HAS Based Empirical QoE Study over TCP and QUIC on Diverse Networks

Md Tariqul Islam[1], **Christian Esteve Rothenberg**[1]

[1]Universidade Estadual de Campinas (UNICAMP)
Faculdade de Engenharia Elétrica e de Computação (FEEC)
Av Albert Einstein, 400, Cidade Universitaria Zeferino Vaz, Campinas, SP, Brasil.

{tariqsaj,chesteve@dca.fee.unicamp.br}

Abstract. *HTTP adaptive streaming (HAS) is the de-facto standard for video services over the Internet delivering increased Quality of Experience (QoE) as a function of the network status. Such adaptive streaming atop HTTP relies predominantly on TCP as the reliable transport protocol. Recently, QUIC, an alternative of TCP transport, has emerged to overcome TCP's native shortcomings and improve the HTTP-based applications QoE. This paper investigates three strategies (Rate, Buffer, and Hybrid) based adaptive bitrate streaming (ABS) algorithms behavioral performance over the traditional TCP and QUIC transport protocol. For this purpose, we experimentally evaluate different cellular network traces in a high-fidelity emulated testbed and compare the performance of ABS algorithms considering QoE metrics over TCP and QUIC. Our empirical results show that each ABS algorithm's (Conventional, BBA, and Arbiter) QoE performance is biased for TCP. As a result, QUIC suffers the ineffectiveness of traditional state-of-art ABS algorithms to improve video streaming performance without specific changes.*

1. Introduction

Multimedia traffic, precisely video streaming traffic, becomes the most dominant application on today's Internet. Besides, a large number of video streaming is provided via over-the-top (OTT) platforms. Hence, the popularity of OTT platforms such as Netflix, Hulu, and Amazon are growing enormously for video-on-demand (VoD) services. A recent study showed that video streaming traffic's rapid growth would control 80% of the total Internet traffic in 2021 [Cisco 2018]. Aside from that, the recent COVID-19 pandemic increases video content viewing time by 60% in 2020 [Conviva 2020].

HTTP adaptive streaming (HAS) is considered the de-facto standard to deliver VoD services [Thang et al. 2014], and Dynamic Adaptive Streaming over HTTP (DASH) is the most dominating format for implementing HAS [Kua et al. 2017]. In this form, the Adaptive Bitrate Streaming (ABS) algorithm is deployed in a client-side player that detects a favorable quality of the video stream between multiple bitrates and resolutions. Usually, the ABS algorithm relies on the network status (e.g., throughput) and client-side buffer level conditions for dynamically selecting appropriate video streaming bitrates without causing stalls/rebuffering events in the playback. Such metrics (e.g., bitrate and stall) have a strong influence on end-user Quality of Experience (QoE) [Dobrian et al. 2011]. Moreover, the ABS algorithms were initially designed and built over TCP transport as the HTTP standard requires a reliable transport [Fielding and Reschke 2014]. As a result, HAS has been using the TCP transport protocol predominantly for many years. Despite having the reliability and in-order delivery benefits of TCP, both HTTP/1.1 and HTTP/2 versions of application-level HTTP standard over traditional TCP transport suffers from Head-of-Line (HOL) blocking problem [Marx et al. 2019]. In contrast, for the third version of the HTTP standard (HTTP/3) [Bishop et al. 2020], the new transport protocol QUIC [Langley et al. 2017] running over the UDP solves the issues of HOL blocking.

This paper aims to analyze the QoE performance of typical ABS algorithms built on mainly the TCP transport concept. However, the recent network technology is evolving towards the fifth-generation (5G) network to meet the high traffic demand as the legacy of the third (3G) and fourth (4G) generation networks. In this paper, we try to get a few answers to the following questions:

- How does the state-of-art ABS algorithm perform over QUIC and traditional TCP across diverse network conditions for high-quality video?
- Does QUIC outperform TCP when parallel clients are competing for the same video content?
- Does QUIC outperform TCP on a restless network?

To interpret the questions mentioned earlier, we evaluated a head-to-head QoE performance comparison of ABS algorithms over QUIC and traditional TCP transport protocol in a controlled testbed environment. We leveraged the similar approaches conducted in earlier work [Bhat et al. 2017] [Timmerer and Bertoni 2016] with our testbed incorporating the 5G network in addition to the 3G and 4G network, and DASH content from Ultra High Definition (UHD) 4K dataset. For this purpose, we undergo the following experiments:

1. QoE performance of ABS algorithms over QUIC and TCP across selected 3G, 4G, and 5G network traces for the single and parallel client competing DASH content.
2. QoE performance of ABS algorithms over QUIC and TCP on the restless network with extreme bandwidth fluctuation.

The rest of paper is organized as follows. Section 2 presents background in the areas of HAS and transport protocols. Section 3 exhibits related works. Section 4 presents experimental setup supporting TCP and QUIC transport. Experimental results are discussed in Section 5. Finally, Section 6 concludes the paper with future work directions.

2. Background

2.1. HTTP Adaptive Streaming (HAS)

HTTP Adaptive Streaming (HAS) works by breaking the video content into a sequence of small HTTP-based file segments [Thang et al. 2014]. Each segment contains a short interval (between 2 and 10 seconds in duration) playback time of a video with different representation level information (encoded with different bitrates and resolutions). An index file contains this information. Each different HAS implementation strategy has given different names of that index file. For Dynamic Adaptive Streaming over HTTP (DASH) implementation, the index file is called Media Presentation Description (MPD) [Kua et al. 2017]. DASH standard only specifies the structure of different representations of video content in MPD format. In a HAS streaming session, the DASH format compatible client will first download the MPD from a server that contains the representation level information for video content. Based on that information, it will request the individual segments that include appropriate video quality. Client-side ABS algorithms are mainly responsible for dynamically selecting the appropriate segments based on current network conditions (e.g., throughput) and client playback buffer level. The purpose of this dynamic segment selection to adapt to changes in network conditions and avoid the unwanted stall/rebuffering event. The ABS algorithms have three strategies to choose the appropriate video segments as follows: *1. Rate based* - This strategy makes an estimation of the bandwidth based on delivery rates of previously downloaded segments and adopts the best representation (quality) of the next segment, *2. Buffer based* - This strategy monitors the state of playback buffer before every segment downloads and makes appropriate decisions for the next segment, *3. Hybrid* - This strategy is a combination of rate and buffer based algorithms and adopts the best representation (quality) of the next segment from those two results.

2.2. Transport Options: TCP and QUIC

HAS was initially designed and implemented on top of TCP as the application-level HTTP standard requires a reliable transport protocol [Fielding and Reschke 2014]. Afterward, HAS predominately used TCP long time for the benefits of reliability and in-order delivery. But, the initial HTTP/1.1 standard with persistent connection feature has suffered from a well-known HOL blocking problem. Such a situation occurs as each client has limited TCP connections to the server and a delay in a new request queue over those connections. Although a pipeline feature was added later to make multiple requests over a single connection, the HOL problem has not been resolved because it requires responses to arrive in order. The next version, HTTP/2 [Belshe et al. 2015] standardized by the Internet Engineering Task Force (IETF), has come with a multiplexing feature to overcome this issue. In HTTP/2 multiplexing, a single TCP connection can handle multiple requests in parallel, and responses don't require to arrive in order. It also embraces a) server push, b) stream priority, and c) stream termination features. Nevertheless, another kind of HOL blocking still exists in TCP transport for HTTP/2 standard. When HTTP/2 uses TCP, if a packet loss occurs in the TCP stream, it makes all subsequent TCP streams wait until that packet is re-transmitted and recovered. In HAS, such retransmission may cause unwanted delays while downloading video segments and force the ABS algorithm to adopt degraded quality segments.

Concerning data transfer, privacy issues in recent OTT platforms are delivering their streaming services with encryption. A new Transport Layer Security (TLS) is imposed over the TCP and under the HTTP to facilitate the privacy and data security for communications over the Internet through encryption. In this paper, we adopt the TCP term considering the combination of HTTP+TLS+TCP, which is also referred to as HTTPS (HyperText Transfer Protocol Secure). TLS requires a new handshake to ensure that the session is secured alongside the initial TCP handshake, leading to a little time-consuming connection establishment.

As mentioned above, the shortcomings of TCP lead to the development of alternative transport protocol such as Google developed QUIC protocol running on top of UDP [Langley et al. 2017]. QUIC inherits all the HTTP features over TCP, aiming to reduce connection establishment, improving congestion control, multiplexed/pipelined requests without HOL blocking, forward error correction, and seamless connection migration. Recently, QUIC was adopted as a transport protocol for the HTTP/3 standard [Bishop et al. 2020]. Therefore, HTTP/3 standard over QUIC uses a single handshake for a secure connection and avoids the HOL blocking issue from the multiplexing feature. In HTTP/3 over QUIC multiplexing, each stream is independent of each other, and subsequent streams are not affected while a particular stream packet loss occurs.

3. Related Work

Table 1 presents relevant details about the work related to the performance of HAS over TCP and QUIC transport.

Table 1. Related work in the area of HAS performance evaluation over TCP and QUIC (Category 1 and 2) and current research approaches for getting better HAS performance over newly standardized QUIC transport (Category 3).

Category	Main Insight	Work
1	**(1) QUIC offers poor performance than TCP for adaptive video streaming.** **(2) An open-source implementation is applied for QUIC transport.**	[Timmerer and Bertoni 2016] evaluated TCP and QUIC on dynamic adaptive streaming with varying network link utilization and throughput. The authors stated that QUIC does not provide improvement in the overall streaming performance. [Bhat et al. 2017] evaluated the performance of the ABS algorithms by head-to-head comparing between TCP and QUIC transport. The authors stated that QUIC does not provide significant QoE benefits to the existing ABS algorithms because these algorithms were designed over TCP.
2	**(1) QUIC offers better performance than TCP for adaptive video streaming with Google provided QUIC server and player.** **(2) In category 1, QUIC was less competitive than TCP due to using open source implementation of QUIC or not using Google provided server with the latest version of QUIC transport.**	[Arisu and Begen 2018] evaluated QUIC in terms of the users' frame-seek requests and frequent network changes. They used Google provided toy server and client (player), and showed QUIC provides better QoE by reducing the wait times and the buffer starvation rates. [Kakhki et al. 2017] found that QUIC provided better streaming QoE, but only for high-quality video-streaming using YouTube. [Zinner et al. 2017] showed QUIC with 0-RTT connection establishment performed better than the other protocols for the playback start in YouTube video streaming.
3	**(1) Novel strategies to make the QoE performance of adaptive video streaming more robust with QUIC transport protocols.** **(2) Modification of ABS algorithms to obtain better performance over QUIC.**	[Li et al. 2016] and [Hayes et al. 2017] provided MMT and SDN based approach to improve the QoE of adaptive streaming over QUIC transport. [Bhat et al. 2018] showed QUIC performed better than TCP with a modified DASH-based ABS approach (SQUAD), which inherits retransmit segments' ability to improve to QoE of a viewer watching the video. [Nguyen et al. 2020] presented a retransmission technique (H2BR) for the modified ABS algorithms perform well by improving the average video quality with HTTP/3 atop QUIC compared to HTTP/2 atop TCP in the context of packet loss and retransmission.

In this work, we evaluate QoE performance on diverse network conditions (similar to category 1) and state some future direction to obtain better QoE performance over QUIC transport.

4. Experimental setup

Testbed. Our testbed framework is based on the following components: (*i*) Mininet[1]- an emulated network environment, (*ii*) goDASH [Raca et al. 2020b] - an open-source DASH video player, (*iii*) Caddy[2] - a web server hosting DASH video content (*iv*) DITG[3] - a background traffic (cross traffic) generator, and (*v*) Linux traffic control (TC)[4] - a traffic controller in the Linux kernel. The topology contains two Open vSwitch (Switch 1 and 2). Figure 1 depicts DASH clients and DITG cross-traffic sender connected with Switch 1, and on the opposite side, a single web server hosting DASH video content and DITG cross-traffic receiver connected with Switch 2.

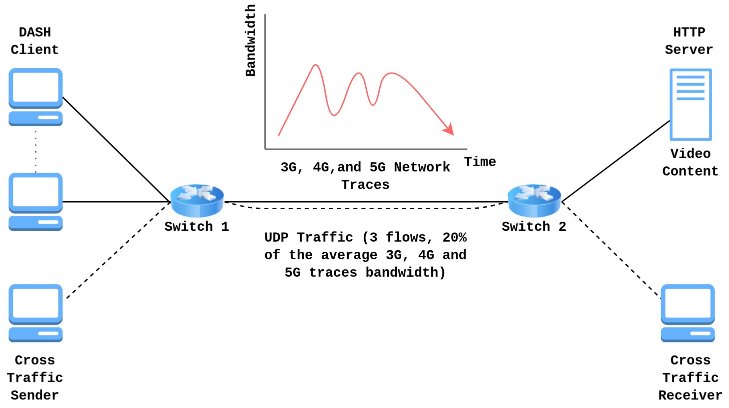

Figure 1. Testbed topology

DASH Player. To stream the DASH video, we use goDASH, a lightweight headless streaming player at the client-side. It has the feature of supporting eight different state-of-art ABS algorithms, two transport protocols TCP and QUIC. Also, goDASH provides a video streaming log containing per segment objective QoE metrics (segment arrival time, stall, bitrate, resolution, etc.) and five different real-time output from QoE models (ITU-T P.1203 MOS, Claey, Dunamu, Yin, and Yu). Since HTTPS require a secure and encrypted connection, goDASH is equipped with its goDASHbed[5] to set security certificates for both TCP and QUIC protocol. Our testbed derives all the features of goDASHbed. We have customized the topology scenario, network traces, DITG traffic utilization, and DASH content and algorithms choice. The testbed offers a Caddy (v2) webserver hosting DASH video content. Caddy is a Web Server Gateway Interface (WSGI) server with support HTTP/3 atop experimental QUIC and HTTP/1.1 and HTTP/2 atop TCP. Hence the DASH clients can stream videos either over TCP or QUIC. Caddy leverage the quic-go[6] library, an open-source implementation of QUIC transport protocol (draft-29) [Iyengar and Thomson] written in Go language.

Video Source. We use a 4-second segment duration short science fiction film (Tears of Steel), sourced from a publicly available 4K DASH video

[1] http://mininet.org/
[2] https://caddyserver.com/
[3] http://www.grid.unina.it/software/ITG/
[4] https://linux.die.net/man/8/tc
[5] https://github.com/uccmisl/goDASHbed
[6] https://github.com/lucas-clemente/quic-go

Table 2. Quality Representations: (Bit-rates vs Resolutions) *Tears of Steel* Video

Bit-rates (Kbps)	230	375	560	750	1050	1750	2350	3000	3850	4300	15000	25000	40000
Resolutions	320x 180	384x 216	512x 288	512x 288	640x 360	736x 414	1280x 720	1280x 720	1920x 1080	1920x 1080	1920x 1080	3840x 2160	3840x 2160

dataset [Quinlan and Sreenan 2018]. This video content has a total duration of over 14 minutes, encoded using H.264/AVC, and contains eight resolutions across thirteen representation rates. The detailed mapping of the video resolution to bitrate is shown in Table 2.

ABS Algorithms. For the sake of our QoE evaluation, we work with the following three algorithms: *1. Conventional (Rate-based) [Li et al. 2014]*, *2. BBA (Buffer based) [Huang et al. 2014]*, and *Arbiter (Hybrid) [Zahran et al. 2018].*

Trace-based Network Performance. We emulated different network conditions through Linux TC (Hierarchical Token Bucket) between link Switch 1 and 2 using the downlink bandwidth parameter from 3G [Riiser et al. 2013], 4G [Raca et al. 2018], and 5G [Raca et al. 2020a] cellular network traces. For our experiment (1), we randomly selected a total of 15 different mobility traces where 3G (Bandwidth: mean=1.74 and std=0.92 Mbps), 4G (Bandwidth: mean=12.50 and std=14.64 Mbps), and 5G (Bandwidth: mean=44.41 and std=53.40 Mbps) traces represent low, moderate, and higher bandwidth scenarios respectively. Figure 2 illustrates the statistical overview of the 15 different traces. Apart from this, for the experiment (2), we have selected another driving mobility trace of the 5G network to emulate the behavior of a unstable network where network mode frequently changes from 5G to LTE or HSPA+, as presents in Figure 3.

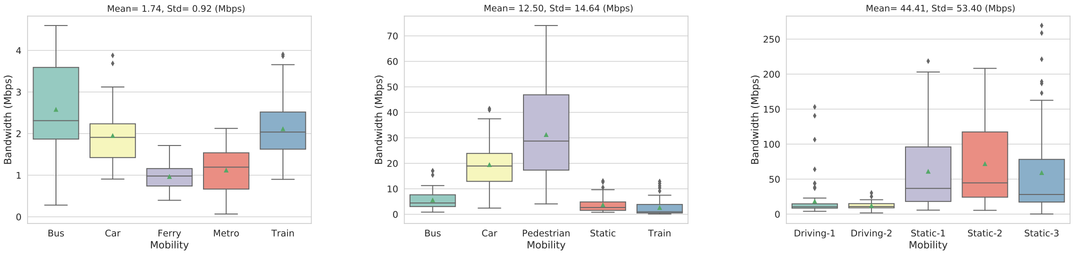

Figure 2. Fifteen different mobility traces over 3G (left), 4G (center), and 5G (right) networks statistical overview through box-plot

Background Traffic. The DITG tool was used to introduce the cross-traffic over testbed video streaming traffic. Using DITG, we sent three concurrent flows of UDP traffic from one sender host to another receiver host. The amount of total UDP cross-traffic in such quantity occupies approximately 20% of the average 3G, 4G, and 5G traces bandwidth.

QoE Metrics. We have considered the following five metrics that play an essential role in defining video streaming performance and end user's QoE:

1. **Startup Delay**- The delay measured in the startup phase of video streaming is known as startup delay. We have measured the first two segments' arrival time in the player buffer for the startup delay metrics as goDASH uses two segments for the initial buffer threshold.

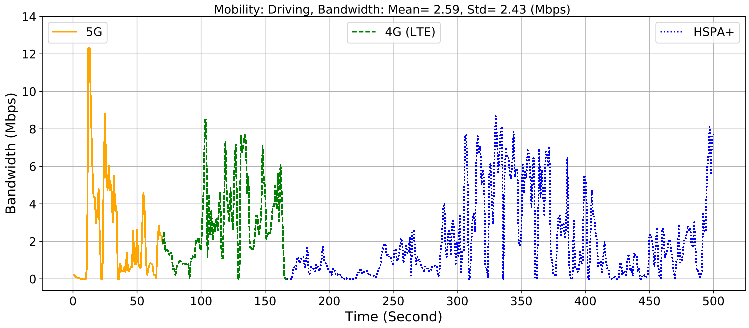

Figure 3. Single selected driving mobility traces over 5G network contains extreme bandwidth fluctuation and frequent connection changes

2. **Stall Ratio**- Insufficient number of video segments in player buffer creates stall events. The higher number of stalling events increased the user's abandonment rate when playing the video. We defined the Stall Event Ratio[7] 0 as No-Stall, 0 to 0.1 as Mild-Stall, and above 0.1 as Severe-Stall.
3. **Quality Switches**- This metric defines the number of quality levels switched over video streaming. In general, poor network condition leads to the high number of changes in quality level.
4. **Average Bit-rate**- The average bit-rate over a video session is a crucial metric for displaying video streaming quality. The ABS algorithms aim to maximize the average quality bit-rate so that end-user perceived better quality of the video.
5. **P1203 MOS**- The mean opinion score (MOS) values range between 1 and 5, calculated according to ITU-T Rec. P.1203.1 (mode 0: bit-rate, framerate, and resolution) standardization[8]. The main shortcoming of the ITU-T standardization is its limitation to H.264 (AVC) encoding content to Full HD (1920x1080). Hence in our experiment, each video streams conduct by setting the maximum 1920x1080 resolution in a goDASH player end.

As the current goDASH version has not the feature of using HTTP/2 standard, thus for the following stated each experiment, goDASH clients stream the video using the above mentioned three ABS algorithms from the single caddy server either by HTTP/1.1 atop TCP or HTTP/3 atop QUIC transport.

5. Results

This section shows the result obtained from two selected experiments using TCP or QUIC transport with a different client, network traffic combinations. The following experiments were repeated three times for unbiased statistical evaluation, and only the average values were taken. During each streaming session, the video content runs up to 3 minutes.

5.1. Experiment 1: QoE performance using 3G, 4G, and 5G network traces for single and parallel DASH clients.

This set of experiments was carried out based on 15 selected network traces. We customized the selected traces for every sample with 4-second granularity. Linux TC applies

[7] Stall Ratio = 100 * (sum of the total stall duration / entire video duration); The "stall duration" indicates streaming pauses during the playback to fill up the buffer again

[8] https://github.com/itu-p1203/itu-p1203

changes to the link bandwidth after every 4 seconds so that at least two segments can easily download between 4-second intervals. This set of experiments covers the successive scenarios: a) Single DASH client with and without background (cross) traffic to stream DASH content, and b) Three concurrent DASH clients without any background (cross) traffic streaming the same DASH content at the same time.

Due to space limitations and no significant difference in video performance observed by adding the background traffic, we only shed light on the result with no background traffic for the first set of single client experiments. Figure 4 presents the overall single client video streaming performance for three ABS algorithms, three network type link utilization, and two transport protocols. Figure 4 contains the QoE output of video streaming performance metrics: Average Bit-rate, P1203 MOS, Quality Switches, Stall Ratio, and Startup Delay, respectively, from top to bottom.

Bit-rate. The rate-based Conventional ABS algorithm performs best to choose average bit-rate over both transport (TCP and QUIC) and three networks (3G, 4G, and 5G) scenarios. For the most part, all three ABS algorithms choosing a higher average bit-rate over TCP transport than QUIC. The average bit-rate metric results provide an interesting insight that the difference of choosing bit-rate deliberately downturn from the 3G to 5G network scenario over both TCP and QUIC. Herby, video streaming performs nearly similar over TCP and QUIC transport when there is high bandwidth available in the link; otherwise, TCP outperforms QUIC.

P1203 MOS. The MOS results retain the same observation for the bit-rate metric. Each of the cases' buffer-based BBA ABS algorithm underperforms, and the hybrid Arbiter ABS algorithm performs moderately. In the case of each ABS algorithm, TCP transport has higher MOS values predominantly.

Quality Switches. Video streaming has undergone the most quality changes for the BBA algorithm and least quality changes for the Conventional algorithm. As Conventional adopts the segment quality decision based on estimated bandwidth, it helps to get consistent video streaming quality. In general, the difference in quality switches over TCP and QUIC for all ABS algorithms is negligible. In the 5G network, the quality switches' amplitude is roughly equal and less than 3G and 4G network over TCP and QUIC.

Stall Ratio. The obtained results show that Conventional predominately experienced high and BBA experienced low stall events over TCP and QUIC. From the earlier Bit-rate and MOS results, we have observed that Conventional always rely on high-quality segments. Thus, it took more time to rebuffer while the buffer empties than the BBA approach. On average, all ABS algorithms suffered severe stall events on 3G and 4G networks, and QUIC holds higher stall events than TCP.

Startup Delay. Interestingly, this metric shows that both TCP and QUIC exhibit nearly equal startup delay for each ABS algorithm. Figure 7 helps to interpret similar startup delay results. We noticed the ABS algorithm always downloaded video segments from the low quality over both TCP and QUIC. In this set of experiments, the link's bandwidth utilization changed after each 4-second interval, influencing the DASH player to download at least two simultaneous segments in the same bandwidth condition of 4-second interval. The first two segments always maintain the same low quality, which we set an initial buffer limit to start playback. Thus, there is no such significant difference in startup delay.

The result might vary if there was a larger initial buffer threshold to start the playback. However, the 3G network holds a higher average startup delay for all ABS algorithms and transports (TCP and QUIC) than the 4G and 5G networks.

The second set of parallel clients' experiment results is depicted in Figure 5. In this experiment, the Conventional ABS algorithm holds the highest bit-rate for all combinations. Likewise, the single client experiment TCP is still aggressive to download the higher bit-rate than QUIC. As a consequence, each ABS algorithm predominantly gains higher P1203 MOS using TCP transport. However, parallel clients have the rest of the metrics' behavior, nearly similar to the single client. Due to the sharing bandwidth resource for multiple clients, more stall events occur, and overall QoE of video streaming performance is reduced than the single client as natural.

5.2. Experiment 2: QoE performance for extreme bandwidth fluctuation.

As stated in [Raca et al. 2020a], the collected 5G trace observed many bandwidth fluctuations due to a lack of 5G base stations across all driving routes, forcing the device to use 4G and 3G. Thus, this experiment has conducted under a single driving mobility trace (Bandwidth: mean=2.56 and std=2.43 Mbps) of the 5G network with 1-second granularity. Linux TC applies changes to the link bandwidth after every 1 second to emulate a restless connection switch scenario. This experiment covers a single DASH client with and without background (cross) traffic scenario to stream DASH content.

We observed that differences in QoE metrics' output with and without background (cross) traffic are negligible. Due to space limitations, Figure 6 describes the performance of single client video streaming with background traffic in terms of QoE metrics using a scenario of frequent fluctuations in link bandwidth usage. To download the desirable bit-rate, each of the ABS algorithms obtains the highest bit-rate over TCP, and as a consequence, TCP has higher P1203 MOS values than QUIC. BBA suffered most quality changes in the Quality Switches scenario, and Conventional suffered least quality changes over TCP and QUIC.

It is visible in Figure 7, which has drawn from one of the random samples of this experiment that Conventional was more constant to select the quality of the segments using TCP, and BBA often changes the segment quality using both TCP and QUIC. Also, Conventional and Arbiter suffered nearly zero stall events over TCP. Both algorithms rely on bandwidth estimation to select the quality segments; hence TCP transport facilitates adjusting to the get properly requested segments and filling the player buffer in time. On the other hand, BBA relies on buffer status to decide each segment and holds low quality. Thus, players' buffer always remains fill-up with low-quality segments, leading to no stall event over both TCP and QUIC.

Lastly, the Startup Delay result is distinct from the previous experiment. In this case, each ABS algorithm has taken startup playback delays slightly different from each other. The fluctuation of the link bandwidth in the granularity of 1 second might lead to this. In the end, BBA as an ABS algorithm and QUIC as a transport protocol has occupied higher startup delay than others.

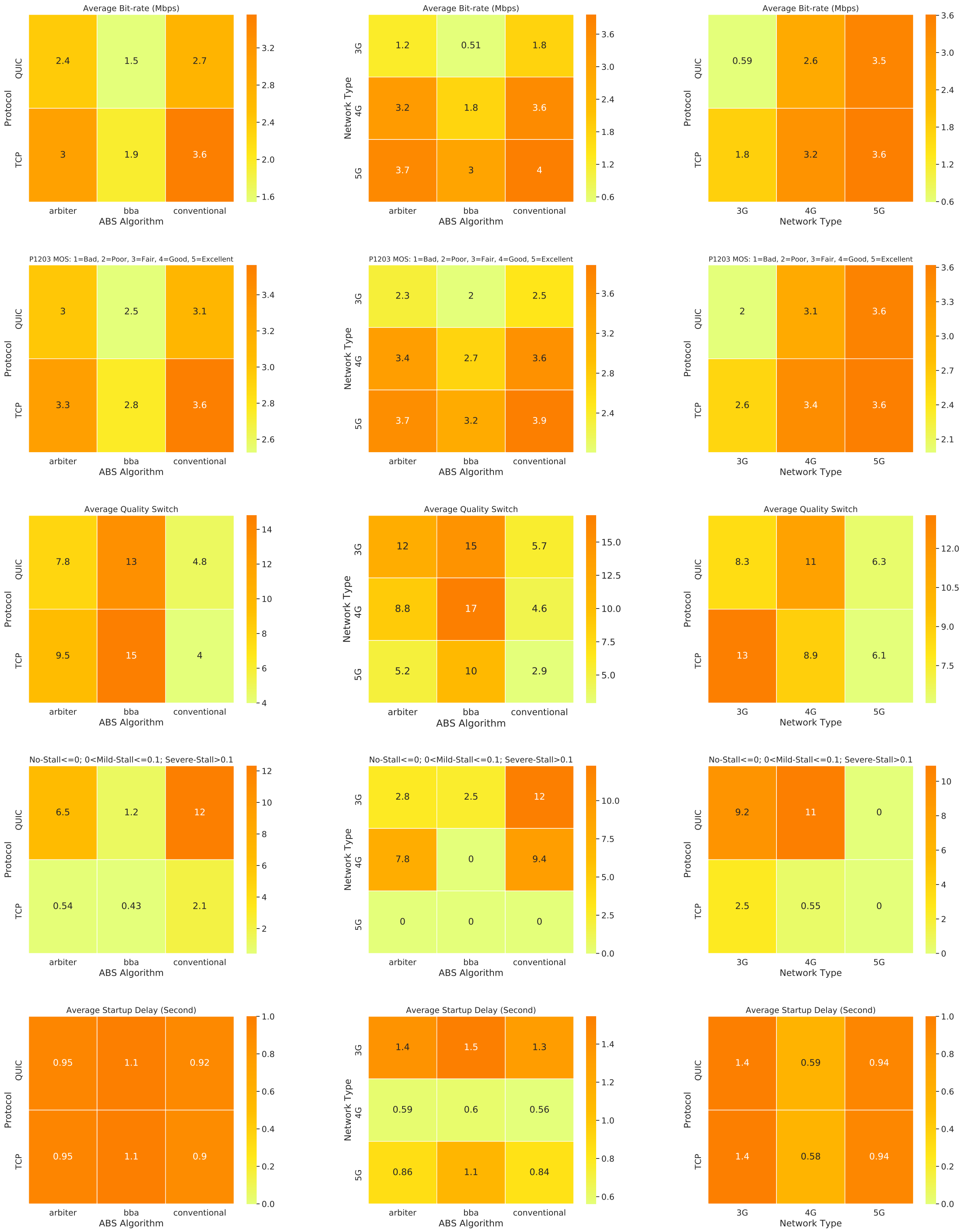

Figure 4. Experiment (1): Single Client- Average Bit-rate, P1203 MOS, Quality Switches, Stall Ratio, Startup Delay: Protocol vs ABS Algorithm (left), Network Type vs ABS Algorithm (center) and Protocol vs Network Type (right)

6. Conclusion and Future Work

In this paper, we investigated the three strategies (Rate, Buffer, and Hybrid) based ABS algorithm streaming performance over TCP and QUIC using different cellular network

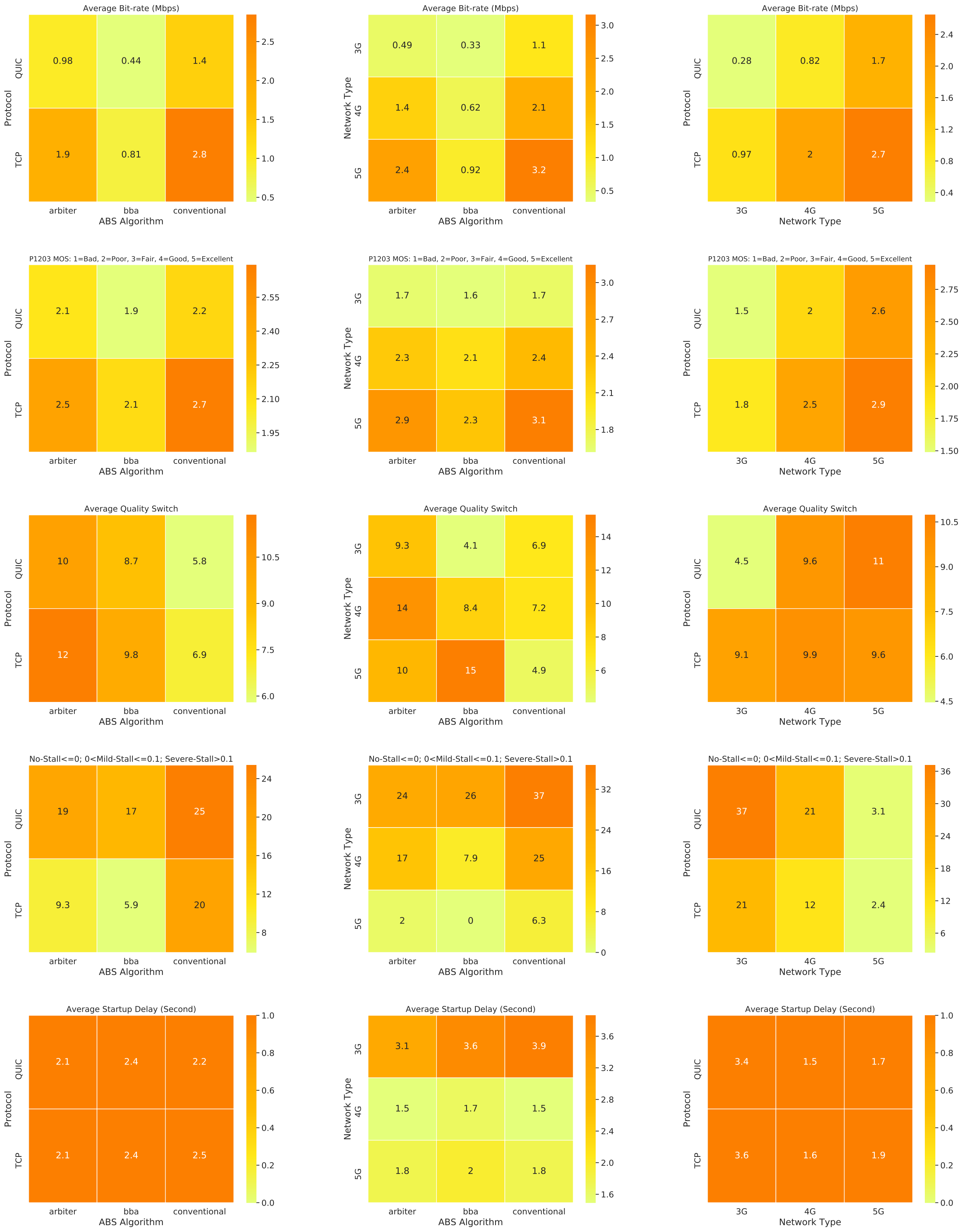

Figure 5. Experiment (1): Parallel Client- Average Bit-rate, P1203 MOS, Quality Switches, Stall Ratio, Startup Delay: Protocol vs ABS Algorithm (left), Network Type vs ABS Algorithm (center) and Protocol vs Network Type (right)

traces in a controlled testbed. Based on the well-known QoE metrics, our empirical

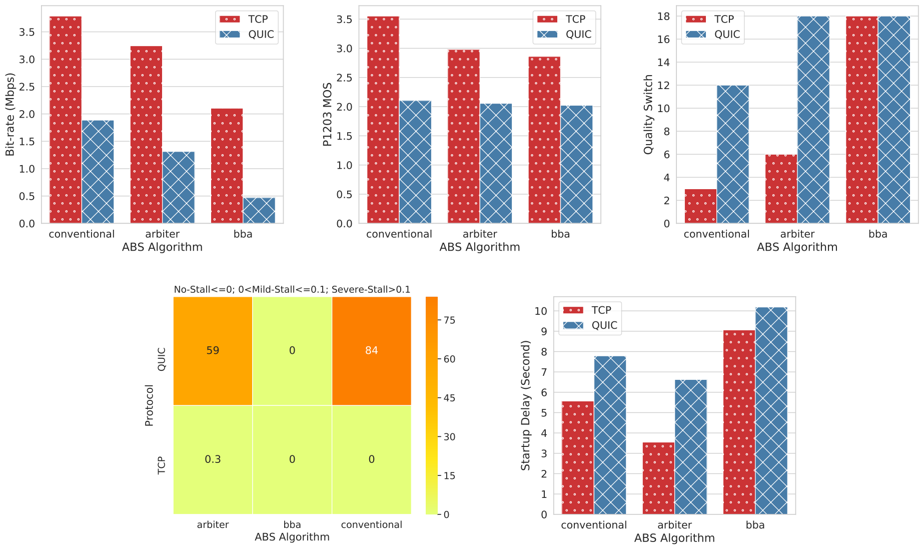

Figure 6. Experiment (2): Single Client- Average Bit-rate, P1203 MOS, Quality Switches, Stall Ratio, Startup Delay: With background traffic

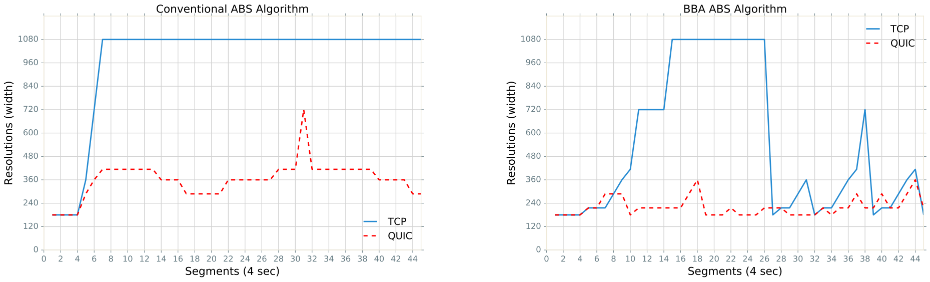

Figure 7. Single Client- Segment selection of ABS algorithms in term of quality (resolution) during the 3-minute (4 seconds * 45 Segments) video session

study provided all ABS algorithms using TCP transport, achieved a high quality of video streaming performance under varying network conditions, and verify the earlier work [Bhat et al. 2017] conclusion. We also found a rate-based Conventional ABS algorithm provides considerably better performance compared to other algorithms. The poor performance over QUIC transport indicates that the traditional state-of-art ABS algorithms are built mainly on TCP in mind. As a result, despite HOL issues in HTTP/1.1, TCP still performs better than HTTP/3 over QUIC. To deal with the IETF[9] mentioned QUIC feature, state-of-art ABS algorithms require modification in terms of segment requests to embrace the potential benefit of QUIC multiplexing, disable HOL blocking, congestion control, and service migration. In the future, we plan to extend our empirical study adding more DASH segment size content, different buffer levels of DASH player, more realistic lousy network condition, and an alternative implementation of QUIC transport, taking into account the latest efforts made by IETF[10]. Moreover, it would be interesting to find out any novel approaches, in addition to existing findings, to modify the segment choice decision of ABS algorithms to achieve better performance over QUIC transport.

[9] https://datatracker.ietf.org/doc/draft-ietf-quic-recovery/

[10] https://datatracker.ietf.org/doc/draft-ietf-quic-transport/

Acknowledgment

This research was supported by the Innovation Center, Ericsson, Brazil, grant UNI.67. This document, however, does not necessarily represent Ericsson's official viewpoint.

References

Arisu, S. and Begen, A. C. (2018). Quickly starting media streams using quic. In *Proceedings of the 23rd Packet Video Workshop*, pages 1–6.

Belshe, M., Peon, R., and Thomson, M. (2015). Hypertext transfer protocol version 2 (http/2).

Bhat, D., Deshmukh, R., and Zink, M. (2018). Improving qoe of abr streaming sessions through quic retransmissions. In *Proceedings of the 26th ACM international conference on Multimedia*, pages 1616–1624.

Bhat, D., Rizk, A., and Zink, M. (2017). Not so quic: A performance study of dash over quic. In *Proceedings of the 27th workshop on network and operating systems support for digital audio and video*, pages 13–18.

Bishop, M. et al. (2020). Hypertext transfer protocol version 3 (http/3). *Internet Engineering Task Force, Internet-Draft draft-ietf-quic-http-27*.

Cisco, V. (2018). Cisco visual networking index: Forecast and trends, 2017–2022. *White Paper*.

Conviva (2020). *Conviva's State of Streaming Q1 2020*. `https://www.conviva.com/state-of-streaming`.

Dobrian, F., Sekar, V., Awan, A., Stoica, I., Joseph, D., Ganjam, A., Zhan, J., and Zhang, H. (2011). Understanding the impact of video quality on user engagement. *ACM SIGCOMM Computer Communication Review*, 41(4):362–373.

Fielding, R. and Reschke, J. (2014). Rfc 7230: Hypertext transfer protocol (http/1.1): Message syntax and routing. *Internet Engineering Task Force (IETF)*, 6(01).

Hayes, B., Chang, Y., and Riley, G. (2017). Omnidirectional adaptive bitrate media delivery using mptcp/quic over an sdn architecture. In *GLOBECOM 2017-2017 IEEE Global Communications Conference*, pages 1–6. IEEE.

Huang, T.-Y., Johari, R., McKeown, N., Trunnell, M., and Watson, M. (2014). A buffer-based approach to rate adaptation: Evidence from a large video streaming service. In *Proceedings of the 2014 ACM conference on SIGCOMM*, pages 187–198.

Iyengar, J. and Thomson, M. QUIC: A UDP-Based Multiplexed and Secure Transport. Internet-Draft draft-ietf-quic-transport-29, Internet Engineering Task Force. Work in Progress.

Kakhki, A. M., Jero, S., Choffnes, D., Nita-Rotaru, C., and Mislove, A. (2017). Taking a long look at quic: an approach for rigorous evaluation of rapidly evolving transport protocols. In *Proceedings of the 2017 Internet Measurement Conference*, pages 290–303.

Kua, J., Armitage, G., and Branch, P. (2017). A survey of rate adaptation techniques for dynamic adaptive streaming over http. *IEEE Communications Surveys Tutorials*, 19(3):1842–1866.

Langley, A., Riddoch, A., Wilk, A., Vicente, A., Krasic, C., Zhang, D., Yang, F., Kouranov, F., Swett, I., Iyengar, J., et al. (2017). The quic transport protocol: Design and internet-scale deployment. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, pages 183–196.

Li, B., Wang, C., Xu, Y., and Ma, Z. (2016). An mmt based heterogeneous multimedia system using quic. In *2016 2nd International Conference on Cloud Computing and Internet of Things (CCIOT)*, pages 129–133. IEEE.

Li, Z., Zhu, X., Gahm, J., Pan, R., Hu, H., Begen, A. C., and Oran, D. (2014). Probe and adapt: Rate adaptation for http video streaming at scale. *IEEE Journal on Selected Areas in Communications*, 32(4):719–733.

Marx, R., De Decker, T., Quax, P., and Lamotte, W. (2019). Of the utmost importance: Resource prioritization in http/3 over quic. In *WEBIST*, pages 130–143.

Nguyen, M., Amirpour, H., Timmerer, C., and Hellwagner, H. (2020). Scalable high efficiency video coding based http adaptive streaming over quic. In *Proceedings of the Workshop on the Evolution, Performance, and Interoperability of QUIC*, pages 28–34.

Quinlan, J. J. and Sreenan, C. J. (2018). Multi-profile ultra high definition (uhd) avc and hevc 4k dash datasets. In *Proceedings of the 9th ACM Multimedia Systems Conference*, pages 375–380.

Raca, D., Leahy, D., Sreenan, C. J., and Quinlan, J. J. (2020a). Beyond throughput, the next generation: a 5g dataset with channel and context metrics. In *Proceedings of the 11th ACM Multimedia Systems Conference*, pages 303–308.

Raca, D., Manifacier, M., and Quinlan, J. J. (2020b). godash-go accelerated has framework for rapid prototyping.

Raca, D., Quinlan, J. J., Zahran, A. H., and Sreenan, C. J. (2018). Beyond throughput: a 4g lte dataset with channel and context metrics. In *Proceedings of the 9th ACM Multimedia Systems Conference*, pages 460–465.

Riiser, H., Vigmostad, P., Griwodz, C., and Halvorsen, P. (2013). Commute path bandwidth traces from 3g networks: analysis and applications. In *Proceedings of the 4th ACM Multimedia Systems Conference*, pages 114–118.

Thang, T. C., Le, H. T., Pham, A. T., and Ro, Y. M. (2014). An evaluation of bitrate adaptation methods for http live streaming. *IEEE Journal on Selected Areas in Communications*, 32(4):693–705.

Timmerer, C. and Bertoni, A. (2016). Advanced transport options for the dynamic adaptive streaming over http. *arXiv preprint arXiv:1606.00264*.

Zahran, A. H., Raca, D., and Sreenan, C. J. (2018). Arbiter+: Adaptive rate-based intelligent http streaming algorithm for mobile networks. *IEEE Transactions on Mobile Computing*, 17(12):2716–2728.

Zinner, T., Geissler, S., Helmschrott, F., and Burger, V. (2017). Comparison of the initial delay for video playout start for different http-based transport protocols. In *2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, pages 1027–1030. IEEE.