Measurement of Cloud-based Game Streaming System Response to Competing TCP Cubic or TCP BBR Flows

Xiaokun Xu
Worcester Polytechnic Institute
Worcester, MA
xxu11@wpi.edu

Mark Claypool
Worcester Polytechnic Institute
Worcester, MA
claypool@wpi.edu

ABSTRACT

Cloud-based game streaming is emerging as a convenient way to play games when clients have a good network connection. However, high-quality game streams need high bitrates and low latencies, a challenge when competing for network capacity with other flows. While some network aspects of cloud-based game streaming have been studied, missing are comparative performance and congestion responses to competing TCP flows. This paper presents results from experiments that measure how three popular commercial cloud-based game streaming systems – Google Stadia, NVidia GeForce Now, and Amazon Luna – respond and then recover to TCP Cubic and TCP BBR flows on a congested network link. Analysis of bitrates, loss rates and round-trip times show the three systems have markedly different responses to the arrival and departure of competing network traffic.

ACM Reference Format:
Xiaokun Xu and Mark Claypool. 2022. Measurement of Cloud-based Game Streaming System Response to Competing TCP Cubic or TCP BBR Flows. In *Proceedings of the 22nd ACM Internet Measurement Conference (IMC '22), October 25–27, 2022, Nice, France.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3517745.3561464

1 INTRODUCTION

Cloud computing infrastructures combined with high-capacity networks have created the emerging market of cloud-based game systems that stream game content as video down to the player. Systems that capitalize on this opportunity and provide game-streaming services include Sony PlayStation Now, Microsoft xCloud, Google Stadia, NVidia GeForce Now, and Amazon Luna. Cloud-based game streaming as a market is growing rapidly with a value of $865.8 million USD in 2021 and is expected to expand at a compound annual growth rate of 48.2% from 2021 to 2027 [14].

Cloud-based game streaming differs from traditional computer games in that game-streaming clients do not run full versions of the game engine. Instead, only the cloud-based server processes the relatively heavyweight game and graphics tasks – applying physics, resolving collisions, processing AI, and rending the game frames – streaming the game as video to the game client. This allows the game client to be relatively lightweight, mostly just playing the streamed game frames much as would a video player and sending player input back up to the server. However, the significant disadvantages of cloud-based game streaming are the increased traffic required for the game frame streaming and the added round-trip latency for all player actions. In particular, the bitrate requirements for frequent, high-quality video frames can cause congestion, degrading player quality of experience and impacting co-located network traffic.

Prior work has shown that cloud-based game streaming requires a high capacity network and is sensitive to network latency [2, 9, 20]. While studies have analyzed network traffic for specific cloud systems like Google Stadia, NVidia GeForce Now and Sony PSNow [11, 13, 30], lacking are comparative aspects across systems, especially how cloud-based game streams respond to congestion. This latter aspect, congestion, could be self-induced when the network capacity is insufficient to support their maximum bitrates or co-induced when the cloud-based game streaming competes for capacity with other network flows on the bottleneck link.

Previous work has compared the congestion response for some cloud-based game streaming systems [32] but only for TCP Cubic [15], whereas an alternate TCP congestion control protocol that has gained traction is bottleneck bandwidth and round-trip time (BBR) [6]. While Cubic uses packet losses to adjust window sizes and, hence, sending rates, BBR uses delivery rates and round-trip times to determine the window sizes. As such, flows that compete with TCP BBR often face different network conditions than do flows that compete with TCP Cubic [26]. Moreover, missing in previous work is analysis of how quickly cloud-based game streaming systems

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
IMC '22, October 25–27, 2022, Nice, France
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9259-4/22/10…$15.00
https://doi.org/10.1145/3517745.3561464

respond to flows that join and congest the network and then how those same game systems recover when the flows depart and congestion eases.

This paper presents an analysis of the network congestion response for three commercial cloud-based game streaming systems – Google Stadia, NVidia GeForce Now and Amazon Luna – providing a direct comparison of their bitrates over time and impact on network congestion when competing for scarce capacity with both TCP Cubic and TCP BBR flows over a range of network conditions. To do so, we designed a novel methodology that runs the game systems via a script, playing the same game automatically on each system to ensure similar player actions across runs. The network testbed and experiments are done over the Internet, but are designed to be as comparable as possible by interlacing runs of each game system serially to minimize temporal differences, and by doing 15 runs for each test condition to provide for a large sample.

The results show the three game systems have similar bitrates when faced with capacity constraints, but behave quite differently when TCP flows arrive and also when they depart. When competing for network capacity with TCP Cubic flows, Stadia takes more than its fair share, Luna shares fairly equally, and GeForce defers to the competing flow more than necessary. Results differ considerably when the game systems compete for network capacity with TCP BBR flows, with Stadia being more fair, Luna losing its fair share, and GeForce deferring even more. These trends are generally exacerbated by small queues at the congested router. The time to respond to congestion caused by an arriving TCP flow and then the time to recover when that TCP flow leaves shows Stadia and Luna can be the quickest to adapt, but that their agility depends upon the bottleneck capacity and router queue size, with large queues generally slowing both congestion response and recovery.

The rest of this paper is organized as follows: Section 2 provides related work on the network aspects of cloud-based game streaming and congestion control protocols; Section 3 describes our methodology, including testbed setup and experiment design and parameters; Section 4 analyzes the experimental results; Section 5 mentions limitations and future work; and Section 6 summarizes our conclusions.

2 RELATED WORK

This section presents work related to this paper including: TCP Cubic and TCP BBR (Section 2.1), interaction between congestion control algorithms (Section 2.2) and measurements of cloud-based game streaming systems (Section 2.3).

2.1 TCP Cubic and BBR

Ha et al. [15] develop TCP Cubic as an incremental improvement to earlier congestion control algorithms. TCP Cubic is less aggressive than previous TCP congestion control algorithms in most steady-state cases, but can probe for more bandwidth quickly when needed. TCP Cubic’s window size is dependent only on the last congestion event, providing for more fairness to flows that share a bottleneck but have different round-trip times. TCP Cubic has been the default in Linux, Microsoft Windows, and Apple Mac OS as of: Linux kernel 2.6.19 (in 2007), Windows 10.1709 Fall Creator’s Update (in 2017), Windows Server 2016 1709 update (in 2017), and Mac OS X Yosemite (in 2014).

Cardwell et al. [6] develop the Bottleneck Bandwidth and Round-trip time (BBR) as an alternative to Cubic. TCP BBR uses the maximum bandwidth and minimum round-trip time observed over a recent time window to build a model of the network and set the congestion window size, allowing it to grow up to twice the bandwidth-delay product. BBR has been deployed by Google servers since at least 2017 and is available as a TCP congestion control option for the Linux since kernel 4.9 (end of 2016).

In our experiments, we use the default Linux kernel versions (v5.4) for both TCP Cubic and TCP BBR.

2.2 Interaction Between Congestion Control Algorithms

Cao et al. [5] analyze measurement results of BBR and Cubic over a range of different network conditions. They produce heat maps and a decision tree that identifies conditions which show when there are performance benefits from BBR over Cubic. They find it is the relative difference between the bottleneck queue size and bandwidth-delay product that dictates when BBR performs well.

Ware et al. [26] model how TCP BBR interacts with loss-based congestion control protocols (e.g., TCP Cubic). Their validated model shows TCP BBR becomes window-limited by its in-flight cap which then determines BBR’s bandwidth consumption. Their models allow for predictions of TCP BBR’s throughput when competing with TCP Cubic with less than a 10% error.

Turkovic et al. [23] study the interactions between congestion control algorithms. They measure performance in a network testbed using a “representative” algorithm from three main groups of TCP congestion control – loss-based (TCP Cubic), delay-based (TCP Vegas [4]) and hybrid (TCP BBR) – using 2 flows with combinations of protocols competing with each other. They also do some evaluation of QUIC [19] as an alternative transport protocol to TCP. They observe bandwidth fairness issues, except for Vegas and BBR, and find BBR is sensitive to even small changes in round-trip time.

Miyazawa et al. [21] and Claypool et al. [10] analyze TCP Cubic and TCP BBR flows competing for capacity on a bottleneck link. They measure shared throughputs showing: 1) TCP flows have balanced bitrates intra-protocol (e.g., TCP BBR with TCP BBR), but that 2) TCP flows have imbalanced bitrates inter-protocol (e.g., TCP BBR with TCP Cubic). In particular, bitrate imbalance between TCP Cubic and TCP BBR flows goes through cycles as Cubic's increase until loss and subsequent decrease coupled with BBR's adjustment to the amount of data sent based on latency causes alternation in throughputs.

Our work complements the above work by investigating the interaction with cloud-based game streaming systems, heretofore untested, first TCP Cubic and then TCP BBR.

2.3 Cloud-based Game System Measurements

There are studies analyzing the network performance of early commercial cloud-based game systems, such as OnLive [29] and Gaikai [28]. Manzano et al. [20] collect and analyze network traffic traces from five different games on both OnLive and Gaikai. They find these cloud-based game streaming systems have higher bitrates than do traditional network games. Claypool et al. [9] make more detailed analysis and observations of OnLive network traffic traces and find OnLive has network turbulence more akin to high-definition, live video, with large, frequent packets and high bitrates.

For current systems, Suznjevic et al. [22] measure network traffic for NVidia GeForce Now and find GeForce requires bitrates significantly higher than earlier cloud-based game streaming systems (about 25 Mb/s today compared to 6 Mb/s previously). Xu et al. [30] measure Google Stadia game traffic for several games, showing Stadia has a traffic pattern similar to but still significantly different than streaming video and at much higher rates than previous cloud-based game streaming systems or video (about 19 Mb/s compared to 6 Mb/s). Domenico et al. [11] study the networking for Google Stadia, NVidia GeForce Now and Sony PS Now, finding bitrates up to 45 Mb/s for Stadia and GeForce but only 13 Mb/s for PS Now, and all services are resilient for up to 5% packet loss.

While the above papers are helpful for characterizing network characteristics for cloud-based game streaming systems, they do not measure system congestion response when faced with competing network flows, particularly the response when new flows start to congest the network and the recovery when they stop.

The closest work to our own that we are aware of is from Carrascosa and Bellalta [7] that limits link capacities for Google Stadia during gameplay, finding Stadia adjusts the resolution and/or frame rate in response to a bitrate reduction. However, the experiments conducted do not necessarily represent responses when TCP flows are competing on the network, including specific response to TCP BBR or TCP Cubic, nor are other, non-Stadia systems considered and compared as does our work.

3 METHODOLOGY

To observe the response of cloud-based game streaming systems to competing network flows, we selected three popular commercial systems and a game common to all (Section 3.1), setup a measurement testbed that allowed for controlling congestion conditions (Section 3.3), gathered network traces (Section 3.4), and analyzed the data (Section 4).

3.1 System and Game Selection

We selected three cloud-based game streaming platforms – Google Stadia, NVidia GeForce Now, and Amazon Luna – based on their current and likely future popularity for game players. While Luna and GeForce offer native applications for client-side players, since all three support play via the Google Chrome browser, we use Chrome as the game client for a fair comparison across systems.

For game selection, as for the platform, we sought a game that could be played on each system to allow for a fair comparison. We selected one of the few games available on all: *Ys VIII: Lacrimosa of Dana* (Nihon Falcom, 2016) – a third person action/exploration game. In our experiments, each Ys run, the game loads the same map and during gameplay, three characters (one controlled by the player) fight enemies for 10 minutes.

Since gameplay visuals (i.e., what is streamed to the client and the player sees) depend upon the player's actions, we developed innovative scripts to play the game automatically, thus providing identical, repeatable gameplay conditions across runs and across platforms. Our scripts automatically open the game (with input appropriate for each system), load the same game map, and then play the game automatically as might a human player. The script executes player actions, including jump, run, attack, cast abilities and camera rotation, at a frequency and pattern that a human player does (although not necessarily in response to what is happening on the screen). This means the same actions can be repeated exactly across all runs.

3.2 Network Conditions

Our goal is to assess the response and recovery for the cloud-based game streaming systems considering congestion arising from both network capacity limits and competing traffic, and explore the difference in adaptation when facing TCP Cubic versus TCP BBR flows. The network capacity limits alone allow comparison of system responses to self-induced congestion arising from various "last-mile" network conditions provided, say, by an Internet Service Provider (ISP), as well as provide baseline performance for constrained conditions without competing TCP flows. Adding competing traffic allows comparison of system congestion responses and recovery for co-induced congestion caused by the presence of other network flows on the bottleneck link. We consider link capacities that are: 1) above the maximum required for each system, but that are less than twice the needed capacity when competing with another flow, 2) right at the maximum capacity required, and 3) 40% below the maximum capacity required.

The dynamics of many congestion control algorithms (e.g., TCP) are influenced by the size of the queue at the bottleneck router [10]. A general rule of thumb is that the bottleneck link's buffer queue size should be a multiple (typically 2x) of the product of the bottleneck capacity and the round-trip delay, otherwise known as the bandwidth-delay product (BDP) – i.e., the BDP is computed by taking the link capacity (bottleneck) in bits per second and multiplying it by the round-trip time (delay) in seconds. Other guidelines suggest a "good" queue size is $(BDP)/\sqrt{n}$, where n is the number of flows at the bottleneck link. However, there are also routers that have considerably larger buffers, a phenomena known as "buffer bloat" [12]. We consider a range of queue sizes, including those that are: 1) shallow, at about one-half the BDP, 2) typical, at about 2x the BDP[1], and 3) bloated, at about 7x the BDP.

3.3 Measurement Testbed

Figure 1 depicts the general setup for our measurement testbed. Our testbed automatically plays the game YS via Chrome on the game client depicted in the figure, connecting through our custom router to the appropriate cloud-service provider (one of Google Stadia, NVidia GeForce Now or Amazon Luna). For experiments with competing traffic, the bottleneck link (from the router down to the clients) is shared by an iperf client that does bulk-downloads from an iperf server using TCP. The game client has a PC running Windows 10 Pro, connecting to the cloud-based game streaming service via Chrome version 98.0.4758.102 (64-bit). The PC hardware is an Intel i7 eight-core CPU @ 2.0 GHz with 64 GB RAM with a 1 Gb/s Ethernet NIC. The PC has an LED monitor with 1920x1080 pixels running at 60 Hz. The iperf client and server are both Alienware PCs each with an 8-core Intel i7-4790K CPU @ 4 GHz with 16 GB RAM running Ubuntu 20.04 LTS, Linux kernel version 5.4, and connect with 1 Gb/s Ethernet NICs.

[1]Some guidelines are for queues to be 1x BDP, but others are larger based on delay [3], hence we settle on 2x BDP.

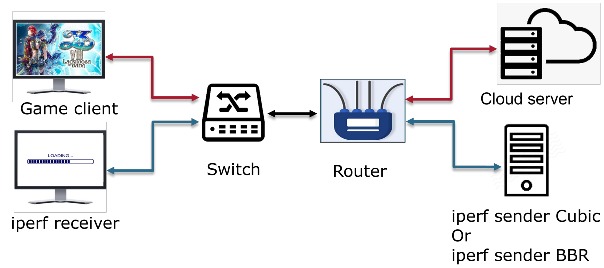


Figure 1: Measurement testbed.

The game client and iperf client PCs connect via a 1 Gb/s switch to a Raspberry Pi 4 configured to act as a network router. The Pi has a 5 GHz 64-bit quad-core CPU with 8 GB of RAM and runs Ubuntu 20.04 LTS, Linux kernel version 5.4, using `tc` [27] and `netem` to constrain the network capacity and add delay. Wireshark is used to gather all network traces for throughput analysis, gathering the game streaming traffic on the router and the iperf TCP traffic on the client. All three game-streaming systems use UDP as the transport protocol and the iperf server uses either TCP Cubic and TCP BBR (v1.0), as appropriate.

Examples of a tc-netem commands run on our router are:

```
tc qdisc add dev eth0 root handle 1: \
  netem delay 4ms
tc qdisc add dev eth0 parent 1: handle 2: \
  tbf rate 15mbit burst 1mbit limit 510kbit
```

The first command adds delay to the system (used to make sure all systems have the same round-trip times) and the second command sets the capacity limit and buffer size.

The router connects to the Internet via our campus network. As a baseline measure of throughput, Google's `M-Lab` Internet speed test consistently shows the campus network through the router to our client PC has downstream bitrates over 900 Mb/s and upstream bitrates over 200 Mb/s. These rates are well-beyond what the streaming services require – i.e., our campus network is not the bottleneck. According to the IP addresses observed and the server location information released by the platforms, the game servers used in our experiments are all on the U.S. east coast, physically near our university in New England.

Based on `ping` measurements from our client, the Stadia servers have an average round-trip time of 11.9 ms, GeForce servers 4.5 ms and Luna servers 16.4 ms. For equal comparison across systems, our router adds 4.5 ms round-trip delay to Stadia, 12 ms to GeForce and 15 ms for iperf to provide about a 16.5 ms round-trip time for all. While a 16.5 ms may be a better round-trip time than that of many residential connections, our focus is on a comparison of congestion response, not necessarily the quality of the individual connections.

3.4 Experiments

Our pilot studies determined 3 minutes of gameplay provided for a steady state for bitrate for cloud-based game streams. Our Wireshark traces begin after the game is being played (i.e., loading, menus, etc. are *not* included — just gameplay).

We measured the steady-state bitrates for our systems (Stadia, GeForce and Luna) and selected game (Ys) on an unconstrained network, and the averages are shown in Table 1 with the standard deviation in the parenthesis. Based on these rates, we tested 3 network conditions: a "good" connection with a capacity limit of 35 Mb/s which is above the baseline bitrates, a "normal" connection with a capacity limit of 25 Mb/s which is right at the baseline bitrates, and a "bad" connection with a capacity limit of 15 Mb/s which is below the baseline bitrates. We verified a solo iperf flow can saturate the link on our testbed at all three capacities with a 16.5 millisecond round-trip time.

Table 1: Game system bitrates without capacity constraints or competing traffic. Units are Mb/s. Mean values are reported with standard deviations in parentheses.

System	Bitrate (Mb/s)
Stadia	27.5 (2.3)
GeForce	24.5 (1.8)
Luna	23.7 (0.9)

Additional scripts automatically: 1) connect to the router to: a) set the queue size and bottleneck capacity limit, as appropriate, and b) launch Wireshark; 2) launch a `ping` command from the client to the game server[2]; and 3) start `presentmon`[3] to record frame rates at the client.

In summary, for each round, the fully-automated experiment procedure is:

(1) Connect to the iperf server and set the TCP congestion control algorithm to Cubic or BBR, as appropriate.
(2) Start the game in the browser and wait for the game to load.
(3) Connect to the router to set the bitrate, delay and queue size and start Wireshark.
(4) Initiate a `ping` from the client to the appropriate game server and start `presentmon`.
(5) Run the script on the game client which launches and then plays the game Ys.
(6) After 3 minutes, start iperf on the iperf client.
(7) Continue the script which plays the game Ys for 3 more minutes, then stop iperf.
(8) Continue the script which plays the game Ys for a 3 final minutes.
(9) Close the game and all data collection tools and reset the router to the unconstrained conditions.
(10) Repeat the above procedure for each of the three systems (Stadia, GeForce and Luna.
(11) Repeat the above procedure for both of the TCP congestion control algorithms (Cubic and BBR).

We repeat the above procedure 15 times for each network condition (capacity constraint and router queue size combination), cloud-based game streaming system and TCP congestion control algorithm. Since Internet conditions from the campus network to the game servers can change over time, we stripe across game service to keep system comparisons as temporally close as possible. For consistency, all this is done by the scripts automatically, without manual intervention. Thus, the order of experimental runs through the parameters from outer loop to inner loop is: [1 to 15 iterations] [Cubic, BBR congestion control] [B35, B25, B15 capacity constraint] [7x, 2x, 0.5x router queue size] [Stadia, GeForce, Luna game system].

A complete run of all systems and all iterations takes about 48 hours providing for performance that accounts for any time-of-day affects. Data was gathered for two consecutive weekdays in March 2022.

Table 2 provides a summary of the key experimental parameters.

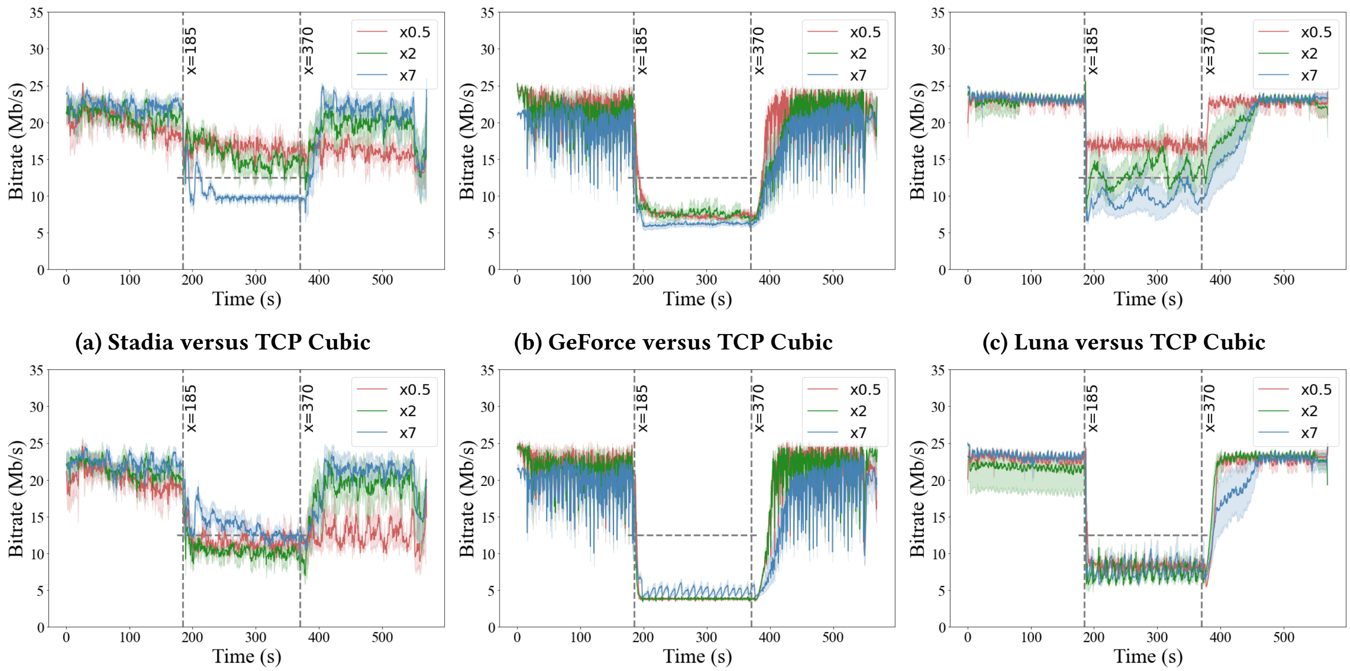
Table 2: Experimental parameters.

Game system	Stadia, GeForce, or Luna
Game	Ys VIII: Lacrimosa of Dana
Capacity limit	15, 25, or 35 Mb/s
Queue size	0.5x, 2x, or 7x BDP
Competing TCP flow	Cubic or BBR
Trace length	9 minutes (3 with iperf)
Iterations	15 runs per condition

4 ANALYSIS

This section compares the different cloud-based game streaming systems with capacity and queue limits, both with and without competing TCP flows, either TCP BBR or TCP Cubic, considering: 1) game streaming bitrates (Section 4.1); 2) bitrate fairness combined with a measure of adaptiveness to congestion (Section 4.2); and 3) indicators of player quality of experience (Section 4.3).

[2]Identified automatically in a script via the Wireshark trace.
[3]https://github.com/GameTechDev/PresentMon



(a) Stadia versus TCP Cubic (b) GeForce versus TCP Cubic (c) Luna versus TCP Cubic

(d) Stadia versus TCP BBR (e) GeForce versus TCP BBR (f) Luna versus TCP BBR

Figure 2: Game system bitrate versus time with a 25 Mb/s capacity constraint and a simultaneous iperf TCP flow from 185s to 370s. Each line shows a separate run with a different bottleneck queue size (0.5x, 2x, or 7x) in multiples of the bandwidth delay product (BDP).

4.1 Bitrates

We start analysis with a bitrate comparison (computed every 0.5 seconds) of each cloud-based game streaming system for each queue size (0.5x, 2x and 7x BDP) where the competing TCP flow runs for 3 minutes in the middle of the 9 minute run. Figure 2 depicts the results for the 25 Mb/s capacity constraint. The top row is for TCP Cubic and the bottom row is for TCP BBR. The left column is Stadia, the middle is GeForce and the right is Luna. For each graph, the x-axis is gameplay time, in seconds, and the y-axis is the measured bitrate, in Mb/s. The mean bitrate for the game system is shown with a colored line with the shading depicting 95% confidence intervals across the 15 runs. There is one line for each queue size: 0.5x BDP - red, 2x BDP - green, and 7x BDP - blue. The left vertical dashed line at 185s shows when the iperf TCP flow starts and the right vertical dashed line at 370s shows when the flow stops. The horizontal dashed line in the middle provides a visual reference for a fair share of 1/2 the bottleneck capacity limit (12.5 Mb/s in this scenario).

Before the competing flow arrives (i.e., up to time 185s), the three systems have similar maximum bitrates near the capacity limit, Luna having the least bitrate variation and GeForce the most. When the iperf TCP flows arrive (time 185s), bitrates for all three systems decrease, indicating they respond to the presence of other traffic competing for the available capacity and, similarly, the systems recover to their original bitrates sometime after the TCP flows leave (after time 370s). Note, however, that the bitrate when the TCP flow is running differs across the three systems – GeForce's bitrates are noticeably lower than the fair share of the capacity, whereas Luna's and Stadia's bitrates are around the fair share. When competing with TCP Cubic, the Stadia and Luna bitrates depend upon the bottleneck queue size, with a larger queue (7x) resulting in a lower bitrate by the game system than with a small queue (0.5x). When competing with TCP BBR, the game system bitrates are mostly independent of the queue size, but Stadia shares the capacity more fairly than it does with TCP Cubic and Luna gets less of the capacity than it does with TCP Cubic. Note, too, that the time it takes the game systems to adjust their bitrates to TCP flow arrivals at 185s is generally much shorter than the time it takes the game systems to recover to their original bitrates after the TCP flows leave at 370s. A summary analysis of the combined response-recovery times is provided in in Section 4.2.

We next analyze the bitrate difference between each game system and the competing iperf TCP flow for each network condition from time 220s to 370s, deliberately not computing

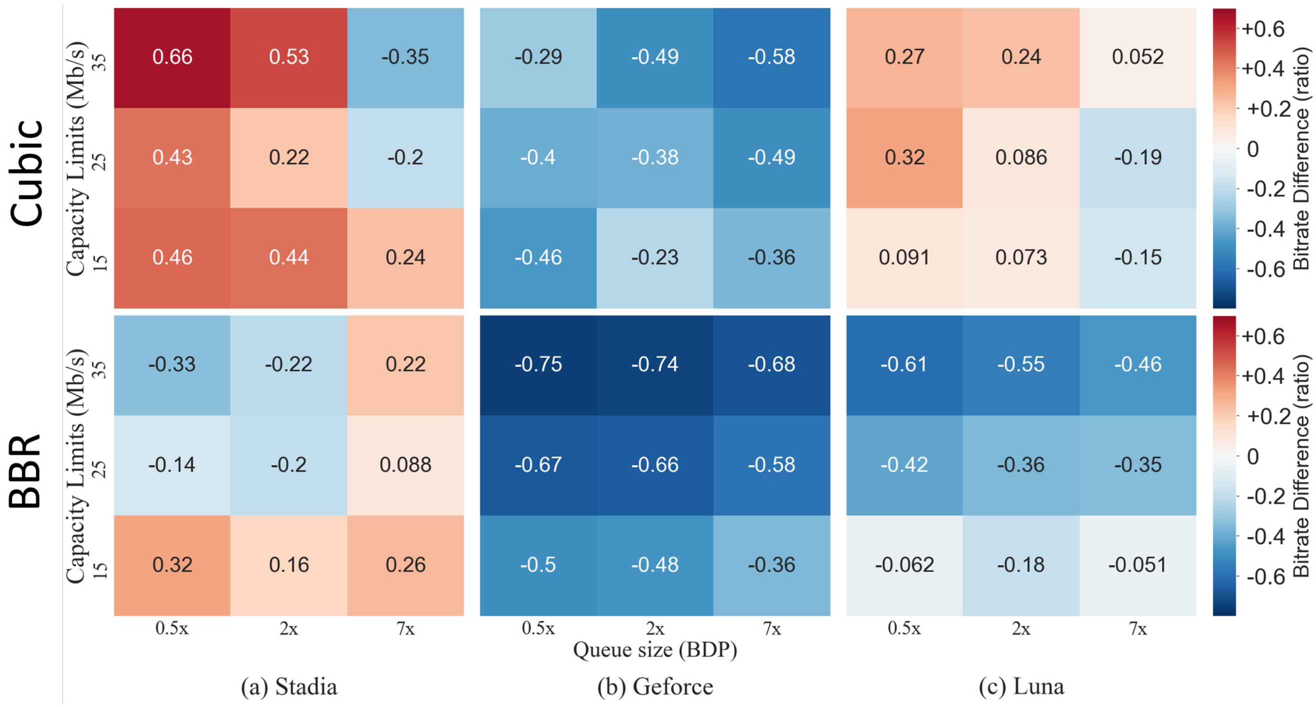


Figure 3: Ratio of bitrate difference (i.e., difference ÷ capacity) for a game system competing with a TCP Cubic flow or a TCP BBR flow.

the bitrates during the initial response to congestion, which are analyzed separately in Section 4.2.

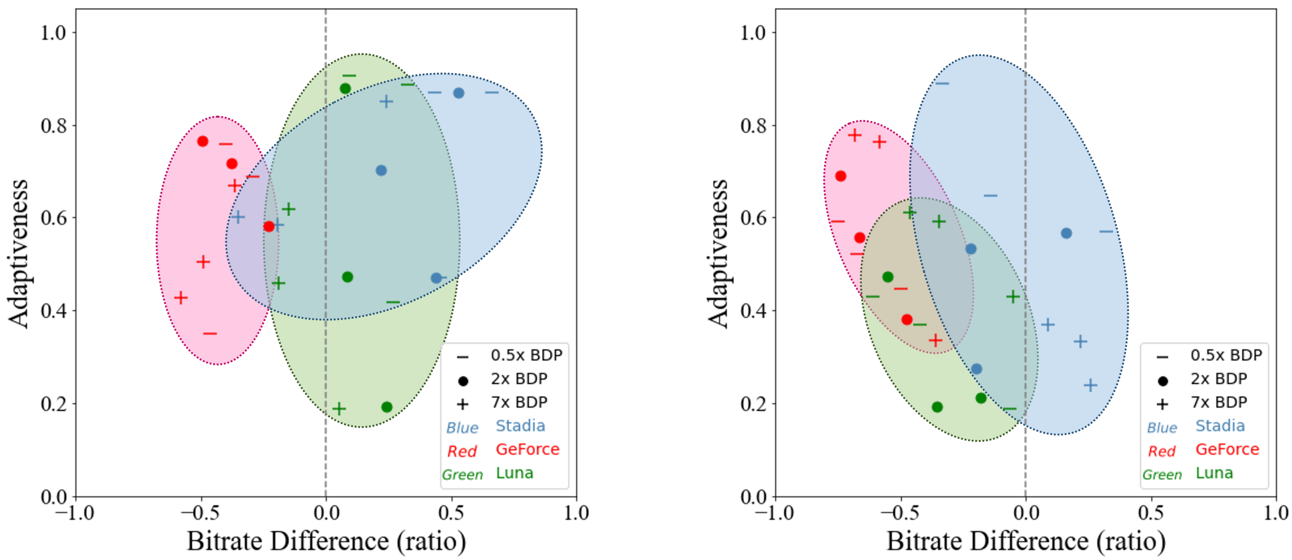
Figure 3 uses heatmaps to depict the results. The top row of heatmaps is for the game systems competing with TCP Cubic and the bottom row with TCP BBR. There is one large box for each game system (Stadia on the left, GeForce in the middle, and Luna on the right), with the smaller boxes within each representing one network condition – 35, 25, and 15 Mb/s capacities as rows and 0.5x, 2x and 7x BDP queues as columns. The numbers in the boxes are the average difference in throughput for the game system minus the competing TCP flow, shown normalized by the capacity (thus ranging from -1 to 1). The warm, red tones show where the game system has a higher bitrate than the TCP flow and the cool, blue tones where the game system has a lower bitrate.

Visually, when competing with TCP Cubic, GeForce is entirely "cool" and always gets less than its fair share of the capacity. In contrast, Stadia and Luna have mostly "warm" areas where they get more than their fair share, with Stadia having several "hot" areas, the "hottest" for a small queue (0.5x BDP) and high capacity (35 Mb/s). However, both Stadia and Luna have two "cool" areas with less than their fair share for large queues (7x BDP).

When competing with TCP BBR, GeForce is still entirely "cool," but the colors are darker than for Cubic, which means a competing TCP BBR flow gets more capacity than does a competing TCP Cubic flow, and generally the smaller queues and higher capacities have "cooler" areas. Luna with BBR is also entirely "cool" in visible contrast to Luna with TCP Cubic which is "warmer". The "coolest" is for a small queue (0.5x BDP) and high capacity (35 Mb/s). Stadia with BBR has both "warm" and "cool" areas, and when compared to Stadia with Cubic most of the "heat" has settled, with the exception of the large queue (7x BDP) which is "warmer" for BBR. This last change is likely because BBR caps its sending rate to twice the computed BDP which, in turn, limits queuing at the router that happens with TCP Cubic (see Section 4.3).

4.2 Adapativeness and Fairness

We next analyze how quickly a game system adjusts its bitrate when the TCP flow arrives and, when the flow departs, how quickly the game system adjusts its bitrate back to the initial level. We call the former *response time* and the latter *recovery time*. To determine response time, we calculate the mean original bitrate before the TCP flow arrives (from 125s to 185s) and the mean bitrate for one minute when the game system stabilizes, having adjusted to the TCP flow (from 310s to 370s). Response time is the number of seconds it takes the game system from time 185s until the average bitrate is



(a) Game system versus TCP Cubic

(b) Game system versus TCP BBR

Figure 4: Adaptiveness versus fairness. Adaptiveness is the average of the normalized response and recovery times, flipped so higher is better. Fairness is the ratio of the bitrate difference between the game system and the competing TCP flow – 0 is the most fair, positive means the game system has a higher bitrate, negative means the competing TCP flow has a higher bitrate.

within one standard deviation of the adjusted bitrate. Similarly, recovery time is the number of seconds it takes the game system from time 370s until the bitrate is within one standard deviation of the original bitrate.

We combine[4] response time and recovery time into one measure of adaptiveness (A):

$$A = \frac{1}{2}(1 - \frac{C}{C_{max}}) + \frac{1}{2}(1 - \frac{E}{E_{max}})$$

where C is the response time to contract the game system bitrate when a TCP flow arrives and E is the recovery time to expand the game system bitrate to the previous level when the TCP flow departs. C_{max} and E_{max} are the maximum observed response time and recovery time, respectively, to normalize the results. Adaptiveness ranges from 0 to 1 and oriented so 0 is the worst (least adaptive) and 1 is the best (most adaptive).

Fairness is computed as the difference in throughput for the game system minus the competing TCP flow, shown normalized by the capacity (thus ranging from -1 to 1). A 0 indicates equal sharing of the link capacity between the game system and a competing TCP flow, a -1 indicates the competing TCP flow gets all the capacity, and a +1 indicates the game system gets all the capacity.

Figure 4 shows scatter plots of adaptiveness versus fairness, with Figures 4a and 4b showing the game system competing with TCP Cubic and TCP BBR flows, respectively.

For each graph, the x-axis is the ratio of bitrate difference, and the y-axis is the adaptiveness (the average of the response and recovery times, each normalized by the maximum across the three tested systems and flipped so higher is better).

Each point is a single game system and network configuration, with **-** indicating a 0.5x BDP queue, **•** a 2x BDP queue, and **+** a 7x BDP queue. The encompassing colored circles represent the general areas of operation (Stadia - blue, GeForce - red, and Luna - green). The grey dashed line in the middle ($x = 0$) is where both the game system and the competing TCP flow get an equal share of the capacity.

From the graphs, GeForce always gets less of the link capacity (the pink ovals are to the left of the vertical center line) than does the competing TCP flow and GeForce has medium adaptiveness (ranging from about 0.35 to 0.8). Stadia has generally the best adapativeness, being slightly better when competing with TCP Cubic than when competing with

[4] A breakdown of the individual response and recovery times is available in our technical report [31].

Table 3: Round-trip time (ms) without a competing TCP flow.

	BDP 0.5x			BDP 2x			BDP 7x		
Capacity	Stadia	GeForce	Luna	Stadia	GeForce	Luna	Stadia	GeForce	Luna
15 Mb/s	16.0 (1.7)	16.8 (1.5)	17.2 (2.1)	19.2 (8.8)	17.4 (8.1)	17.3 (8.6)	20.0 (11.5)	20.1 (9.5)	19.7 (9.4)
25 Mb/s	16.6 (2.2)	16.8 (1.6)	17.0 (1.5)	20.6 (10.2)	16.9 (5.8)	17.2 (5.6)	22.0 (13.9)	20.7 (9.5)	18.1 (10.9)
35 Mb/s	17.1 (1.4)	18.2 (1.8)	16.4 (1.6)	20.6 (10.5)	18.6 (5.2)	16.8 (5.7)	20.9 (12.3)	20.9 (9.6)	17.3 (9.5)

TCP BBR. Stadia is slightly unfair when competing with TCP Cubic, but Stadia is the most fair when competing with TCP BBR (the blue ovals are slightly to the right and centered on the vertical center line, respectively). Luna is the most fair (the green oval is centered on the vertical center line) with a range of adaptations (high to low) when competing with TCP Cubic, but is visibly less responsive when competing with TCP BBR and also generally unfair in that the TCP BBR flow gets more capacity (the green oval is to the left of the vertical center line).

4.3 Indicators of Quality of Experience

Indicators of the player quality of experience (QoE) of a cloud-based game are delay (which manifests when the bottleneck router queue becomes filled with excess traffic) and frame rate.

Another indicator of QoE is loss rate [24], where lost game frames can degrade visual quality for the player. For all network conditions and all game systems, loss rates are near 0 when there is no competing TCP flow. Even with a competing TCP flow, loss rates are well under one percent in all cases, albeit slightly higher for small queues and when competing with TCP BBR – the latter does not treat loss as an indicator of congestion. Tables of the loss rates for all systems and each network condition, both with and without competing TCP flows, can be found in our technical report [31].

Games played over a cloud-based game streaming system are sensitive to delay since all player input needs to be sent to the server, acted upon by the game engine, rendered, and then sent back to the client before the player can see the outcome of their actions. There are inherent delays in the end systems – e.g., input delay from the mouse and monitor on the client, game engine updates and rendering on the server – but the network round-trip time is "extra" delay that would not be present if the game was played entirely locally. Table 3 (without a competing TCP flow) and Table 4 (with a competing TCP flow, either Cubic or BBR) have the round-trip times for the 3 systems for each condition (capacity, queue size and congestion control algorithm). Each value is the mean for the 3 minutes gameplay with standard deviations shown in parentheses.

For all systems, when there is no competing TCP flow, the round-trip times are low, near minimal (about 16 ms) for small queues, but increasing by about 25% for Stadia and GeForce for larger queues. The small differences in delays between systems may be noticeable to users, but are small enough to not appreciably affect performance or QoE [1]. These round-trip times do not reach even the delays that would be caused by 0.5x BDP queueing suggesting that, unlike TCP Cubic, the systems themselves do not saturate available capacity until there is loss.

When there is a competing TCP Cubic flow, the round-trip times are consistently at the limit dictated by the queue size both for TCP Cubic and TCP BBR, which makes sense given that TCP Cubic generally increases sending rates until there is packet loss. This illustrates how large router queues in the presence of competing flows during congestion result in added delay for game-streaming systems. Note that this holds for competing TCP BBR flows, as well, but for game systems competing with TCP BBR over large queues (7x BDP), all three systems have about half the round-trip times experienced when competing with TCP Cubic. This difference is explained by the BBR protocol that caps the maximum congestion window to twice the BDP which, in turn, limits the number of packets that can build up in the queue to about the BDP. For the large queues (7x BDP), the delay differences when competing with TCP Cubic versus TCP BBR (about 110 ms vs. 55 ms) are meaningful and would result in about a 10% decrease in QoE for users [24].

Frame rate is another key indicator of the game quality, where higher frame rates can improve player performance and are generally associated with a better player quality of experience (QoE) [8, 17, 18, 33]. Without a competing TCP flow, the frame rates of all systems for all network conditions are near 60 f/s, so only frame rate analysis with competing TCP flows is presented. Table 5 shows the mean frame rates for the 3 systems under each network condition, with standard deviation in parentheses, computed for the 3 minutes when the competing TCP flow runs. The game system competing with TCP Cubic is shown adjacent to TCP BBR.

In general, game system frame rates are independent of the capacity. For both Cubic and BBR, frame rates are higher with greater capacity and larger queues. When competing with TCP Cubic, frame rates are generally high (50+ f/s), although frame rates tend to be lower (about 50 f/s) for Stadia with small (0.5x BDP) and medium (2x BDP) queues than the frame rates for Luna or GeForce with the same conditions. When competing with TCP BBR, with small (0.5x BDP) and medium (2x BDP) queues, frame rates degrade for all game systems. Stadia and Luna in particular have frame rates

Table 4: Round-trip time (ms) with a competing TCP flow.

	BDP 0.5x						BDP 2x						BDP 7x					
	Stadia		GeForce		Luna		Stadia		GeForce		Luna		Stadia		GeForce		Luna	
Capacity	Cubic	BBR	Cubic	BBR	Cubic	BBR	Cubic	BBR	Cubic	BBR	Cubic	BBR	Cubic	BBR	Cubic	BBR	Cubic	BBR
15 Mb/s	17.4 (2.5)	20.7 (2.8)	17.7 (2.3)	24.7 (8.3)	17.7 (2.1)	23.4 (3.3)	40.1 (6.8)	45.8 (10.1)	38.3 (5.7)	44.2 (7.9)	40.0 (5.7)	41.8 (9.2)	111.6 (12.4)	77.8 (15.9)	110.7 (10.9)	59.8 (14.5)	108.9 (14.7)	57.1 (14.8)
25 Mb/s	17.8 (2.6)	20.7 (2.6)	17.8 (2.4)	24.7 (8.3)	18.6 (3.8)	23.4 (3.3)	40.0 (6.0)	44.2 (7.5)	39.3 (5.9)	42.4 (8.3)	40.6 (5.8)	39.6 (9.1)	110.6 (9.3)	55.9 (15.7)	109.2 (10.8)	54.9 (12.9)	106.1 (16.7)	52.0 (12.8)
35 Mb/s	18.3 (2.9)	20.2 (5.4)	19.5 (3.7)	20.6 (2.5)	17.7 (2.3)	24.3 (7.3)	39.0 (8.1)	43.5 (8.3)	42.0 (6.8)	39.7 (10.1)	40.8 (5.1)	37.0 (7.5)	110.6 (12.5)	54.4 (10.7)	114.5 (14.0)	49.8 (10.8)	109.9 (10.9)	40.3 (9.7)

Table 5: Frame rate (f/s) with competing TCP flow.

	BDP 0.5x						BDP 2x						BDP 7x					
	Stadia		GeForce		Luna		Stadia		GeForce		Luna		Stadia		GeForce		Luna	
Capacity	Cubic	BBR	Cubic	BBR	Cubic	BBR	Cubic	BBR	Cubic	BBR	Cubic	BBR	Cubic	BBR	Cubic	BBR	Cubic	BBR
15 Mb/s	50.8 (1.83)	38.8 (1.50)	57.9 (0.46)	51.7 (1.10)	53.7 (1.02)	22.3 (3.14)	50.9 (0.27)	40.2 (1.14)	59.1 (0.22)	53.9 (0.43)	56.5 (0.94)	44.8 (1.04)	56.9 (0.43)	57.5 (2.07)	59.8 (0.09)	59.5 (0.35)	58.1 (0.59)	59.3 (0.15)
25 Mb/s	50.5 (0.65)	40.2 (1.47)	57.9 (0.28)	54.9 (0.21)	53.7 (0.66)	39.0 (1.53)	51.0 (0.61)	40.5 (0.52)	59.3 (0.16)	55.6 (0.34)	56.9 (0.18)	53.9 (1.09)	58.7 (0.22)	59.4 (0.08)	59.9 (0.06)	59.6 (0.05)	58.1 (0.11)	59.5 (0.19)
35 Mb/s	51.0 (1.94)	41.2 (1.83)	59.9 (0.15)	56.5 (0.37)	56.9 (0.36)	47.3 (0.71)	51.5 (1.73)	47.2 (1.87)	59.9 (0.05)	57.2 (0.19)	58.0 (0.22)	57.5 (0.34)	58.8 (0.30)	58.9 (0.81)	59.9 (0.03)	59.7 (0.05)	58.6 (0.21)	59.7 (0.14)

around 40 f/s when competing with BBR, and Luna has an average frame rate as low as 22 f/s with 15 Mb/s capacity. In contrast, GeForce has more resilient frame rates, always averaging above 50 f/s. For BBR, compared to Cubic, GeForce has slightly lower frame rates. Stadia and Luna have much lower frame rates, and Luna has the lowest frame rates of only 22.3 f/s at low capacity (15 Mb/s) and a small queue (0.5x BDP).

5 LIMITATIONS AND FUTURE WORK

Our experiments are for one game only and prior work has shown that the bitrates for different games on the same cloud-based game streaming system can vary considerably [30]. Future work should see if the comparative differences illustrated here hold for other games, as well. Similarly, whether the results hold for other prominent cloud-based game streaming systems such as those by Microsoft or Sony is not known and could be studied.

Our router uses only a drop-tail queue, whereas Active Queue Management approaches (AQM) that signal congestion earlier (e.g., Flow Queue CoDel [16]) might be considered.

As analyzed in Section 4.3, beyond bitrates, round-trip times and frame rates provide some indication of player experience, but Quality of Experience (QoE) is impacted by other aspects, such as frame resolution and visual quality, as well. Future work might assess and compare QoE across systems and games.

Our system comparisons focus on throughput fairness, but could instead consider harm-based analysis [25] where responses to congestion would be assessed based on their impacts to throughput, loss and round-trip time compared to the impacts the competing traffic has on itself (e.g., how much harm TCP Cubic flows cause to other TCP Cubic flows).

Our congestion scenarios are only a single, bulk-download TCP flow whereas many if not most congestion scenarios are be more varied. In particular, cloud-based game streams may often compete with streaming video (e.g., Netflix) over last-mile residential networks. Our future work is to consider HTTP-based streaming, live video conferencing and multiple flows and mixtures of flows.

6 CONCLUSIONS

Emerging cloud-based game streaming systems hold out the promise of providing a convenient gaming experience for players, as long as the network conditions are sufficient. In particular, streaming computer games need high definition frames sent at high frame rates (typical targets are 60 f/s). This, in turn, requires high bitrates that have the potential to congestion last mile networks, particularly when competing for capacity with other network flows. While network flows use TCP and most of those TCP flows use Cubic [15] as their congestion control algorithm, there are alternate congestion control algorithms in use, such as Bottleneck Bandwidth and Round-trip time (BBR) [6]. This paper compares three commercial systems – Google Stadia, NVidia GeForce Now and Amazon Luna – with repeated runs of the same game across capacity constraints and bottleneck queue sizes, while the game systems compete for capacity with a TCP Cubic or TCP BBR flow.

Analysis of the results show the three systems have similar bitrates near capacity at 25 Mb/s, and even for constrained conditions (lower capacity, smaller queue sizes) none of the three systems has self-induced congestion, keeping packet queuing low and packet loss minimal in the absence of competing traffic. When competing with a bulk-download TCP Cubic flow, Luna generally shares the available capacity fairly, but GeForce defers and lets the TCP flow have about twice what is fair and Stadia dominates taking about twice what is fair. When competing with a bulk-download TCP BBR flow, Stadia generally shares the available capacity fairly, but GeForce reduces its bitrate even more than for TCP Cubic for all network conditions, and Luna, too, has bitrates considerably below what is a fair share. Stadia is generally the most adaptive of the three systems, responding quickly to arriving TCP flows and recovering when the flows depart for both TCP Cubic and TCP BBR. GeForce tends to have the slowest response to arriving TCP flows, particularly for TCP BBR. Luna has fast response and recovers fairly fast for when competing with TCP Cubic, but has greatly reduced recovery times when competing with TCP BBR. Trends for long response/recovery times are generally exacerbated by large bottleneck queues (buffer bloat), whereas trends for unfairness are exacerbated by small queue sizes. In some cases of constrained capacity and small queues, Stadia never responds or recovers to the competing TCP flow, while Luna never recovers from a competing TCP BBR flow once it departs at high capacity.

Large bottleneck queues (buffer bloat) also result in larger delays for the game systems, which is bad for player quality of experience; this effect is more pronounced when the game system is competing with TCP Cubic whereas a competing TCP BBR flow limits the extent to which the bottleneck queue grows. More constrained network conditions also result in lower frame rates displayed by the game system, most noticeably for Stadia. Frame rate decreases are most pronounced when the game system is competing with TCP BBR, with frame rates down from the target 60 f/s to as low as about 22 f/s.

These results provide a better understanding of game system interactions with constrained and competitive network links both for competing TCP Cubic and competing TCP BBR flows and should be useful to better plan for, and hopefully deter, resulting network congestion.

REFERENCES

[1] Maha Abdallah, Carsten Griwodz, Kuan-Ta Chen, Gwendal Simon, Pin-Chun Wang, and Cheng-Hsin Hsu. 2018. Delay-Sensitive Video Computing in the Cloud: A Survey. *ACM Transactions on Multimedia Computing, Communications, and Applications* 13, 3s (2018), 1–29.

[2] Hamed Ahmadi, Sepideh Khoshnood, Mahmoud Reza Hashemi, and Shervin Shirmohammadi. 2013. Efficient Bitrate Reduction using a Game Attention Model in Cloud Gaming. In *Proceedings of the IEEE International Symposium on Haptic Audio Visual Environments and Games (HAVE)*. 103–108. https://doi.org/10.1109/HAVE.2013.6679619

[3] G. Appenzeller, I. Keslassy, and N. McKeown. 2004. Sizing Router Buffers. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication (SIGCOMM)*.

[4] Lawrence S. Brakmo, Sean W. O'Malley, and Larry L. Peterson. 1994. TCP Vegas: New Techniques for Congestion Detection and Avoidance. In *Proceedings of the Conference on Communications Architectures, Protocols and Applications*. London, UK, 24–35.

[5] Yi Cao, Arpit Jain, Kriti Sharma, Aruna Balasubramanian, and Anshul Gandhi. 2019. When to Use and When not to Use BBR: An Empirical Analysis and Evaluation Study. In *Proceedings of the Internet Measurement Conference (IMC)*. Amsterdam, NL.

[6] N. Cardwell, Y. Cheng, C. S. Gunn, S. H. Yeganeh, and Van Jacobson. 2017. BBR: Congestion-based Congestion Control. *Communications of the ACM* 60, 2 (Jan. 2017), 58–66.

[7] M. Carrascosa and B. Bellalta. 2022. Cloud-gaming: Analysis of Google Stadia Traffic. *Computer Communications* (April 2022), 99–116.

[8] Kajal Claypool and Mark Claypool. 2007. On Frame Rate and Player Performance in First Person Shooter Games. *Springer Multimedia Systems Journal (MMSJ)* 13 (2007), 3–17.

[9] Mark Claypool, David Finkel, Alexander Grant, and Michael Solano. 2012. 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.

[10] Saahil Claypool, Mark Claypool, Jae Chung, and Feng Li. 2019. Sharing but not Caring - Performance of TCP BBR and TCP CUBIC at the Network Bottleneck. In *Proceedings of the 15th IARIA Advanced International Conference on Telecommunications (AICT)*. Nice, France.

[11] Andrea Di Domenico, Gianluca Perna, Martino Trevisan, Luca Vassio, and Danilo Giordano. 2021. A Network Analysis on Cloud Gaming: Stadia, GeForce Now and PSNow. *Network* 1, 3 (2021).

[12] Jim Gettys and Kathleen Nichols. 2011. Bufferbloat: Dark Buffers in the Internet: Networks without Effective AQM May Again be Vulnerable to Congestion Collapse. *ACM Queue* 11 (Nov. 2011), 40–54.

[13] Philippe Graff, Xavier Marchal, Thibault Cholez, Stéphane Tuffin, Bertrand Mathieu, and Olivier Festor. 2021. An Analysis of Cloud Gaming Platforms Behavior under Different Network Constraints. In *Proceedings of the 17th International Conference on Network and Service Management (CNSM)*. Izmir, Turkey, 551–557.

[14] IMARC Group. 2022. Cloud Game Market. (Feb. 2022). https://www.imarcgroup.com/cloud-gaming-market [Accessed 24-Feb-2022].

[15] Sangtae Ha, Injong Rhee, and Lisong Xu. 2008. CUBIC: A New TCP-Friendly High-Speed TCP Variant. *ACM SIGOPS Operating Systems Review* 42, 5 (2008).

[16] T. Hoeiland-Joergensen, P. McKenney, D. Taht, J. Gettys, and E. Dumazet. 2018. The Flow Queue CoDel Packet Scheduler and Active Queue Management Algorithm. *IETF Request for Comments (RFC) 8290* (Jan. 2018).

[17] J. Huhti. 2019. *The Effect of High Monitor Refresh Rate on Game Quality of Experience.* Ph.D. Dissertation. University of Vaasa, Vaasa, Finland.

[18] Joohwan Kim, Josef Spjut, Morgan McGuire, Alexander Majercik, Ben Boudaoud, Rachel Albert, and David Luebke. 2019. Esports Arms Race: Latency And Refresh Rate For Competitive Gaming Tasks. *Journal of Vision* 19, 10 (Sept. 2019).

[19] Adam Langley, Alistair Riddoch, Alyssa Wilk, Antonio Vicente, Charles Krasic, Dan Zhang, Fan Yang, Fedor Kouranov, Ian Swett, Janardhan Iyengar, Jeff Bailey, Jeremy Dorfman, Jim Roskind, Joanna Kulik, Patrik Westin, Raman Tenneti, Robbie Shade, Ryan Hamilton, Victor Vasiliev, Wan-Teh Chang, and Zhongyi Shi. 2017. The QUIC Transport Protocol: Design and Internet-Scale Deployment. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication (SIGCOMM)*. Los Angeles, CA, USA, 183–196.

[20] M. Manzano, J. A. Hernández, M. Urueña, and E. Calle. 2012. An Empirical Study of Cloud Gaming. In *Proceeding of the 11th Annual Workshop on Network and Systems Support for Games (NetGames)*. 1–2. https://doi.org/10.1109/NetGames.2012.6404021

[21] Kouto Miyazawa, Kanon Sasaki, Naoki Oda, and Saneyasu Yamaguchi. 2018. Cycle and Divergence of Performance on TCP BBR. In *Proceedings of the 7th IEEE International Conference on Cloud Networking (CloudNet)*.

[22] Mirko Suznjevic, Ivan Slivar, and Lea Skorin-Kapov. 2016. Analysis and QoE Evaluation of Cloud Gaming Service Adaptation under Different Network Conditions: The Case of NVIDIA GeForce NOW. In *Proceedings of the 8th International Conference on Quality of Multimedia Experience (QoMEX)*. Lisbon, Portugal, 1–6. https://doi.org/10.1109/QoMEX.2016.7498968

[23] Belma Turkovic, Fernando A. Kuipers, and Steve Uhlig. 2019. Interactions between Congestion Control Algorithms. In *Proceedings of the Network Traffic Measurement and Analysis Conference (TMA)*. 161–168. https://doi.org/10.23919/TMA.2019.8784674

[24] A. Wahab, N. Ahmad, M. G. Martini, and J. Schormans. 2021. Subjective Quality Assessment for Cloud Gaming. *MDPI Multidisciplinary Scientific Journal* 4, 3 (Aug. 2021), 404––419.

[25] Ranysha Ware, Matthew K. Mukerjee, Srinivasan Seshan, and Justine Sherry. 2019. Beyond Jain's Fairness Index: Setting the Bar For The Deployment of Congestion Control Algorithms. In *Proceedings of the 18th ACM Workshop on Hot Topics in Networks (HotNets)*. Princeton, NJ, USA, 17–24. https://doi.org/10.1145/3365609.3365855

[26] R. Ware, M. K. Mukerjee, S. Seshan, and J. Sherry. 2019. Modeling BBR's Interactions with Loss-Based Congestion Control. In *Proceedings of the Internet Measurement Conference (IMC)*. Amsterdam, Netherlands,.

[27] Wikipedia contributors. 2020. Tc (Linux). (2020). https://en.wikipedia.org/wiki/Tc_(Linux) [Accessed 27-Jan-2020].

[28] Wikipedia contributors. 2022. Gaikai. (2022). https://en.wikipedia.org/wiki/Gaikai [Accessed 22-Jan-2022].

[29] Wikipedia contributors. 2022. OnLive. (2022). https://en.wikipedia.org/wiki/OnLive [Accessed 15-Jan-2022].

[30] Xiaokun Xu and Mark Claypool. 2021. A First Look at the Network Turbulence for Google Stadia Cloud-based Game Streaming. In *Proceedings of the IEEE Global Internet Symposium (GI)*. Virtual Conference.

[31] Xiaokun Xu and Mark Claypool. 2022. *Measurement of Cloud-based Game Streaming System Response to Competing TCP Cubic or TCP BBR Flows.* Technical Report WPI-CS-TR-22-03. Computer Science Department at Worcester Polytechnic Institute.

[32] Xiaokun Xu and Mark Claypool. 2022. Measurement of the Responses of Cloud-based Game Streaming to Network Congestion. In *Proceedings of the 32nd ACM International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV)*. Athlone, Ireland.

[33] T. Zinner, O. Hohlfeld, O. Abboud, and T. Hossfeld. 2010. Impact of Frame Rate and Resolution on Objective QoE Metrics. In *Proceedings of the 2nd International Workshop on Quality of Multimedia Experience (QoMEX)*. Trondheim, Norway.