\boldsymbol{a}^*)\}$ be the number of players on the server m^* hosting most players given by centralized optimum, i.e., server m^* is the one with the most players. Assuming that player n chooses server m^*, we can get

$$(N_{m^*}(\boldsymbol{a}^*) - 1)\mathcal{R}_{\min} \leq \sum_{i\in\mathcal{N}\setminus\{n\}: a_i = m^*} \mathcal{R}_i \leq \mathcal{T}_n \leq \mathcal{T}_{\max}\,. \tag{35}$$

By rewriting (35), we can obtain that $N_{m^*}(\boldsymbol{a}^*)$ should satisfy the following condition,

$$N_{m^*}(\boldsymbol{a}^*) \leq \left\lfloor \frac{\mathcal{T}_{\max}}{\mathcal{R}_{\min}} \right\rfloor + 1. \tag{36}$$

Based on (36), we can further derive

$$\begin{aligned} \sum_{n=1}^{N} I_{\{a_n^* \in \overline{\boldsymbol{\psi}}\}} &= \sum_{m=2}^{M} N_m(\boldsymbol{a}^*) \\ &\leq (M-1)N_{m^*}(\boldsymbol{a}^*) \leq (M-1)\left(\left\lfloor \frac{\mathcal{T}_{\max}}{\mathcal{R}_{\min}} \right\rfloor + 1\right). \end{aligned} \tag{37}$$

Next, since $\sum_{n\in\mathcal{N}} I_{\{a_{\tilde{n}} \in \overline{\boldsymbol{\psi}}\}} < N$, for an arbitrary NE $\tilde{\boldsymbol{a}}$, at least one player $\tilde{n}$ chooses the initial server ψ, namely $a_{\tilde{n}} = 1$. Considering $\tilde{\boldsymbol{a}}$ is a NE, with the definition of NE, player $\tilde{n}$ will not improve their gaming experience by associating with other servers, and we can get

$$\sum_{i\in\mathcal{N}\setminus\{\tilde{n}\}: \tilde{a}_i = m} \mathcal{R}_i \geq \mathcal{T}_{\tilde{n}}, \forall m \in \overline{\boldsymbol{\psi}}\,, \tag{38}$$

which implies that

$$N_{m^*}(\tilde{\boldsymbol{a}})\,\mathcal{R}_{\max} \geq \sum_{i\in\mathcal{N}\setminus\{\tilde{n}\}: \tilde{a}_i = m} \mathcal{R}_i \geq \mathcal{T}_{\tilde{n}} \geq \mathcal{T}_{\min}\,. \tag{39}$$

Through basic operations, (39) can also be written in the following form

$$N_{m^*}(\tilde{\boldsymbol{a}}) \geq \frac{\mathcal{T}_{\min}}{\mathcal{R}_{\max}} \geq \left\lfloor \frac{\mathcal{T}_{\min}}{\mathcal{R}_{\max}} \right\rfloor. \tag{40}$$

Therefore, we have

$$\sum_{n=1}^{N} I_{\{\tilde{a}_n \in \overline{\boldsymbol{\psi}}\}} = \sum_{m=2}^{M} N_m(\tilde{\boldsymbol{a}}) \geq (M-1)\left\lfloor \frac{\mathcal{T}_{\min}}{\mathcal{R}_{\max}} \right\rfloor. \tag{41}$$

According to (34), (37) and (41), we can obtain (33), which completes the proof. □

The PoA, concerning the number of beneficial offloading players, quantifies the gap between the worst-case performance of the NE and the centralized optimum. Theorem 5 points out that the effectiveness of the proposed distributed algorithm is close to that of the centralized optimum when the gap between players in terms of gaming requirements $\mathcal{R}_n$ and the overload threshold $\mathcal{T}_n$ for achieving beneficial offloading is not large.

5.2 Overall Gaming Experience Loss

In addition to the number of beneficial offloading players, the overall gaming experience loss $\sum_{n\in\mathcal{N}} \mathcal{L}_n(\boldsymbol{a})$ is another important metric. We denote $\bar{\boldsymbol{a}}$ as the centralized optimum that minimizes the overall gaming experience loss, i.e., $\bar{\boldsymbol{a}} = \arg\min_{\boldsymbol{a}\in\prod_{n=1}^{N}\mathcal{A}_n} \sum_{n\in\mathcal{N}} \mathcal{L}_n(\boldsymbol{a})$ and the PoA with respect to the overall experience loss as

$$\text{PoA} = \frac{\max\limits_{\boldsymbol{a}\in\gamma} \sum\limits_{n\in\mathcal{N}} \mathcal{L}_n(\boldsymbol{a})}{\sum\limits_{n\in\mathcal{N}} \mathcal{L}_n(\bar{\boldsymbol{a}})}\,. \tag{42}$$

Note that a smaller overall gaming experience loss is more desirable. Hence, for the overall experience loss, the smaller the PoA is, the better performance the distributed algorithm possesses. Before presenting Theorem 6, we first introduce

$$\mathcal{L}_{n,\min} \triangleq \lambda_1\left(\frac{S_n}{W} + P\right) - \frac{\lambda_2\omega_1\mathcal{R}_n}{1 + \mathrm{e}^{\omega_2\mathcal{R}_n/\kappa + \omega_3}} - \lambda_3\mathcal{V}_n'\,, \tag{43}$$

$$\mathcal{L}_{n,\max} \triangleq \lambda_1\left(\frac{S_n}{W} + P\right) - \frac{\lambda_2\omega_1\mathcal{R}_n}{1 + \mathrm{e}^{\frac{\omega_2}{\kappa}\left(\mathcal{R}_n + \sum\limits_{i\in\mathcal{N}\setminus\{n\}} \frac{\mathcal{R}_i}{M}\right) + \omega_3}} - \lambda_3\mathcal{V}_n'\,. \tag{44}$$

Theorem 6. *For the multi-player resource competition game Γ, the PoA for the overall gaming experience loss in (42) satisfies*

$$1 \leq \text{PoA} \leq \frac{\sum\limits_{n=1}^{N} \min\{\mathcal{L}_n\left(\psi, \hat{\boldsymbol{a}}_{-n}\right), \mathcal{L}_{n,\max}\}}{\sum\limits_{n=1}^{N} \min\{\mathcal{L}_n\left(\psi, \overline{\boldsymbol{a}}_{-n}\right), \mathcal{L}_{n,\min}\}}\,. \tag{45}$$

Proof. Since the optimal solution $\bar{\boldsymbol{a}}$ is certainly better than any other solutions, PoA ≥ 1 naturally holds.

For a NE $\hat{\boldsymbol{a}} \in \gamma$, if $\bar{a}_n \neq \psi$, we conclude that for player n the resource occupation of other players on server $\hat{a}_n$ can at most be $\frac{1}{M}\sum_{i\in\mathcal{N}\setminus\{n\}} \mathcal{R}_i$. This conclusion can be proved by contradiction. Suppose that a player n at the NE $\hat{\boldsymbol{a}}$ finds that

there is a server in which the resource occupation of other players is greater than $\frac{1}{M}\sum_{i\in\mathcal{N}\setminus\{n\}}\mathcal{R}_i$, we have that

$$\sum_{i\in\mathcal{N}\setminus\{n\}:\hat{a}_i=\hat{a}_n}\mathcal{R}_i > \frac{1}{M}\sum_{i\in\mathcal{N}\setminus\{n\}}\mathcal{R}_i\ . \tag{46}$$

Since players at a NE have no incentives to change the decision unilaterally, there also exists

$$\sum_{i\in\mathcal{N}\setminus\{n\}:\hat{a}_i=\hat{a}_n}\mathcal{R}_i\ \leq \sum_{i\in\mathcal{N}\setminus\{n\}:\hat{a}_i=m}\mathcal{R}_i,\ \forall m\in\mathcal{M}\ . \tag{47}$$

From (47), we can get that

$$M\sum_{i\in\mathcal{N}\setminus\{n\}:\hat{a}_i=\hat{a}_n}\mathcal{R}_i\ \leq \sum_{m\in\mathcal{M}}\sum_{i\in\mathcal{N}\setminus\{n\}:\hat{a}_i=m}\mathcal{R}_i\ . \tag{48}$$

According to (46) and (48), a contradiction is reached, i.e.,

$$\begin{aligned}\sum_{i\in\mathcal{N}\setminus\{n\}}\mathcal{R}_i &< M\sum_{i\in\mathcal{N}\setminus\{n\}:\hat{a}_i=\hat{a}_n}\mathcal{R}_i\\ &\leq \sum_{m\in\mathcal{M}}\sum_{i\in\mathcal{N}\setminus\{n\}:\hat{a}_i=m}\mathcal{R}_i\ \leq \sum_{i\in\mathcal{N}\setminus\{n\}}\mathcal{R}_i\ .\end{aligned} \tag{49}$$

This contradiction means that the resource occupation received by player n must be less than or equal to $\sum_{i\in\mathcal{N}\setminus\{n\}}\mathcal{R}_i/M$. Combining this fact with (3), if $\hat{a}_n\in\overline{\psi}$, we can have

$$\mathcal{F}_n\left(\hat{\boldsymbol{a}}\right) \geq \frac{\omega_1\mathcal{R}_n}{1+\mathrm{e}^{\frac{\omega_2}{\kappa}(\mathcal{R}_n+\sum_{i\in\mathcal{N}\setminus\{n\}}\frac{\mathcal{R}_i}{M})+\omega_3}} = \mathcal{F}'_n(\hat{\boldsymbol{a}})\ . \tag{50}$$

By combining (4), the experience loss of player n associated with a server in $\overline{\psi}$ satisfies

$$\mathcal{L}_n\left(\overline{\psi},\hat{\boldsymbol{a}}_{-n}\right) \leq\ \lambda_1\mathcal{D}_n - \lambda_2\mathcal{F}'_n(\hat{\boldsymbol{a}}) - \lambda_3\mathcal{V}'_n = \mathcal{L}_{n,\max}\ . \tag{51}$$

In addition, if $\mathcal{L}_n\left(\psi,\hat{\boldsymbol{a}}_{-n}\right) < \mathcal{L}_{n,\max}$ and $\hat{a}_n\in\overline{\psi}$, player n can further choose the initial server ψ to achieve lower experience loss, i.e., $\hat{a}_n=\psi$. Thus, we get

$$\begin{aligned}\mathcal{L}_n(\hat{\boldsymbol{a}}) &= \min\{\mathcal{L}_n\left(\psi,\hat{\boldsymbol{a}}_{-n}\right),\mathcal{L}_n\left(\overline{\psi},\hat{\boldsymbol{a}}_{-n}\right)\}\\ &\leq \min\{\mathcal{L}_n\left(\psi,\hat{\boldsymbol{a}}_{-n}\right),\mathcal{L}_{n,\max}\}\ .\end{aligned} \tag{52}$$

Besides, for the centralized optimum $\overline{\boldsymbol{a}}$, if $\overline{a}_n\in\overline{\psi}$, we have

$$\mathcal{F}_n\left(\overline{\boldsymbol{a}}\right) \leq \frac{\omega_1\mathcal{R}_n}{1+\mathrm{e}^{\omega_2\mathcal{R}_n/\kappa+\omega_3}}\ . \tag{53}$$

Then, we can infer that

$$\mathcal{L}_n(\overline{\psi},\overline{\boldsymbol{a}}_{-n}) \geq \lambda_1\mathcal{D}_n - \frac{\lambda_2\omega_1\mathcal{R}_n}{1+\mathrm{e}^{\omega_2\mathcal{R}_n/\kappa+\omega_3}} - \lambda_3\mathcal{V}'_n = \mathcal{L}_{n,\min}\ . \tag{54}$$

Moreover, if $\mathcal{L}_n\left(\psi,\overline{\boldsymbol{a}}_{-n}\right) < \mathcal{L}_{n,\min}$ and $\overline{a}_n\in\overline{\psi}$, the total experience loss can be further reduced by letting player n choose the initial server ψ, i.e., $\overline{a}_n=\psi$. The reason is that the decision of player n to change to ψ does not cause extra resource competition of other players associating with $\overline{\psi}$. In this case, we get

$$\begin{aligned}\mathcal{L}_n(\overline{\boldsymbol{a}}) &= \min\{\mathcal{L}_n\left(\psi,\overline{\boldsymbol{a}}_{-n}\right),\mathcal{L}_n(\overline{\psi},\overline{\boldsymbol{a}}_{-n})\}\\ &\geq \min\{\mathcal{L}_n\left(\psi,\overline{\boldsymbol{a}}_{-n}\right),\mathcal{L}_{n,\min}\}\ .\end{aligned} \tag{55}$$

According to (52) and (55), we can obtain (45) and conclude the proof. □

From Theorem 6, we can intuitively infer that when the number of available servers increases, the performance of the worst-case NE can be improved. Specifically, the larger the number of servers (i.e., M) is, the smaller $\mathcal{L}_{n,\max}$ is. In addition, when players are able to have good gaming experience by associating with the initial server ψ, viz., $\mathcal{L}_n\left(\psi,\boldsymbol{a}_{-n}\right)$ is smaller, the worst-case NE is closer to the centralized optimum $\bar{\boldsymbol{a}}$, i.e., a lower PoA is achieved.

6 Experimental Results

In our experiments, we first simulate a wireless environment with 4 BSs and multiple game players and their respective mobile devices. Each BS supports 5 wireless channels with channel bandwidth of 10 MHz, as commonly used in LTE system [36]. By adopting the wireless settings and the proposed algorithm used in [23] to perform simulations, i.e., letting mobile devices to compete for their wireless transmission rates, we can obtain the actual gaming resolution of each gaming request of players.

Then, for these gaming requests, we employ the distribution of player interval times and the distribution of game session lengths in [37], derived from OnLive logs [38] consisting of 1.6 million player sessions, to further synthesize a sequence of cloud gaming requests. Next, we send these gaming requests to a cloud datacenter with 10 physical servers, each of which is equipped with AMD Ryzen Threadripper 1950X (a CPU with 18 physical cores), 4 GTX 1080 graphics card, and 128 GB RAM. Each physical server hosts multiple VMs, while each VM is allocated with 128 GB disk (enough for all games adopted) and created based on Win10 1703 with all common software dependencies of games installed, including the GamingAnywhere server program. Note that pre-copying game files to VMs based on the prediction of the distribution of player requests, which is complementary to our work, can further optimize the efficiency of resource utilization. Finally, once the server association decisions of players are determined by the proposed algorithm, the cloud will copy game files for each player to a VM in the chosen server, and then start the game session. Parameter-related experiment settings are given in Table 4.

TABLE 4: Symbol-related Experiment Settings

Parameter	Value
Number of BSs (B)	4
Sub-channels / BS (C)	5
Channel bandwidth (ω)	10 MHz
Number of physical servers (M)	10
Number of physical CPU cores (κ) / server	18
Memory size of RAM (μ) / server	128 GB

By analyzing the game popularities in various categories on Steam and Epic Games, we select 40 games, some of which have been given in Table 3, with various hardware requirements and types, for available cloud gaming. Based on the modeling of the above player activities and the 40 chosen games, we can simulate the cloud gaming requests of players, and hence use them throughout the experiments to verify the performance of the proposed distributed algorithm. Since the final reached NE of the multi-player resource competition game is not constant, we perform 100

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2020.3002023, IEEE Transactions on Cloud Computing

times for each type of experiment and average all related metrics, in order to evaluate the performance of the algorithm more accurately.

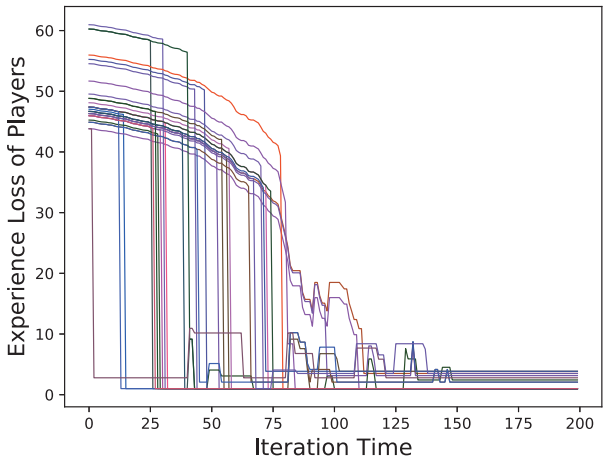


Fig. 3: Players chosen at random finally reach a NE.

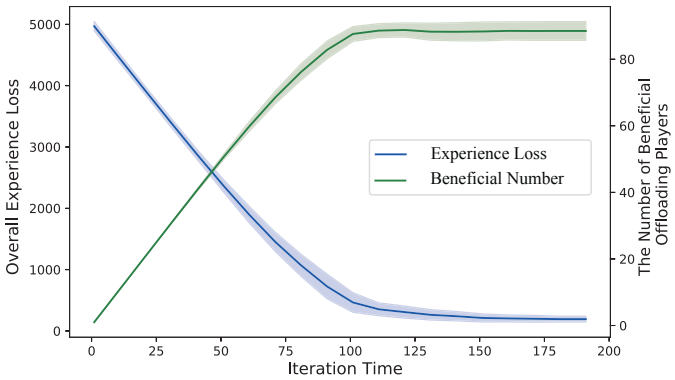


Fig. 4: Variation and convergence of two important metrics.

6.1 Convergence of the Resource Competition Game

For portraying the convergence of the multi-player resource competition game, we collect all the statics in experiments and present the experience variation of 30 players chosen at random. During the competition process, as shown in Fig. 3, the experience loss of each player is high at the beginning when all players are associating with the initial server ψ. Nonetheless, with the proposed algorithm, there is one player updating their strategy at each iteration. By virtue of this, the holistic gaming experience of players is improving over the iterations. It demonstrates that the algorithm can keep improving the gaming experiences of players and eventually lead the competition game to converge to an equilibrium.

Further, in order to corroborate the overall performance of the proposed algorithm, two metrics in terms of the overall experience loss and the number of beneficial offloading players are investigated. As shown in Fig. 4, both metrics can finally reach a steady state, which demonstrates the effectiveness of the proposed algorithm, especially on improving the gaming experience of players. In addition, with 100 experiments performed, the blue and green shaded areas in Fig. 4 represent the variation range of 1) the total experience loss and 2) the number of beneficial offloading players, respectively. We can intuitively observe that the performance of the proposed distributed algorithm is relatively stable and can satisfy most scenarios.

6.2 Performance Comparison with Other Strategies

In order to evaluate the performance of our algorithm, we compare the distributed VM placement algorithm, i.e., the potential game strategy, with three heuristic solutions:

1) **Polling Placement Strategy**: The dispatch server numbers all servers and uses them in a cyclical order. Whenever there is a gaming request, the server, represented by the polling number, is assigned to the requesting player, that is, the placement strategy is $\{1, 2, ..., M, 1, 2, ...\}$.
2) **Random Greedy Placement Strategy**: When the server is initialized, all players queue to wait for the placement of VMs. At every decision slot, the dispatch server randomly chooses a player (and then dequeues it) and assigns a server that can provide the best gaming experience for their specific requirements, until the queue is empty.
3) **Sorted Greedy Placement Strategy**: This strategy is similar to the random greedy placement strategy, except that this strategy dequeues the player, who can have the best gaming experience (rather than at random), at every decision slot according to the present workloads of all servers.

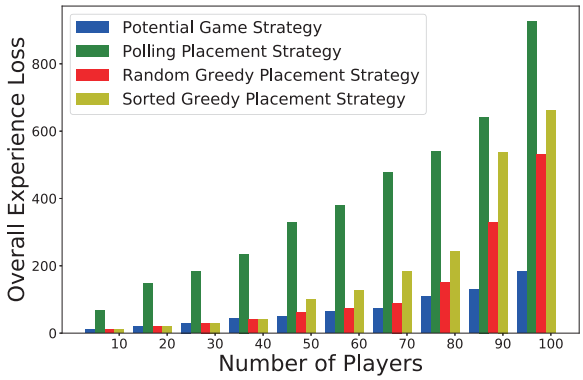


Fig. 5: Performance comparison on experience losses.

Comparison results of different algorithms are given in Fig. 5. For the metric of the overall experience loss, when there are more than 60 players, the proposed algorithm can achieve up to 75%, 54%, and 66% improvement compared with the polling placement strategy, the random greedy placement strategy, and the sorted greedy placement strategy, respectively. Besides, if the algorithm is not performed in a distributed manner, complete information is required for the cloud, i.e., all players need to report all their local parameters to the cloud. This will result in a high system overhead for large amounts of information collection and may also cause privacy issues. In addition, since different players may pursue different interests, players may not have the motivation to follow a centralized solution. It is worth

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCC.2020.3002023, IEEE Transactions on Cloud Computing

noting that when the resource usages of servers increase gradually, i.e., with more players, our distributed algorithm can still achieve better overall performance improvement than other strategies. Meanwhile, service providers generally expect to maximize their server efficiencies to reduce their expenditures, which fits well with the proposed algorithm.

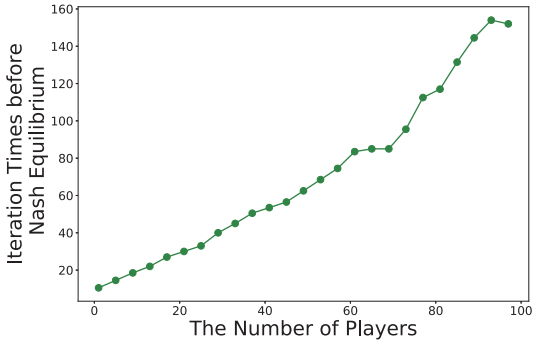


Fig. 6: Iteration times for convergence.

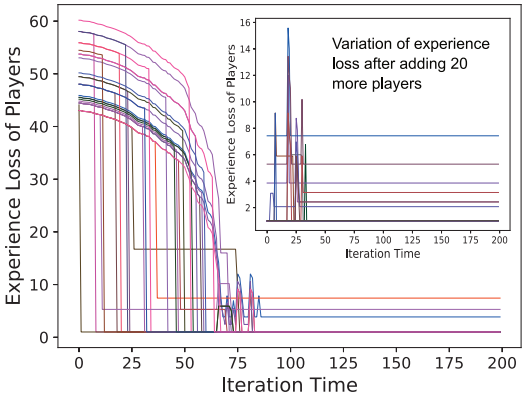


Fig. 7: Experience loss after adding 20 more players.

6.3 Additional Investigation

Next, we study the relationship between the number of players and the number of iterations required for the multi-player resource competition game. The results in Fig. 6 show that as the number of players increases, the average number of decision slots for convergence increases almost linearly. It can be seen that our algorithm scales well with the number of players. Further, for a certain number of players, cloud gaming service providers can use our algorithm to quickly determine VM placement strategies for a large number of servers with the time complexity $\mathcal{O}(M \log M)$, instead of getting stuck in solving NP-hard problems with centralized optimization algorithms.

Different from Fig. 3, as in Fig. 7, we simulate a practical scenarios in which players dynamically request cloud gaming services. Specifically, we first bring 80 players to a stable state (NE) by using the proposed distributed algorithm. After this stable state is established as in Fig. 7, we add another 20 gaming requests from new players. From Fig. 7, we can observe that the overall 100 players requires 38 decision slots to reach a stable state again, which is of the same order of magnitude as the number of new players, i.e., 20, instead of 100, the total number of players. This means that when the stable state of the players has been established, if new players join, the number of decision slots, required for all players including new players to reach a stable state, is only related to the number of players arriving later, which also corroborates that our algorithm scales well with the number of players.

7 Conclusion

In this paper, we have presented a distributed algorithm based on the potential game theory. We have proved that optimization problems in multi-player cloud gaming scenarios are NP-hard. By constructing a potential function, we have proved the presence of Nash Equilibrium in the multi-player competition game. We have quantified the performance of the algorithm in terms of PoA, and demonstrated the convergence of our algorithm. We have presented experimental results to demonstrate that the proposed algorithm achieves superior performance than other strategies and scales well with the increasing number of players.

In the future, a more appropriate fit function to quantify the parameters will be constructed for the cloud game scene. Moreover, the constraints of multidimensional parameters in cloud gaming will be taken into account simultaneously to achieve further optimization.

Acknowledgement

This work was supported by the National Key R&D Program of China (No.2019YFB2101901 and No.2018YFC0809803), National Science Foundation of China (No.U1711265 and No.61902333), the Shenzhen Institute of Artificial Intelligence and Robotics for Society (AIRS), Chinese National Engineering Laboratory for Big Data System Computing Technology and Canadian Natural Sciences and Engineering Research Council. Especially, we would like to thank the editors of IEEE TCC and the reviewers for their help and support in making this work possible.

References

[1] "Mobile Revenues Account for More Than 50% of the Global Games Market as It Reaches $137.9 Billion in 2018." [Online]. Available: https://newzoo.com/insights/articles/global-games-market-reaches-137-9-billion-in-2018-mobile-games-take-half/

[2] W. Cai, R. Shea, C.-Y. Huang, K.-T. Chen, J. Liu, V. C. Leung, and C.-H. Hsu, "The future of cloud gaming," *Proc. IEEE*, vol. 104, no. 4, pp. 687–691, Apr. 2016.

[3] W. Cai, R. Shea, C.-Y. Huang, K.-T. Chen, J. Liu, V. C. Leung, and C.-H. Hsu, "A survey on cloud gaming: Future of computer games," *IEEE Access*, vol. 4, pp. 7605–7620, 2016.

[4] W. Cai, M. Chen, and V. C. M. Leung, "Toward gaming as a service," *IEEE Internet Comput.*, vol. 18, no. 3, pp. 12–18, May 2014.

[5] M. Agiwal, A. Roy, and N. Saxena, "Next generation 5g wireless networks: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 3, pp. 1617–1655, Thirdquarter 2016.

[6] X. Zhang, H. Chen, Y. Zhao, Z. Ma, Y. Xu, H. Huang, H. Yin, and D. O. Wu, "Improving cloud gaming experience through mobile edge computing," *IEEE Wirel. Commun.*, vol. 26, no. 4, pp. 178–183, Aug. 2019.

[7] 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. Multimed. Comput. Commun. Appl.*, vol. 10, no. 1s, p. 10, Jan. 2014.

[8] T. Mastelic, A. Oleksiak, H. Claussen, I. Brandic, J.-M. Pierson, and A. V. Vasilakos, "Cloud computing: Survey on energy efficiency," *ACM Comput. Surv.*, vol. 47, no. 2, pp. 33:1–33:36, Dec. 2014.

[9] M. Masdari, S. S. Nabavi, and V. Ahmadi, "An overview of virtual machine placement schemes in cloud computing," *Journal of Network and Computer Applications*, vol. 66, pp. 106 – 127, 2016.

[10] 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. of the 11th annual workshop on network and systems support for games*, 2012, p. 3.

[11] W. Cai, Z. Wang, J. B. Ernst, Z. Hong, C. Feng, and V. C. Leung, "Decentralized applications: The blockchain-empowered software system," *IEEE Access*, vol. 6, pp. 53 019–53 033, 2018.

[12] K. I. Kim, S. Y. Bae, D. C. Lee, C. S. Cho, H. J. Lee, and K. C. Lee, "Cloud-based gaming service platform supporting multiple devices," *ETRI Journal*, vol. 35, no. 6, pp. 960–968, 2013.

[13] E. Depasquale, A. Zammit, M. Camilleri *et al.*, "An analytical method of assessment of remotefx as a cloud gaming platform," in *17th IEEE Mediterranean Electrotechnical Conference (MELECON 2014)*, Apr. 2014, pp. 127–133.

[14] Q. Hou, C. Qiu, K. Mu, Q. Qi, and Y. Lu, "A cloud gaming system based on nvidia grid gpu," in *2014 13th International Symposium on Distributed Computing and Applications to Business, Engineering and Science*, 2014, pp. 73–77.

[15] J. Wu, C. Yuen, and N.-M. Cheung, "Enabling adaptive high-frame-rate video streaming in mobile cloud gaming applications," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 12, pp. 1988–2001, 2015.

[16] Q. Zhu and T. Tung, "A performance interference model for managing consolidated workloads in qos-aware clouds," in *2012 IEEE Fifth International Conference on Cloud Computing*, 2012, pp. 170–179.

[17] S. Wang, Y. Liu, and S. Dey, "Wireless network aware cloud scheduler for scalable cloud mobile gaming," in *2012 IEEE International Conference on Communications (ICC)*, 2012, pp. 2081–2086.

[18] H.-J. Hong, D.-Y. Chen, C.-Y. Huang, K.-T. Chen, and C.-H. Hsu, "Qoe-aware virtual machine placement for cloud games," in *2013 12th Annual Workshop on Network and Systems Support for Games (NetGames)*, 2013, pp. 1–2.

[19] 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. 2014.

[20] Y. Li, X. Tang, and W. Cai, "Play request dispatching for efficient virtual machine usage in cloud gaming," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 12, pp. 2052–2063, Dec. 2015.

[21] M. Marzolla, S. Ferretti, and G. D'Angelo, "Dynamic resource provisioning for cloud-based gaming infrastructures," *Comput. Entertain.*, vol. 10, no. 1, pp. 4:1–4:20, Dec. 2012.

[22] D. Finkel, M. Claypool, S. Jaffe, T. Nguyen, and B. Stephen, "Assignment of games to servers in the onlive cloud game system," in *Proc. of the 13th Annual Workshop on Network and Systems Support for Games*, 2014, p. 4.

[23] E. Meskar, T. D. Todd, and D. Zhao, "Energy aware offloading for competing users on a shared communication channel," *IEEE Trans. Mobile Comput.*, vol. 16, no. 1, pp. 87–96, Jan. 2017.

[24] T. Q. Dinh, Q. D. La, T. Q. S. Quek *et al.*, "Distributed Learning for Computation Offloading in Mobile Edge Computing," *IEEE Trans. Commun.*, vol. 66, no. 12, pp. 6353–6367, Dec. 2018.

[25] F. Liu, G. Tang, Y. Li, Z. Cai, X. Zhang, and T. Zhou, "A survey on edge computing systems and tools," *Proc. IEEE*, vol. 107, no. 8, pp. 1537–1562, Aug. 2019.

[26] T. Taleb, K. Samdanis, and B. Mada, "On multi-access edge computing: A survey of the emerging 5g network edge cloud architecture and orchestration," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1657–1681, Thirdquarter 2017.

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

[28] K. Chen, Y. Chang, H. Hsu, D. Chen, C. Huang, and C. Hsu, "On the quality of service of cloud gaming systems," *IEEE Trans. Multimed.*, vol. 16, no. 2, pp. 480–495, Feb. 2014.

[29] T. S. Rappaport *et al.*, *Wireless communications: principles and practice*. prentice hall PTR New Jersey, 1996, vol. 2.

[30] T. M. Mitchell, *Machine Learning*, 1st ed. New York, NY, USA: McGraw-Hill, Inc., 1997.

[31] K.-H. Loh, B. Golden, and E. Wasil, "Solving the maximum cardinality bin packing problem with a weight annealing-based algorithm," in *Operations Research and Cyber-Infrastructure*. Springer, 2009, pp. 147–164.

[32] S. H. Owen and M. S. Daskin, "Strategic facility location: A review," *European journal of operational research*, vol. 111, no. 3, pp. 423–447, 1998.

[33] S. A. Cook, "The complexity of theorem-proving procedures," in *Proc. of the third annual ACM symposium on Theory of computing*, 1971, pp. 151–158.

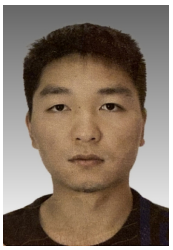
[34] D. Monderer and L. S. Shapley, "Potential games," *Games and Economic Behavior*, vol. 14, no. 1, pp. 124 – 143, 1996.

[35] T. Roughgarden, *Selfish routing and the price of anarchy*. MIT press Cambridge, 2005, vol. 174.

[36] J. A. Ayala-Romero, A. Garcia-Saavedra, M. Gramaglia, X. Costa-Perez, A. Banchs, and J. J. Alcaraz, "Vrain: A deep learning approach tailoring computing and radio resources in virtualized rans," in *the 25th Annual International Conference on Mobile Computing and Networking (MobiCom 2019)*, 2019.

[37] Y. Li, Y. Deng, X. Tang, W. Cai, X. Liu, and G. Wang, "Cost-efficient server provisioning for cloud gaming," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 14, no. 3s, pp. 55:1–55:22, Jun. 2018.

[38] D. Finkel, M. Claypool, S. Jaffe, T. Nguyen, and B. Stephen, "Assignment of games to servers in the onlive cloud game system," in *2014 13th Annual Workshop on Network and Systems Support for Games*, Dec. 2014, pp. 1–3.



Yiwen Han [S'18] received his B.S. degree from Nanchang University, China, and M.S. degree from Tianjin University, China, in 2015 and 2018, respectively, both in communication engineering. He received the Outstanding B.S. Graduates in 2015 and M.S. National Scholarship of China in 2016. He is currently pursuing the Ph.D. degree in computer science at Tianjin University. His current research interests include edge computing, cloud gaming, and deep learning.

Dongyu Guo [S'18] is a postgraduate in Tianjin University, China. He received the B.S. degree in the Department of Computer Science and Technology, Yanshan University in 2017. His current research direction is user optimization in cloud game scenarios, and the application of game theory in mobile cloud game scenarios.

Wei Cai [S'12-M'16] is currently an Assistant Professor of Computer Engineering, the Director of the Human-Cloud Systems Laboratory, School of Science and Engineering at The Chinese University of Hong Kong, Shenzhen, China. He received Ph.D., M.Sc., and B.Eng. from The University of British Columbia (UBC), Seoul National University, and Xiamen University in 2016, 2011, and 2008, respectively. From 2016 to 2018, he was a Postdoctoral Research Fellow with UBC. Focusing on the research of distributed computing, human-computer interaction, and multimedia, he has authored over 50 technical papers in the topics of cloud, blockchain, and video games. He is serving as an associate editor of IEEE Transactions on Cloud Computing. He was a recipient of the 2015 Chinese Government Award for the Outstanding Self-Financed Students Abroad, the UBC Doctoral Four-Year-Fellowship from 2011 to 2015, and the Brain Korea 21 Scholarship. He also received the best student paper award from ACM BSCI2019 and the best paper awards from CCF CBC2018, IEEE CloudCom2014, SmartComp2014, and CloudComp2013.

Xiaofei Wang [S'06, M'13, SM'18] (xiaofeiwang@tju.edu.cn) is currently a Professor with the Tianjin Key Laboratory of Advanced Networking, School of Computer Science and Technology, Tianjin University, China. He got master and doctor degrees in Seoul National University from 2006 to 2013, and was a Post-Doctoral Fellow with The University of British Columbia from 2014 to 2016. Focusing on the research of social-aware cloud computing, cooperative cell caching, and mobile traffic offloading, he has authored over 100 technical papers in the IEEE JSAC, the IEEE TWC, the IEEE WIRELESS COMMUNICATIONS, the IEEE COMMUNICATIONS, the IEEE TMM, the IEEE INFOCOM, and the IEEE SECON. He was a recipient of the National Thousand Talents Plan (Youth) of China. He received the "Scholarship for Excellent Foreign Students in IT Field" by NIPA of South Korea from 2008 to 2011, the "Global Outstanding Chinese Ph.D. Student Award" by the Ministry of Education of China in 2012, and the Peiyang Scholar from Tianjin University. In 2017, he received the "Fred W. Ellersick Prize" from the IEEE Communication Society.

Victor C.M. Leung [S'75, M'89, SM'97, F'03] is a Distinguished Professor of Computer Science and Software Engineering at Shenzhen University. He was a Professor of Electrical and Computer Engineering and holder of the TELUS Mobility Research Chair at the University of British Columbia (UBC) when he retired from UBC in 2018 and became a Professor Emeritus. His research is in the broad areas of wireless networks and mobile systems. He has published widely in these areas. Dr. Leung is serving on the editorial boards of the IEEE Transactions on Green Communications and Networking, IEEE Transactions on Cloud Computing, IEEE Access, IEEE Network, and several other journals. He received the IEEE Vancouver Section Centennial Award, 2011 UBC Killam Research Prize, 2017 Canadian Award for Telecommunications Research, and 2018 IEEE TCGCC Distinguished Technical Achievement Recognition Award. He co-authored papers that won the 2017 IEEE ComSoc Fred W. Ellersick Prize, 2017 IEEE Systems Journal Best Paper Award, 2018 IEEE CSIM Best Journal Paper Award, and 2019 IEEE TCGCC Best Journal Paper Award. He is a Fellow of IEEE, the Royal Society of Canada, Canadian Academy of Engineering, and Engineering Institute of Canada. He is named in the current Clarivate Analytics list of "Highly Cited Researchers".