Buffer sizing and Video QoE Measurements at Netflix

Bruce Spang, Stanford University, bspang@stanford.edu
Brady Walsh, Netflix, bwalsh@netflix.com
Te-Yuan Huang, Netflix, thuang@netflix.com
Tom Rusnock, Netflix, tom@netflix.com
Joe Lawrence, Netflix, jlawrence@netflix.com
Nick McKeown, Stanford University, nickm@stanford.edu

ABSTRACT

In this paper we report on measurement of Netflix video traffic, to understand how the size of router buffers affects the quality of video seen by the end-user (i.e. video QoE). In certain locations, Netflix streams video over TCP New Reno from racks of servers that are directly connected to large routers, which in turn directly peer with commercial ISPs. We vary the size of the router buffers during periods of persistent congestion, and log metrics such as the number of rebuffering events, video quality, and video play delay. We observe buffers that are too small and too large, both of which worsen video QoE. Our results are for a specific, but we believe interesting, case: a router carrying large numbers of long-lived TCP flows, shedding light on how buffers should be sized by CDNs and video content providers. We find that the generality of our results is limited by the specific internal workings of the VOQ-based chassis routers used in our experiments, which have a different buffer architecture than prior work. We conclude that the routers complicate—and potentially mask—our clear understanding of buffer sizing.

1 INTRODUCTION

Internet routers have buffers to avoid discarding packets during times of congestion. Despite decades of work by researchers and in industry, we still have a poor understanding of the correct size for a router buffer.

The networking community generally believes that a too-large router buffer will increase delay and a too-small buffer will increase loss. However, it is not known if this trade-off is fundamental or simply an artifact of a particular congestion control algorithm. More practically, the broader consequences of changing buffer size are also not well understood. There has been a long line of work on how buffer sizing affects network metrics [2, 3, 7, 9, 10, 12, 17, 18], but relatively little work [11] examining the impact of buffer sizing on application performance.

In this work, we describe the results of a series of buffer sizing experiments we ran in collaboration with Netflix, which begin to shed some light on this question. Netflix is one of the largest source of internet traffic in the world, and has a number of congested links in different locations. We tested a variety of buffer sizes at some of these congested locations, and observed the effects on TCP New Reno and on the video quality of experience.

We find that properly sizing a buffer is both crucial for improving network and video quality of experience, and also quite difficult to do. Video quality of experience depends on a number of factors including: the play delay—or time it takes to start the video, the visual quality of the video streamed by the client, and the number of rebuffers—the times when video playback pauses because a client has no video to stream. We have experimented with both too-small and too-large buffers, both of which can negatively impact video performance—affecting play delay on the order of seconds, decreasing the visual quality of video streamed by 5-10%, and increasing the median rate of rebuffers by 50%.

Our results also show that the question of sizing router buffers is affected by the internal workings of the router. Prior work on buffer sizing assumes that the router is *output queued*; i.e. when a packet arrives, it is immediately placed in a queue at the output port, and its departure time is unaffected by packets destined to different outputs. In our experiments, the routers used combined input and output queueing (CIOQ), with large virtual output queues (VOQs) at the ingress, and a small queue at each output. Arriving packets are initially placed in an ingress VOQ. An internal scheduling algorithm transfers packets from the VOQs to the output queue, approximating the behavior of an output queued switch.

While it is theoretically possible for a CIOQ switch to perfectly emulate an output queued switch [6], in practice the emulation is imperfect, depending on the internal speedup and scheduling algorithm. As we will see later, this imperfection complicates our results, and masks some of the clarity we were seeking in our experiments. For example, we are unable to tell whether a bandwidth-delay product is a good or bad size for a buffer, because we are unable to measure the rate at which buffers are served. We are able to make less specific conclusions, like on the general trend of whether a larger or smaller buffer improves performance. We plan to address these concerns in future work.

To summarize, our main observations are:

(1) For metrics related to TCP New Reno, the behavior matches intuition: packet loss increases and RTT decreases as buffers shrink.
(2) Video performance has a sweet spot in terms of buffer size: buffers can be both too small and too large, and in both cases increase the number of rebuffers and decrease the video quality.

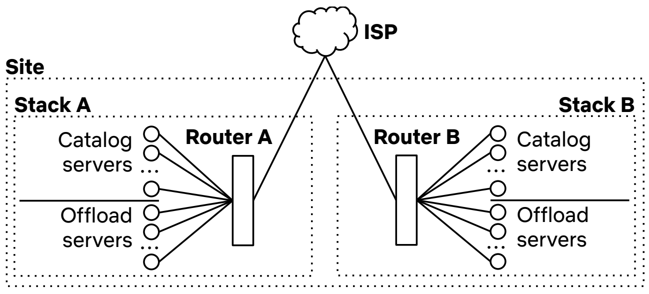

Figure 1: An example site used for experiments. Traffic from the ISP was randomly assigned between the A and B stacks

(3) The ability of the router to emulate perfect output queueing has implications on performance and buffer size.

2 RELATED WORK

Since at least 1990, the rule of thumb for sizing router buffers has been that they need to be at least as large as one bandwidth-delay product (BDP) [12, 18]. The justification for this rule of thumb comes from the buffer size needed for a single TCP Reno flow to keep a bottleneck link fully utilized.

In 2004, Appenzeller et al. [2] argued that as the number of flows increases, the bottleneck link can be kept fully utilized with a much smaller buffer. This is because the aggregate arrival rate concentrates around its expectation. In this setting, they argue that a buffer of size BDP/$\sqrt{N}$ (where N is the number of TCP Reno flows sharing the bottleneck link) is sufficient to keep a link fully utilized. Experiments by Level 3, Internet 2, and on the Stanford campus supported this result [3].

Continuing this line of work, Enachescu et al. [9] showed that if arrivals are distributed according to a Poisson process (either by assumption, multiplexing, or pacing), then the required buffer is small—at most one hundred packets. To reduce the dependence on specific models of TCP, Stanojević et al. [17] propose an adaptive algorithm to shrink the buffer as long as the link utilization is above a predetermined threshold.

After this line of work, one might think that buffers should be reduced as far as possible. In our experiments, we find cases where reducing the sizes of buffers both helps and hurts. Furthermore, we observe interesting QoE effects well before a link becomes under-utilized—which this line of work does not address.

Dhamdhere et al. argue the opposite in [7]. Among other results, they describe a model in which packet loss is proportional to the number of flows using a bottleneck link, so the buffer must grow proportionally to the number of flows in order to limit loss. We also observe similar behavior for loss, which we describe in Section 4.1. However, more loss is not necessarily a bad thing, and we report results from experiments in Section 4.2 where loss significantly increased but quality generally improved.

In the early 2010s, Gettys et al. [10] revisited the question of buffer sizing, observing that the oversized buffers in cable modems cause massive delays and standing queues during congestion, often referred to as *bufferbloat*. This observation led to, among other contributions, the development of new AQM algorithms [14, 15] and new congestion control algorithms [5]. We observe similar large delays and standing queues in internet routers in Section 4.1.

Hohlfeld et al. in [11] run a testbed study on the effect of buffer sizing on the QoE of VOIP, RTP video streaming, and web traffic. Among other findings, they observe that buffer sizing can cause a significant change in QoS metrics (e.g. packet loss, RTT) which result in a much smaller change in QoE metrics. We observe the same, in some cases doubling the rate of packet loss resulting in a much smaller change in QoE metrics.

3 EXPERIMENTAL METHODOLOGY

In this section we describe the Netflix architecture, our experiment, and the metrics we use to evaluate the results.

3.1 Netflix Open Connect Architecture

Netflix's CDN is called *Open Connect*. It consists of many points of presence around the world, called "sites". Figure 1 shows an example site. For fault tolerance, sites are split into two "stacks." Each stack consists of a router, a set of *catalog* servers which store the entire Netflix catalog, and a set of faster *offload* servers which store a smaller set of the most popular videos, so named since they "offload" the popular videos from the catalog servers. The traffic is video traffic. It is primarily long-lived TCP New Reno flows, though it also contains a non-negligible number of short-lived flows. Please refer to [1] for more information.

3.2 Router Architecture

Our experiments were done using Combined Input-Output Queued (CIOQ) internet routers. These routers employ a different buffer architecture than the output queued (or shared buffer) routers of existing work. In an output queued router, all input ports place packets into the same buffer, and this buffer is drained by the output port. In a CIOQ router, input ports place packets into different Virtual Output Queues (VOQs), which are drained according to a scheduling algorithm.

The crucial difference between these two models is the rate at which these buffers are drained: in existing work,

Experiment parameters				Application effects			TCP effects	
Site	A Buffer	B Buffer	Hours	Sess. w/ Rebuffer	Low Qual. Sec.	Play Delay	Min. RTT	Retrans.
#1	50MB	500MB	46	+46.3%	+5.7%	-5.9%	-10.9%	+85.2%
#1	250MB	500MB	30	+5.5%	-1.6%	-3.6%	+9.6%	+17.1%
#1	750MB	500MB	13	+10.6%	+7.2%	+7.8%	-19.2%	-1.6%
#1	1000MB	500MB	15	+4.9%	+9.6%	+9.5%	-16.4%	-10.4%
#2	5MB	50MB	14	+0.5%	-2.1%	-5.7%	-13.5%	+51.1%
#2	12MB	50MB	22	+33.9%	-4.0%	-6.0%	-19.1%	+68.7%
#2	25MB	500MB	90	-15.6%	-5.3%	-13.5%	-34.8%	+130.6%
#3	50MB	500MB	34	-22.1%	-7.0%	-14.8%	-5.1%	+134.8%

Table 1: Average percent change in application performance during congested hours. A positive value corresponds to an increase in that metric for the canary. Lower values correspond to an improvement in the metric for the "A Buffer" size. Statistically significant results (p=0.01) are highlighted in gray. For more information, see Section 4

.

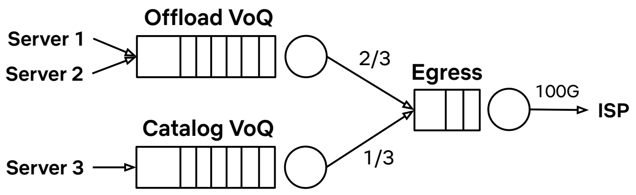

Figure 2: Diagram of router buffer architecture. Packets are queued in each VoQ upon arrival, and are drained at a rate chosen by the egress scheduling algorithm.

buffers are drained at a constant rate which corresponds to the port speed of the output port. In our routers, the rate at which a VOQ is drained is chosen by an internal scheduling algorithm, and can vary over time.

Figure 2 shows an example internal architecture of the routers used in our experiments. The routers are divided into line cards. The buffers for each line card are logically divided into VoQs. Each output port has a few dedicated VoQs on each line card. Each input port is assigned to one of these VoQs, and usually each VoQ has three or four associated servers.

When a packet arrives from a server, it is placed into the corresponding VoQ. The VoQ sends packets to the egress port when allowed to by a scheduler. The egress port has a small 100KB buffer, intended to briefly store packets before transmission and to aid with packet reconstruction.

The size of the VoQ is configurable, and we configure the router so that arriving packets are dropped when the VoQ is full. When we set a buffer size in our experiments, for instance 500MB, we are setting all VoQs to have a limit of 500MB.

We used the default scheduling algorithm for the router in our experiments, which is Deficit Weighted Round Robin (DWRR), with weights proportional to the aggregate capacity of the input ports. For instance in Figure 2, if all servers are connected with 100G links, VoQ 1 will have a weight of 300 and VoQ 2 a weight of 200. We discuss the implications of this scheduling algorithm in Section 5.

3.3 Our Experiment

We first identified a number of sites with persistent congestion to a peer during peak hours. In each site, we ensured that the ISP's traffic was assigned to the two stacks independently and uniformly at random. We ran experiments with the same buffer sizes on both stacks, and observed minimal differences in the amount of traffic and quality metrics. This gave us a controlled A/B test which allowed us to measure the effect of buffer sizing on quality metrics.

We configured the pair of stacks to use a variety of different buffer settings. This resulted in setting the buffer size for each Virtual Output Queue (VoQ) in the router to the corresponding setting. We will discuss more about the implications of this in Section 5.

We observed the difference in performance on both application-level and TCP-level metrics. All traffic in our experiments used TCP New Reno.

3.4 Confounding issues

There are a few confounding issues which limit the generalizability of our results. The major one is the Router's VoQ architecture and scheduling algorithms. The scheduling algorithm means that each queue is served at a variable rate. It is possible that this biases the effect on metrics in one way or another, for instance if an offload VoQ is served at a low rate and a catalog queue is served at a high rate, this could result in some traffic experiencing worse congestion than it otherwise would.

Our experiment is not a perfect controlled trial. Netflix clients can switch between the different stacks. If a video chunk fails to be downloaded from one stack, for example, the client might switch to the other stack or another site entirely. This can also bias results. For instance, we could see an increase in rebuffers because a certain buffer size caused worse QoE, or because the *other* buffer size resulted in very bad QoE and the first stack became more heavily loaded.

3.5 Metric Definitions

A *congested hour* is an hour where the per-second link utilization, averaged over one hour, exceeds 98%. In our experiments, this is the point at which we begin to see an increase in RTT due to congestion.[1]

A *session* refers to one TCP connection between a client and a Netflix server.

The *minimum RTT* for a session is the minimum time between when a packet is sent and its acknowledgment arrives for all packets in a session. Note that it is the absolute minimum, not the minimum of TCP's smoothed RTT.

We measure loss via the *percentage of retransmitted bytes*, which is the fraction of all TCP bytes sent during a time period which are retransmitted.

A *rebuffer* is when video playback halts because data is not available to the client. We look at the percentage of sessions with at least one rebuffer.

We measure video quality using VMAF [4], which models how people perceive the subjective quality of video. The *low quality seconds* metric measures the time-weighted fraction of frames with a VMAF below 80.

In a few cases, we've normalized sensitive metrics. This was done by dividing all metrics in a graph by the largest value in that graph.

4 EXPERIMENT RESULTS

In this section, we discuss the results of our experiments, both on TCP-level metrics and on video QoE metrics.

Table 1 summarizes the results of our experiments, and their effect on video performance. In each row, we report the average percent difference in various performance metrics during congested hours (where link utilization was at least 98%) between the traffic to the A and B stacks. A negative value corresponds to a lower number for the A Stack. For all presented metrics, lower is better. We compute bootstrap confidence intervals [8] for $p = 0.01$, and highlight cells where the confidence intervals does not include zero. As a

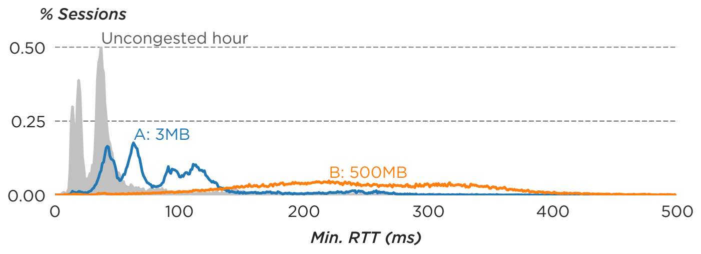

Figure 3: Distribution of minimum observed RTT by a TCP session during a congested hour in Site #2

sense of scale, all statistically significant results we observe are quite large compared to other experiments at Netflix.

4.1 Impact on TCP New Reno

For TCP New Reno, our results match with intuition: as buffers get smaller, we observe an increase in packet loss and decrease in RTT.

Figure 3 shows an example impact of very large buffers on RTT in Site #2. When buffers get very large, there are many negative effects—we observe high RTTs and large variation in RTT. Our preliminary results suggest this could be due to an increase in the percentage of sessions which are receive-window and not congestion-window limited.

We observe that buffers add a consistent delay to TCP flows. It is commonly assumed (*e.g.* [11]) that if a flow sends enough packets during congestion, eventually the flow will observe a RTT which corresponds to the propagation delay through the network with no queueing. Our results show that this is *not* the case. For instance, Figure 4 shows the minimum RTT distribution in Site #1, with two moderately sized buffers (50MB and 500MB). If this assumption were true, we would expect the RTTs during the uncongested and congested hours to be similar, but they are not.

One surprising thing about Figures 3 and 4 is how much larger the RTT is for the 500MB buffer in Site #2 than in Site #1. We will explain this in Section 5.

We observe that loss increases as the size of the buffer decreases, and as the number of flows increases. Figure 5 shows that the percentage of retransmitted bytes for an hour increases as load increases, and increases more quickly for smaller buffers. Since hours with higher load tend to have more flows, this aligns with the predictions of [7, 13, 16].

We found this behavior a bit surprising. One might hope that due to the large number of flows at Netflix, the queue would approximately behave like an $M/M/1$ queue of size B. In particular, loss probability would not strongly depend on B once B is more than a hundred or so packets. This is not supported by our results, however.

[1]The fact that we do not observe an affect on RTT until exceeding 98% utilization is a bit surprising. Previous work defines congestion with a much lower threshold, for instance [3], defines a congested hour as one which exceeds 75% link utilization.

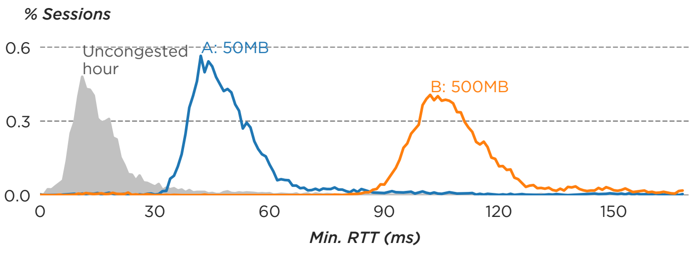

Figure 4: Distribution of minimum observed RTT by a TCP session during a congested hour in Site #1

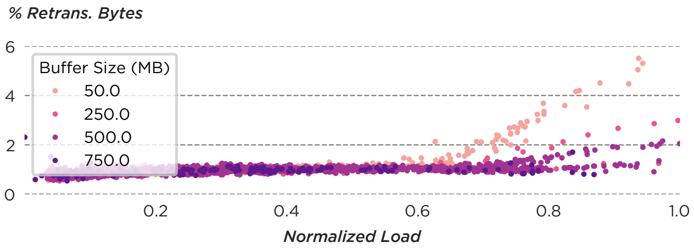

Figure 5: Percentage of retransmitted bytes as a function of normalized load in Site #1. Each point represents an hour. Retransmits tend to increase as load increases, and increase faster for smaller buffers.

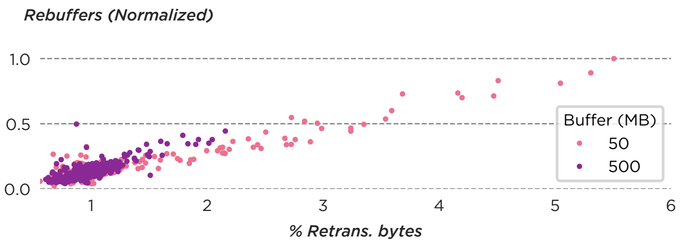

Figure 6: Rebuffers as a function of the percentage of retransmitted bytes. For hours with similar percentages of retransmits, the smaller buffer router has lower rates of rebuffers.

4.2 Impact on Video

Buffer sizing has a big impact on video streaming quality. We generally find that there is a buffer sizing sweet spot for video streaming, and that a buffer which is too small or too large can negatively impact quality.

We consistently observe that reducing the size of a buffer reduces the video play delay.

Once the buffer size becomes too large (e.g. Site #2 and #3 with a 500MB buffer), we see an increase in the number of rebuffers, increases in the amount of low quality video streamed, and an increase in the time it takes to start a video.

If the buffer is too small (e.g. Site #1 , we see the opposite effect: an increase in rebuffers, increase in low quality video. The 50MB buffer for TCP New Reno in Site #1 is an example of a too-small buffer.

We note that the increase in rebuffers we observe due to the smaller buffer is not *solely* due to the increase in packet loss. While rebuffers are correlated with loss, Figure 6 shows that larger buffers tend to have a higher rate of rebuffers for similar loss rates. Furthermore, we see cases where a buffer size which causes an increase in retransmits can correspond with an increase in overall QoE, for instance Site #2 with a 25MB buffer or Site #3 with a 50MB buffer.

5 ROUTER ARCHITECTURE IMPACTS CHOICE OF BUFFER SIZE

In Section 3.2, we described the architecture of the router buffers in our experiment. In this section, we describe some of the effects of this architecture in conjunction with the particular scheduling algorithm we used.

Recall that the VOQ scheduler used a Deficit Weighted Round Robin (DWRR) policy. DWRR associates a weight with each VOQ, and ensures that the *departure rates* of the VOQs are proportional to their weights. However, if one or more VOQs have a lower arrival rate than their weighted fair share, then that VOQ will be served at its arrival rate and the remaining bandwidth will be allocated among the remaining VOQs. For instance, if the "Catalog VOQ" in Figure 2 has an arrival rate lower than 33 Gbps, for instance 20 Gbps, then all 20 Gbps of its traffic will be sent, and the remaining 80 Gbps will be given to traffic from the "Offload VOQ". If this happens, the "Catalog VOQ" will experience no queueing delay.

On its own, this is standard behavior for a VOQ-based system. However, in our setting, there were a number of additional factors:

(1) The VOQs were configured to be shared by many servers.
(2) The scheduling algorithm set the weight of each VOQ based on the total available capacity plugged into the input ports for that VOQ. For instance, if three 100G servers were plugged into a VOQ, that VOQ would essentially have a weight of three.
(3) Due to the cabling, most VOQs consisted of entirely offload servers or entirely catalog servers.

The combination of these factors caused lots of surprising behavior throughout our experiments. Including:

Large queueing delay. Figures 4 and 3 both show the observed Minimum RTT distributions in two different sites, during an experiment where one buffer was set to 500MB. Both are for congested 100G ports, so we expected to see an increase in Min. RTT of about 40ms. Instead, we observe an 80ms increase in Figure 4 and a *100-300ms* increase in Figure 3. This difference is due to the number of cores: Site

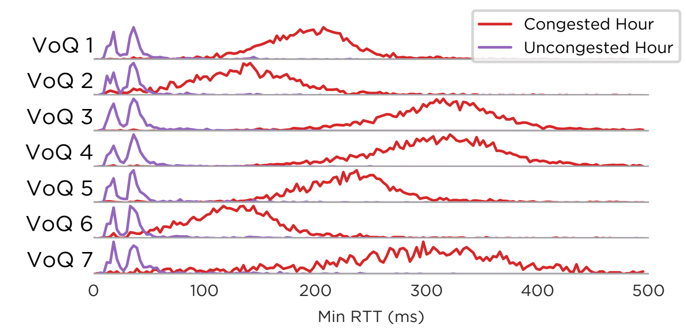

Figure 7: Distribution of the minimum RTTs observed by TCP flows sharing a router, one row per VOQ. Each VOQ has the same size.

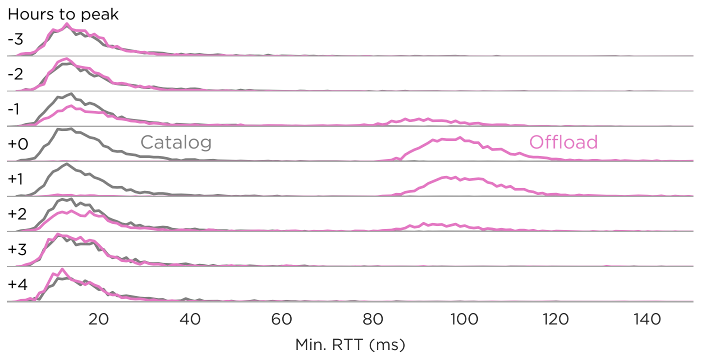

Figure 8: Normalized distributions of the minimum RTT observed by TCP flows from one stack, split by Catalog and Offload servers with one row per hour.

#1 has two cores, and Site #2 eighteen, so the VOQ in Site #2 is being served at a fraction of the rate of Site #1.

Unfairness between cores. Figure 7 shows the distribution of RTTs for seven of Site #2's VOQs during an uncongested and a congested hour. Despite having the same size and a similar RTT distribution during the uncongested hour, the RTT distributions are very different during the congested hour. This is because each VOQ is being served at a different rate by the scheduling algorithm.

Uncongested traffic during congestion. We expected that all traffic sharing a congested port would experience congestion. However, if some VOQ did not have enough traffic to meet its weighted fair share, it would experience no congestion. Figure 8 shows the RTT distribution for the hours leading up to and after peak in Site #1, for traffic using a congested port. Because the catalog servers did not send their weighted fair share of traffic, they experienced no congestion.

We have been working with our router vendor to ameliorate some of these issues, and we believe that it will be possible to fix much of this behavior via a firmware upgrade. However it is not clear to us what the right scheduling behavior is nor how to size buffers given whatever scheduling behavior we observe, and we leave this question for future work.

6 CONCLUSION

We find that TCP Reno generally behaves as expected when changing buffer sizes, with a smaller buffer causing higher loss and lower delay.

The story for the best buffer size for video is much more complicated, but we find that improvements in buffer size can dramatically improve video QoE.

We also find that considering application QoE is crucial when sizing router buffers. Existing buffer sizing work has generally focused on network-level metrics, for instance whether a link is fully utilized [2, 3, 9, 17], loss, queueing delay [7]. In all our experiments there were interesting effects on video QoE which went beyond whether the link was fully utilized, and whether loss or delay increased.

An intuitive idea is that if a buffer never goes empty during periods of congestion, then it could be shrunk to improve (or at least not degrade) application performance. Our results suggest that this intuition is not true. In our experiments with smaller buffers, we observe negative effects on applications (e.g. an increase in rebuffers) while still seeing an elevated distribution of minimum RTTs.

We plan on continuing this line of work. We are exploring ways of understanding and experimenting with buffer sizing in the presence of the VoQ scheduling algorithms, getting a better understanding of the mechanisms by which buffer sizing affects video QoE, and understanding how different congestion control algorithms affect buffer sizing.

ACKNOWLEDGEMENTS

Thanks to many, many people at Netflix for supporting this effort. Especially Grenville Armitage, Lawrence Stewart, and Randall Stewart for invaluable help with our many TCP-related questions, Andy Rhines and Matthew Pawlicki who helped with data analysis and infrastructure, and the Streaming Innovation team for being great internship hosts.

REFERENCES

[1] [n. d.]. Netflix Open Connect. https://openconnect.netflix.com/.

[2] Guido Appenzeller, Nick McKeown, and Isaac Keslassy. 2004. Sizing Router Buffers. In *ACM SIGCOMM Computer Communication Review*. ACM, 281–292. https://doi.org/10.1145/1030194.1015499

[3] Neda Beheshti, Yashar Ganjali, Monia Ghobadi, Nick McKeown, and Geoff Salmon. 2008. Experimental Study of Router Buffer Sizing. In *Proceedings of the 8th ACM SIGCOMM Conference on Internet Measurement (IMC '08)*. ACM, New York, NY, USA, 197–210. https://doi.org/10.1145/1452520.1452545

[4] Netflix Technology Blog. 2017. Toward A Practical Perceptual Video Quality Metric. https://medium.com/netflix-techblog/toward-a-practical-perceptual-video-quality-metric-653f208b9652.

[5] Neal Cardwell, Yuchung Cheng, C. Stephen Gunn, Soheil Hassas Yeganeh, and Van Jacobson. 2017. BBR: Congestion-Based Congestion Control. *Commun. ACM* 60, 2 (Jan. 2017), 58–66. https://doi.org/10.1145/3009824

[6] Shang-Tse Chuang, Ashish Goel, Nick McKeown, and Balaji Prabhakar. 2006. Matching Output Queueing with a Combined Input Output Queued Switch. *IEEE Journal on Selected Areas in Communications* 17, 6 (Sept. 2006), 1030–1039.

[7] A. Dhamdhere, Hao Jiang, and C. Dovrolis. 2005. Buffer Sizing for Congested Internet Links. In *Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies.*, Vol. 2. IEEE, Miami, FL, USA, 1072–1083. https://doi.org/10.1109/INFCOM.2005.1498335

[8] B. Efron. 1979. Bootstrap Methods: Another Look at the Jackknife. *The Annals of Statistics* 7, 1 (Jan. 1979), 1–26. https://doi.org/10.1214/aos/1176344552

[9] Mihaela Enachescu, Yashar Ganjali, Ashish Goel, Nick McKeown, and Tim Roughgarden. 2006. Routers with Very Small Buffers. *INFOCOM* (2006), 1–11. https://doi.org/10.1109/INFOCOM.2006.240

[10] Jim Gettys, Bell Labs, and Kathleen Nichols. 2011. Bufferbloat: Dark Buffers in the Internet. (Nov. 2011), 15.

[11] Oliver Hohlfeld, Enric Pujol, Florin Ciucu, Anja Feldmann, and Paul Barford. 2014. A QoE Perspective on Sizing Network Buffers. In *Proceedings of the 2014 Conference on Internet Measurement Conference - IMC '14*. ACM Press, Vancouver, BC, Canada, 333–346. https://doi.org/10.1145/2663716.2663730

[12] Van Jacobson. 1990. Modified TCP Congestion Avoidance Algorithm.

[13] R. Morris. 1997. TCP Behavior with Many Flows. In *Proceedings 1997 International Conference on Network Protocols*. 205–211. https://doi.org/10.1109/ICNP.1997.643715

[14] Kathleen Nichols and Van Jacobson. 2012. Controlling Queue Delay. *Queue* 10, 5 (May 2012), 20:20–20:34. https://doi.org/10.1145/2208917.2209336

[15] R. Pan, P. Natarajan, C. Piglione, M. S. Prabhu, V. Subramanian, F. Baker, and B. VerSteeg. 2013. PIE: A Lightweight Control Scheme to Address the Bufferbloat Problem. In *2013 IEEE 14th International Conference on High Performance Switching and Routing (HPSR)*. 148–155. https://doi.org/10.1109/HPSR.2013.6602305

[16] Lili Qiu, Yin Zhang, and Srinivasan Keshav. 2001. Understanding the Performance of Many TCP Flows. *Computer Networks* 37, 3-4 (Nov. 2001), 277–306. https://doi.org/10.1016/S1389-1286(01)00203-1

[17] Rade Stanojevic, Robert N. Shorten, and Christopher M. Kellett. 2007. Adaptive Tuning of Drop-Tail Buffers for Reducing Queueing Delays. *ACM SIGCOMM Computer Communication Review* 37, 1 (Jan. 2007).

[18] Curtis Villamizar and Cheng Song. 1994. High Performance TCP in ANSNET. *SIGCOMM Comput. Commun. Rev.* 24, 5 (Oct. 1994), 45–60. https://doi.org/10.1145/205511.205520