SDN-based Game-Aware Network Management for Cloud Gaming

Maryam Amiri[1], Hussein Al Osman[1], Shervin Shirmohammadi[1], Maha Abdallah[2]
[1] Distributed and Collaborative Virtual Environment Research (DISCOVER Lab), University of Ottawa, Canada
{mamir097 | halosman}@uottawa.ca, shervin@discover.uottawa.ca
[2] Sorbonne Universités, UPMC Univ. Paris 06, CNRS, LIP6 UMR 7606, Paris, France
Maha.Abdallah@lip6.fr

Abstract— **Cloud based video games bring new opportunities to the gaming industry, and enable end-users to play high-end graphic games on any low-end device without high performance hardware requirements. As the major computational parts of game processing, including user's input processing, rendering and encoding the game scene, and video streaming are performed in data centers, data centers play a key role in providing high quality games to end-users. Software Defined Networking (SDN) enables the management and control of communication inside the data center in a centralized fashion, such that it can be employed as a means to optimize flow distribution. In this paper, we present a Linear Programing (LP) optimization-based method for optimally assigning game servers to gaming sessions and selecting the best communication path within a cloud gaming datacenter. Our optimization model considers the type of requested games, current server loads, and current path delays. Our experimental results show that the proposed model minimizes the average delay of all players within a datacenter by 9.6%, and outperforms the existing server-centric and network-centric models.**

Keywords—Cloud gaming; Software Defined Network (SDN); Linear Programing;

I. INTRODUCTION

As a result of recent advancements in cloud computing and data centers, there has been a dramatic increase in the demand for cloud computing services in the last few years. In fact, centralizing the resources and computational processes into data centers brings about inexpensive and flexible opportunities for a multitude of existing applications. Gaming services is one of these applications that can benefit from cloud computing advancements. The ubiquitous and scalable nature of data centers leads to cost reductions for game service providers by deploying games faster through the scalable environment of the cloud and increasing their revenue by attracting more users. Furthermore, it enables users to play numerous games using a thin-client regardless of their location or what platform they use (PCs, laptops, tablets, smartphones). These advantages are enabling a new paradigm of online gaming called Cloud Gaming.

In a cloud gaming architecture, the major part of computational operations including game logic processing, video rendering, video encoding, and video streaming are performed in the cloud. Hence, the computational load is drastically decreased at the client.

Despite its advantages, cloud gaming suffers from latency imposed by the network, which amounts to approximately 1.7 times (resp. 3 times) more delay than console (resp. PC) games [13]. This can have a damaging impact on the Quality of Experience (QoE) of players. Moreover, only 70% of end-users are able to meet the required latency threshold of 100 ms to match the experience of console or PC games [4].

In cloud gaming, games are run in the cloud; i.e., gamers' commands are sent to the cloud server and appropriate actions are taken by the game engine so that it produces the corresponding video frames, rendered by a GPU and encoded by a video codec. Afterwards, the compressed frames are sent to players through the network, and finally decoded and played out on thin-client devices. All of the above steps can impose additional latency on cloud gaming systems. A wide range of techniques have been proposed to deal with these latency issues. Most of these techniques focus on the processing of the game and the communication of game data within the cloud, and can be divided to two main categories:

1. Computational methods to optimize the processing of games in data center servers (e.g. gaming logic processing, video rendering, video encoding, and video streaming).
2. Network methods to improve the distribution of streamed video within the cloud.

Since the current cloud infrastructure is not able to guarantee the QoS requirements of cloud gaming systems [5], some of the existing cloud gaming companies, like Gaikai, have invested in their own proprietary data centers [22]. Therefore, solutions to reduce latency have a considerable importance to build the cloud gaming data center's infrastructure in order to meet QoS requirements. However, most of the previous studies that deal with latency in on-demand gaming are primarily focused on computational operations on the server side, and very few works have been proposed to improve latency issues in cloud gaming networks. In this paper, we focus on network solutions to reduce communication delay within the cloud gaming network, and we propose a Linear Programming (LP) method that addresses two main questions: 1) how are gaming sessions mapped onto servers, and 2) which path is used for data communication between the core and the access switch for a gaming session?

II. BACKGROUND AND RELATED WORKS

A. *Effect of Network Latency and Jitter*

Delay sensitivity varies depending on the game genre. For example, First Person Shooter (FPS) games are more sensitive to delay compared to real-time strategy (RTS) games [7]. An empirical study on user tolerance for delay demonstrates that a delay of less than 100 msec is highly desirable for high action paced games, whereas 150 msec is the delay threshold in slow paced games [16]. Beside the game genre, the type of gaming device (e.g. desktop, laptop or smartphone) and even more the experience of the gamers (beginner, intermediate or expert gamer) have a significant impact on delay sensitivity [6, 8, 10].

In addition to latency itself, the amount of latency variation, known as jitter, also has detrimental effects on gamers' QoE [3, 10]. In [19], the authors show that cloud games suffer from a much higher level of jitter compared to console games. In some cases, jitter has been found to be even larger than 100 msec in cloud games, which is a problem since a jitter of more than 70 msec has a significant negative effect on players' QoE [2].

B. *Cloud Gaming Delay Reduction Methods*

The overall delay in cloud gaming consists of processing, playout, and network delays; hence, a variety of techniques have been proposed to cope with these delay sources. Several studies have focused on latency reduction techniques in game engines as well as video encoders [23, 24, 26]. However, few network solutions have been devised to improve latency for cloud games. Among these, several server selection approaches for cloud gaming data centers have been suggested. Choy et al. [4] propose a voting-based heuristic method for server selection and game placement. They introduced a voting-based game-placement strategy in which end users vote for a game to be hosted on a server in order to increase gamer's coverage. Beskow et al [2] suggest another server selection method to find the optimal servers to cover a group of gamers belonging to the same geographic region with the goal of minimizing latency in data transfer. Chen et al. [11] present a method for allocating virtual machines (VM) to physical machines (PM) in the cloud gaming infrastructure that tackles the challenge of trade-off between users' QoE and providers' net profit, and propose a heuristic method that maximizes the net profit of cloud gaming providers along with maintaining just good enough QoE level for gamers.

Besides server selection methods, network routing methods can be employed to reduce latency for delay sensitive applications. In our previous work [20], we proposed a Software Defined Networking (SDN) controller that optimizes the distribution of flows among the various redundant paths inside the cloud network to reduce the delay and jitter experienced by players.

III. DATA CENTER ARCHITECTURE AND SOFTWARE DEFINED NETWORKING

Figure 1 shows a typical architecture of today's datacenters -fat-tree- that consists of a core switch that dispatches client requests among aggregation switches, each of which further dispatches the request to a Top of Rack (ToR) switch, which finally dispatches the request to a processing node running multiple Virtual Machines (VM) [12]. In cloud gaming systems, since large amounts of data is transmitted as compressed video, the core switches are prone to congestion. So, some auxiliary network routes are employed as soon as the main paths are congested. An empirical study of a data center network consisting of 150 inter-switches and 1500 servers shows that 15% congestion situations can last for more than 100 seconds while many network links are often underutilized [17]. Therefore, overprovisioning can be used to address suboptimal network utilization. However, as current data centers usually comprise thousands of physical and virtual machines, overprovisioning is no longer appropriate [15].

Although several methods have been proposed to determine the optimal paths in the network [9, 18, 21], these methods usually select the best routes based on the link metrics provided at the setup time, and they often fail to take into consideration the current status of links, such as current available bandwidth, delay, etc., or the requirements of data flows passing through the network.

SDNs present a new approach to traffic forwarding through a centralized network controller. In short, SDN separates the network controls and the forwarding plane to optimize each layer separately. Also, SDN brings the application layer and the network layer closer together, and allows the network to be more adaptable to different conditions and assists it in responding to application requests. This adaptability is highly desirable for delay sensitive applications like cloud gaming, especially when the number of requests is abruptly increased. In traditional networks, the routing protocols often rely on predefined static routes and do not take into account dynamic parameters like bandwidth utilization, packet loss, etc. SDN provides a means for controlling the network in a centralized manner in order not to only accommodate current network conditions but also optimize QoS parameters.

Since SDN controls the network in a centralized manner, various optimization techniques can be easily applied to find the optimal path based on the current state of links while guaranteeing QoS requirements. These methods can be divided into two main categories:

- Methods that improve utilization of network resources.
- Methods that improve utilization of server resources.

Given that computational resources in game servers and network resources jointly have a significant importance in the performance of cloud gaming architecture, they should

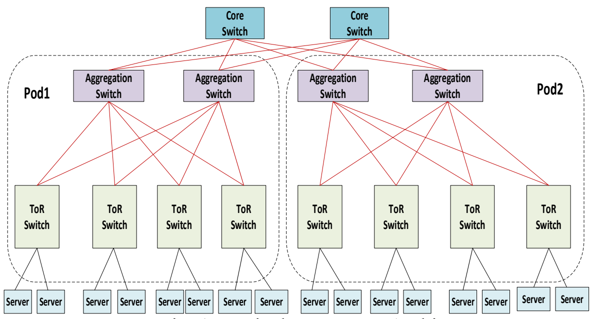


Fig. 1. Typical Datacenter Architecture

both be taken into consideration to meet the games' QoE requirements.

IV. PROPOSED GAME-AWARE OPTIMIZATION METHOD

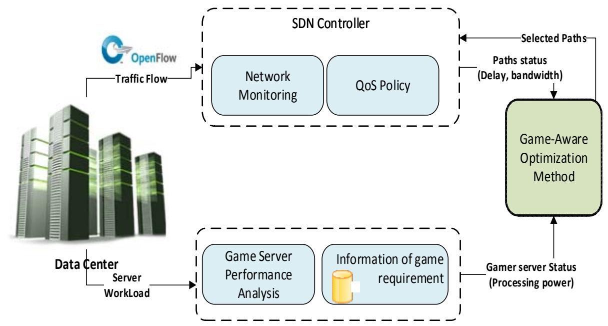
The goal of our proposed method is to optimally assign games to servers in the cloud and select the best communication path within the datacenter for a session's data streams. Towards this end, we employ a Linear Programing (LP) optimization technique. Our technique considers the type of requested games, network information pertaining to link status, and current server loads.

Figure 2 shows the overall framework of the proposed method. The SDN controller periodically monitors the latency and available bandwidth on each link using the OpenFlow protocol. In [20], we described an algorithm to measure the delay of each path between the core switches and the ToR switches. The game server performance analysis module monitors and analyzes the performance of the game servers in terms of available processing resources. Also, the predefined information and requirements of games are stored in an auxiliary database.

Therefore, network-related information, server-related information, and games requirements are fed into the proposed game-aware optimization method by the SDN controller, performance engine, and game database respectively. Afterwards, the proposed game-aware optimization method makes decisions on the server selection and the corresponding paths.

We mathematically formulate the optimization problem using LP. Given $S = (s_1, s_2, \cdots, s_{Ns})$ and $H = (h_1, h_2, \cdots, h_{Nh})$ denoting a set of game servers and switches (including Core, Aggregation and ToR switches) respectively, N_s is the number of available servers and N_h is the number of switches in a datacenter. The topology of a datacenter, including servers, switches, and disjoint paths, can be represented as a graph G(V, P), where V is S ∪ H.and The capacity of server s_j is denoted by C_{s_j}, where $1 \leq j \leq N_s$, which can be thought of as the processing resources available on the server.

The amount of processing and bandwidth resources requested by gaming session i are expressed as rp_i and rb_i, respectively, where $1 \leq i \leq N_g$ and N_g is the number of gaming sessions executing in the datacenter.


SDN Controller
OpenFlow
Traffic Flow
Network Monitoring
QoS Policy
Selected Paths
Paths status (Delay, bandwidth)
Game-Aware Optimization Method
Data Center
Server WorkLoad
Game Server Performance Analysis
Information of game requirement
Gamer server Status (Processing power)
Fig. 2. Proposed Game-Aware Optimization Method

The game database provides the values of rp_i and rb_i based on the pre-defined requirements of the corresponding game. The maximum bandwidth of each path is assumed to be BW_{Max} . Let $d_{n_{ijk}}$ denote the network delay of the kth path connecting the core-switch to server j executing gaming session i within the data center. Note that $1 \leq k \leq N_p$ where N_p is the total number of paths between the core switch and all servers. $d_{n_{ijk}}$ is calculated using the method presented in [20]. Also, let $d_{p_{ij}}$ denote the processing delay for rendering and encoding the ith gaming session executed on the jth server. As [11] shows, the processing delay of a gaming session can be modeled using a sigmoid function that accounts for the amount of resources available on the jth server allocated to run the ith gaming session and rp_i, as shown in equation (2).

$$d_{p_{ij}}(rp_i) = \beta_i/(1 + \sigma_i e^{-\gamma_i rp_i}) \qquad (2)$$

In [11], the authors have experimentally measured $d_{p_{ij}}$ for a large set of games and, for each game, derived the β_i, σ_i and γ_i model parameters from regression.

Next, we define an objective function to determine which path and server can minimize the overall delay associated with all gaming sessions running on the datacenter. The objective function consists of the weighted average of the total network and processing delay. Our goal is to minimize equation (3).

$$O_{total}: \sum_{i=1}^{N_g} \sum_{j=1}^{N_s} (((1-\alpha_i)\, d_{p_{ij}} x_{ij} + (\alpha_i) \sum_{k=1}^{N_p} d_{n_{ijk}} z_{ijk})) \qquad (3)$$

Subject to:

i. $\sum_{i}^{N_g} rp_i x_{ij} \leq C_{s_j} \quad \forall j \in \{1, \dots, N_s\}$

ii. $\sum_{j}^{N_s} \sum_{k}^{N_p} z_{ijk} = 1 \quad \forall i \in \{1, \dots, N_g\}$

iii. $\sum_{j}^{N_s} x_{ij} \leq 1 \quad \forall i \in \{1, \dots, N_g\}$

iv. $\sum_{i}^{N_g} \sum_{k}^{N_p} rb_i z_{ijk} \leq BW_{Max} \quad \forall j \in \{1, \dots, N_s\}$

v. $\sum_{j}^{N_s} d_{p_{ij}} x_{ij} + \sum_{k}^{N_p} d_{n_{ijk}} z_{ijk} < D_{Max} \quad \forall i \in \{1, \dots, N_g\}$

vi. $z_{ijk} \leq x_{ij} \; \forall i \in \{1, \dots, N_g\}, \forall j \in \{1, \dots, N_s\}, \forall k \in \{1, \dots, N_p\}$

vii. $z_{ijk} \in \{0,1\}, \forall i \in \{1, \dots, N_g\}, \forall j \in \{1, \dots, N_s\}, \forall k \in \{1, \dots, N_p\}$

viii. $x_{ij} \in \{0,1\} \quad \forall\, i \in \{1, \dots, N_g\}, \forall\, j \in \{1, \dots, N_s\}$

In (3), we define two binary decision variables x_{ij} and z_{ijk}. Binary variable z_{ijk} is equal to 1 if the k^{th} path is chosen among the N_p available paths, and 0 otherwise. Also, binary variable x_{ij} will take value 1 if the j^{th} server is selected to host the i^{th} gaming session, and 0 otherwise.

Constraint (i) ensures that the total allocated processing resources on a server cannot exceed its maximum processing capacity. Constraint (ii) ensures that each gaming session is routed exactly through a single path, and constraint (iii) indicates that each gaming session is hosted at the most on a single server. Constraint (iv) guarantees that the maximum path bandwidth capacity is not exceeded, while constraint (v) restricts the total amount of delay (processing and network delay) to the maximum tolerable delay (D_{max}) within the datacenter [5]. Constraint (vi) ensures that there is available path among the paths connecting game i to game server j.

As mentioned earlier, different types of games have different network delay sensitivities. Hence, we must take this property into account in our method. In equation (3), the priority factor (α_i) is employed to allocate different weights to the processing and network delays of gaming session i based on its type. The priority factor is dependent on the network delay sensitivity of the game. In order to define the priority factor for each type of game, we introduce a utility model for each type using the same methodology introduced in [25]. This utility model presents the impact of the specified video bitrate on the quality perceived by the gamer [27]. In order to build this model, we capture the video output of 30 seconds worth of game play for different game types using FRAPS [1]. Then, we encode the captured video at different bitrates and calculate the Peak Signal to Noise Ratio (PSNR) as an objective measure of the normalized gamers' quality perception.

Fig. 3 illustrates the utility curves for two games in different genres. We define a general parametric utility model by closely fitting the sigmoid function of equation (4) onto our measured data set where l and k are curve-fit parameters derived from regression and r is the bit rate:

$$U(r) = l/(1 + e^{-kr}) \qquad (4)$$

$$\alpha_i = \left[\frac{dU(r)}{dr}\right]_{r=rb_i} = [lke^{-kr}/(1 + e^{-kr})^2]_{r=rb_i} \ (5)$$

Now, the priority factor (α_i) can be computed using the derivative of the utility function of (4) using (5). The rate of change of the utility curve for a game type reflects how quickly its PSNR deteriorates or improves when the bit rate is decreased or increased respectively.

V. EXPERIMENTAL SETUP

In this section, we provide the results of our experiments to evaluate the impact of the proposed LP-based model on different genres of games. We ran our experiments on a Ubuntu version 14.4 box and a Mininet emulator on the Oracle virtual box version 4.3.

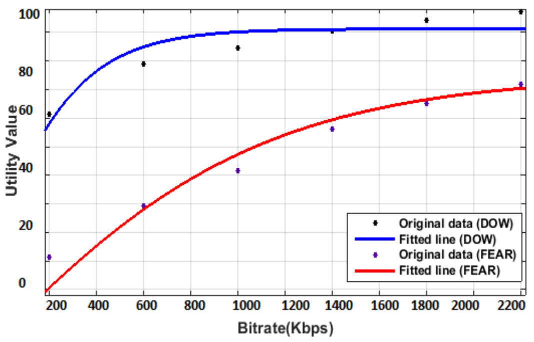


Fig.3. Bitrate Utility Mapping

The Mininet emulator enabled us to create a realistic network experiment with OpenFlow and SDN. Figure 4 shows a toy datacenter network that we employed to examine the performance of our proposed LP model. The toy-network represents a fat-tree topology consisting of five OpenFlow vSwitches (one core switch, two aggregation switches, two ToR switches) and four game servers. We implemented the proposed model in IBM ILOG CPLEX optimizer.

We deployed the OpenFlow controller using the POX controller libraries [14]. The application running on the controller manages network flows and informs the open vSwitches where to send the packets.

In our experiments, we considered a population of 12 end-users that are playing two different game genres. The first six users are playing FEAR (FPS genre) and the second six users are playing Dawn of War (DAW) (RTS genre). These two games exhibit a considerable difference in delay. In our scenario, we used Iperf to generate streams for the simulated gaming flows. We ran Iperf in server mode on game servers A to D. Wireshark was used to monitor the OpenFlow messages, capture, and analyze the streamed packets. We assumed a scenario where the paths from the core switch to game servers A and B are servicing other flows and therefore have higher delays. The delays for paths from the core switch to game servers A, B, C and D are defined as 20ms, 15ms, 10ms and 5ms respectively. Also each game server can serve a maximum of four users.

Table 1 shows the model parameters derived from regression and using equation (5) for both types of games. We run our proposed method using the calculated α_i parameter for each game (Table 1).

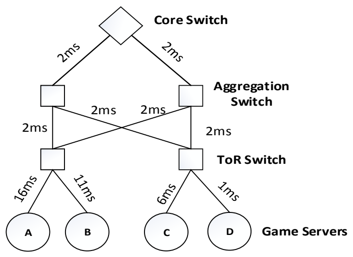
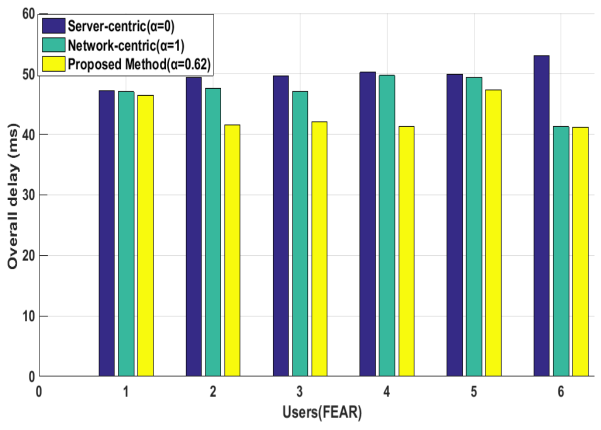
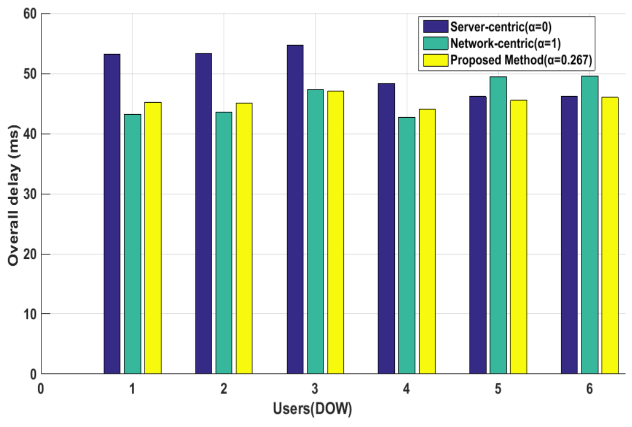


Fig. 4. Data center toy-network topology



(a)



(b)

Fig. 5. a) Overall delay for game FEAR, b) Overall delay for game DAW

We consider three approaches to selecting a server and a path for a gaming session: server centric, network centric, and our proposed method. The server centric approach focuses on minimizing processing delay; on the other hand, the network centric approach strives to minimize network delay. For the server centric approach, the weighted factor α is set to 0 to behave similarly to current server-centric data centers. Moreover, for the network-centric approach where the SDN controller has a centralized view of current network conditions, α is set to 1. By statically setting α to 0 and 1, we can observe the server-centric and network-centric approaches accordingly.

VI. RESULTS

The overall delay obtained for the different types of games is illustrated in Figure 5. The results in Figure 5a show that the average delay variation experienced by players in group 1 (FEAR) using our proposed method is almost 14% and 8% less than the overall delay experienced in the server-centric and network-centric methods, respectively. In addition, Figure 5b shows that the average delay variation experienced by players in group 2 (DAW) using our proposed method is almost 7% and 2% less than the overall delay experienced in the server-centric and network-centric methods, respectively. These improvements stem from the fact that our proposed method considers the delay of paths in the server selection process, whereas the server centric approach does not take into consideration the delay of paths connecting servers to the core switch. Furthermore, in our proposed method, the game servers whose path to the core switch exhibits lower delay are assigned games with higher delay sensitivity. The only time when our method performs slightly worse than the network centric approach is for players 1, 2, and 4 in DAW. This can be explained as follows. Since we defined that each game server can host up to four users, when game servers C and D reach their maximum capacities of 4 for the higher delay sensitive game FEAR, the other servers must be chosen for the remaining games.

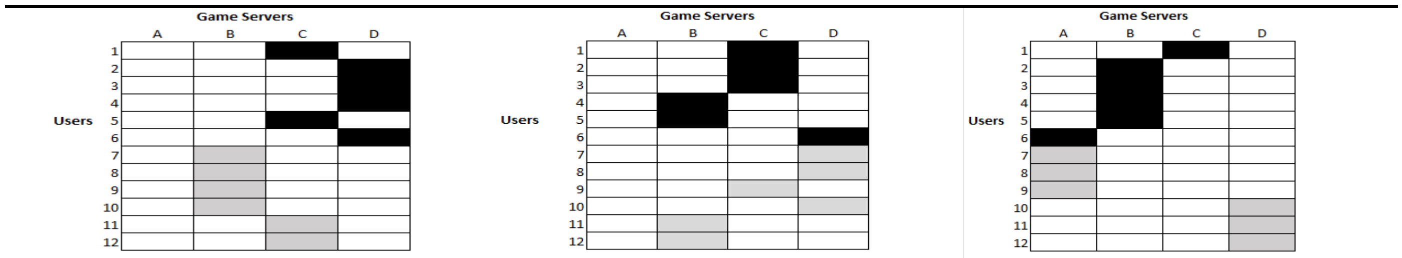
TABLE 1. MODEL PARAMETERS FOR INDIVIDUAL GAMES UTILITY MODEL

	FEAR	DAWN OF WAR
GENRE	***FPS***	***RTS***
l	46.25	43
k	1.789	0.6371
α	0.62	0.267

Therefore, players 1, 2 and 4 in FEAR are assigned to game server B and experience slightly higher delays compared to that of the network centric method. But the delay for all players on average is still 2% lower, as mentioned before.

Figure 6 shows the user assignments on game servers. The black and grey cells represent FEAR and DAW players respectively. In our approach, users playing the game with the higher network delay sensitivity (FEAR) are served by the game severs whose connecting paths have lower delays, while the users playing the game with the lower network delay sensitivity (DAW) are assigned to severs whose paths have higher delays. However, in existing methods, most of FEAR players are assigned to servers whose connecting paths have higher network delays, negatively affecting gameplay in FEAR. Since the server and the network centric approaches select the optimal path and the game server based only on available server or network resources, respectively, they fail to jointly consider both resources to achieve an optimal solution. Moreover, these methods are not able to make a decision based on games requirements.

In our proposed method, by assigning the flows of games that require faster processing time to paths with lower delay, more latitude is available for the processing delay. Consequently, this increases the chance of meeting the overall latency constraint of 100 msec. Moreover, using our proposed method decreases the risk of degradation in the QoE in case of fluctuations in the available bandwidth, since there is more room for network delay. On the contrary, in other methods, most of FEAR gamers are assigned to the servers whose connecting paths have higher network delay, which leads to less room for processing delay. Therefore, in order to meet the overall delay constraint, the virtual machine with higher processing resources must be



a) Proposed Method b) Network-centric c) Server-centric

■ Users 1 to 6 (FEAR) ▨ Users 7 to 12 (DAWn Of War)

Fig. 6. The user assignments on game servers

allocated on those servers, which, in turn, increases power consumption.

VII. CONCLUSION

In this paper we presented a Linear Program optimization-based method for cloud gaming data centers control. Our proposed method optimally assigns game servers to gaming sessions and selects the best communication path within a datacenter. Moreover, our optimization model considers the type of requested game, current server loads, and current path delays. Experimental results showed that the proposed method, on average, outperforms existing algorithms (server-centric and network-centric) by, first, minimizing the overall delay within data centers and, second, assigning the higher delay sensitive games to appropriate game servers.

VIII. REFERENCE

[1] FRAPS, Real-time video capture & benchmarking, available at: http://www.fraps.com// (March 2014).

[2] Beskow, P., Halvorsen, P. and Griwodz, C. Latency reduction in massively multi-player online games by partial migration of game state. In Second International Conference on Internet Technologies and Applications, 153-163, 2007.

[3] Cai, W. and Leung, V. C. Multiplayer cloud gaming system with cooperative video sharing. In 4th IEEE International Conference on Cloud Computing Technology and Science (CloudCom), 640-645, 2012.

[4] Choy, S., Wong, B., Simon, G. and Rosenberg, C. A hybrid edge-cloud architecture for reducing on-demand gaming latency. In Multimedia Systems, 20(5): 503-519, 2014.

[5] Choy, S., Wong, B., Simon, G. and Rosenberg, C. The brewing storm in cloud gaming: A measurement study on cloud to end-user latency. In 11th Annual Workshop on Network and Systems Support for Games (NetGames), 1-6, 2012.

[6] Claypool, M. and Claypool, K. Latency can kill: precision and deadline in online games. In First Annual ACM SIGMM conference on Multimedia Systems (MMSys), 215-222, 2010.

[7] Claypool, M. and Claypool, K. Latency and player actions in online games. In Communications of the ACM, 49(11): 40-45, 2006.

[8] Claypool, M., Finkel, D., Grant, A. and Solano, M. Thin to win? Network performance analysis of the OnLive thin client game system. In 11th Annual Workshop on Network and Systems Support for Games (NetGames), 1-6, 2012.

[9] Dick, M., Wellnitz, O. and Wolf, L. Analysis of factors affecting players' performance and perception in multiplayer games. In 4th ACM SIGCOMM workshop on Network and system support for game,(NetGames), 1-7, 2005.

[10] Hong, H., Chen, D., Huang, C., Chen, K. and Hsu, C. Placing virtual machines to optimize cloud gaming experience. In IEEE Trans. on cloud computing, 3(1):42-53, 2015.

[11] http://www.cisco.com/c/en/us/td/docs/solutions/Enterprise/Data_Center/DC_Infra2_5/DCInfra_1.html#wp1069113.

[12] http://www.geforce.com/whats-new/articles/geforce-grid.

[13] https://openflow.stanford.edu/display/ONL/POX+Wiki.

[14] Huang, Y. and Guerin, R. Does over-provisioning become more or less efficient as networks grow larger? In 13th IEEE International Conference on Network Protocols (ICNP), 2005.

[15] Jarschel, M., Schlosser, D., Scheuring, S. and Hoßfeld, T. Gaming in the clouds: QoE and the users' perspective. In Math. Comput. Model., 57(11-12):2883-2894, 2013.

[16] Kandula, S., Sengupta, S., Greenberg, A., Patel, P. and Chaiken, R. The nature of data center traffic: measurements & analysis. In 9th ACM SIGCOMM conference on Internet measurement,(IMC), 202-208, 2009.

[17] Korkmaz, T. and Krunz, M. Multi-constrained optimal path selection. In INFOCOM 2001. In IEEE INFOCOM, 834-843, 2001.

[18] Lampe, U., Wu, Q., Dargutev, S., Hans, R., Miede, A. and Steinmetz, R. Assessing Latency in Cloud Gaming. Cloud Computing and Services Science. Vol. 453: 52-68, Springer, 2014.

[19] Amiri, M., Alosman, H., Shirmohammadi, S. and Abdallah, M. An SDN Controller for Delay and Jitter Reduction in Cloud Gaming. In ACM Multimedia Conference (MM), Brisbane, Australia, 2015.

[20] Minieka, E. The optimal location of a path or tree in a tree network. Networks,1985, 309-321.

[21] Nelson.R,"Gaikai will be fee-free, utilize 300 data centers in the US".

[22] Semsarzadeh, M., Yassine, A. and Shirmohammadi, S. Video Encoding Acceleration in Cloud Gaming, In IEEE Trans. on Circuits and Systems for Video Technology, 1-14, 2015.

[23] Shi, S., Hsu, C., Nahrstedt, K. and Campbell, R. Using graphics rendering contexts to enhance the real-time video coding for mobile cloud gaming. In 19th ACM international conference on Multimedia (MM), 2011, 103-112.

[24] Thakolsri, S., Khan, S., Steinbach, E. and Kellerer, W. QoE-driven cross-layer optimization for high speed downlink packet access. In Journal of Communications, 4(9):,669-680, 2009.

[25] Tizon, N., Moreno, C., Cernea, M. and Preda, M. MPEG-4-based adaptive remote rendering for video games. In 16th International Conference on 3D Web Technology, 2011.

[26] Video Quality Experts Group. Final report from the Video Quality Experts Group on the validation of objective models of video quality assessment, Phase II (FR_TV2).