QoE Analysis of DASH Cross-Layer Dependencies by Extensive Network Emulation

Denny Stohr[*] Alexander Frömmgen[†] Jan Fornoff[†]
Michael Zink[◇] Alejandro Buchmann[†] Wolfgang Effelsberg[*]

{[*]DMS, [†]DVS } TU Darmstadt
{dstohr, wolfgang.effelsberg}@cs.tu-darmstadt.de
{froemmgen, buchmann}@dvs.tu-darmstadt.de

[◇]ECE Department
University of Massachusetts Amherst
zink@ecs.umass.edu

ABSTRACT

With the rising importance of video streaming in the Internet, dynamic adaptive streaming over HTTP (DASH) has been established as a key technology for video delivery. Yet, variable network conditions often result in a limited quality of experience (QoE)—with the interrelation of cross-layer network factors and DASH mechanisms widely unexplored. To understand the complex dependencies between DASH configurations and network conditions, we propose a systematic extensive large-scale emulation approach with state-of-the-art QoE metrics. Using this approach with a real DASH player in Mininet, we emulated more than $10,000$ combinations of static and dynamic network conditions and DASH configurations to derive their QoE.

The obtained results show that no single DASH configuration provides the highest achievable QoE. Depending on the network conditions, combinations of the TCP congestion control, segment sizes and the DASH adaptation algorithm provide higher QoE—showing the possibility of performance improvements. Furthermore, the extensive emulations show a linear relation between delay, loss and QoE that is mostly independent of bandwidth.

CCS Concepts

•**Networks** → *Network simulations;* •**Information systems** → **Multimedia streaming;**

1. INTRODUCTION

Video streaming is the most significant service in the Internet in terms of data volume [2]. It accounts for revenues in the range of billions of dollars to Over-The-Top (OTT) service providers like YouTube and Netflix [16]. Success in this competitive market requires a constantly high customer satisfaction—a low quality of experience (QoE) is not tolerated [12]. Due to changing network conditions, such as fluctuating throughput, achieving a constantly high QoE is challenging.

Dynamic adaptive streaming over HTTP (DASH) copes with such changing conditions in OTT data delivery by dynamically adapting the video quality according to the bottleneck bandwidth. Therefore, the HTTP server provides each video segment in multiple representations (video bitrates). The DASH player adapts the video quality by downloading an appropriate segment (video bitrate) to strive for continuous playback. The player's decision is based on local metrics such as the experienced download throughput or the current player buffer level. Yet, DASH inherently suffers from low link utilization and a limited view on system parameters for adaptation decisions [19]. This often leads to playback interruptions (stalling) and a low, fluctuating playback quality. Their causes are hard to analyse and avoid in dynamic systems. Proposed novel DASH strategies focus on a subset of the delivery system and fail to address the cross-layer dependencies in the network stack.

In this paper, we propose to automate extensive network emulations for DASH with Mininet [7] to systematically vary network conditions and configuration parameters. In combination with the current research for QoE-metrics in adaptive video streaming, this approach enables a systematic analysis of cross-layer dependencies and a QoE driven development. The recently increased processing capabilities, e.g., in the cloud, support such extensive emulations. To show the benefits and feasibility of a large-scale emulation approach, this paper examines different DASH *bandwidth estimators*, *video segment sizes* and TCP *congestion control (CC) algorithms* in various network scenarios regarding *bandwidth*, *latency* and *packet loss*. Beside *static scenarios*, we employ *dynamic network conditions* replaying real bandwidth traces con-

Internet-QoE, August 22–26, 2016, Florianopolis, Brazil

 ISBN 978-1-4503-4425-8/16/08…$15.00

DOI: http://dx.doi.org/10.1145/2940136.2940141

Table 1: Overview of the investigated configurations and conditions on different layers.

	Zone/Layer	Configuration	Instantiations	Related Work		# Combinations
Viewer Mechanism	QoE		$\mathrm{MOS}_{\mathrm{Stal.}}$, $\mathrm{MOS}_{\mathrm{Rep.}}$, $\mathrm{MOS}_{\mathrm{Avg.}}$, $\mathrm{Bitrate}_{\mathrm{Avg.}}$	[8], [9], [14]		
	Video Content	Segment Size [s]	2, 10	[14]		2
	DASH Player	Bandwidth Estimator	$\mathrm{Adapt}_{\mathrm{Def.}}$, Adapt_3, Adapt_{10}	[21]		3
	Transport Protocol	Congestion Control	Vegas, Cubic, Reno	[4], [5], [10], [15]		3
Network Conditions		Bandwidth [Mbps]	static: 0.384, 1.2, 2, 5, 10, 20, 50	[14]	7	+ = 19
			dynamic: Bus and Car [5]	[5]	12	
		Latency [ms]	10, 20, 35, 50, 100, 150			6
		Packet loss [%]	0, 0.5, 1.0, 2.0, 5.0	[14]		5
# Combinations					3,780 + 6,480 =	10,260
# Emulations					10 * 10,260 =	102,600

taining sudden bandwidth changes as experienced in the wild. Since the goal of video streaming is to achieve the best possible QoE for viewers, we quantify the impact of the system using established quality metrics. These consider the expected user satisfaction and measure QoE based on the playback quality and stalling, which have shown to be the most significant impact factors for HTTP adaptive streaming (HAS) [19].

The contributions of this paper are the following:

1. A methodology to systematically vary application configurations and network conditions in extensive network emulations to examine their impact on QoE metrics.
2. Based on extensive emulations of over 10,000 combinations, we derive a linear model for stalling and QoE based on loss and delay.
3. We identify significant potential for an increased QoE by applying combinations of configurations parameters for specific network conditions.

The remainder of this paper is structured as follows: Based on an overview of related work in Section 2, Section 3 describes the large-scale emulation approach and its variations. Section 4 presents detailed analysis results, such as the derived QoE model. Finally, Section 5 concludes the paper and gives an outlook on future work.

2. RELATED WORK

DASH has been studied in the past with a focus on various aspects such as the TCP congestion control (CC), adaptation strategies as well as QoE in mobile and wired networks. We arrange the overview for this wide range of publications along the lines of Table 1, presenting the viewers' perspective and QoE metrics, the configuration parameters of the application mechanisms, and the impact of different network conditions.

In general, the perceived QoE is determined by stalling events, startup delays, the fidelity of the video representation and switches of the representation. Rodríguez *et al.* [18] examined the effect of video representation switches (VRS) showing that the frequency of representation switches impacts the viewers' subjective perception significantly. Hoßfeld *et al.* [8] quantified the impact of such VRS. The authors found that the time spent on the highest representation layer has a more significant impact on the Mean Opinion Score (MOS) than the frequency of switches and introduce a QoE model based on their findings. In order to quantify stalling effects during playback, Hoßfeld *et al.* [9] introduced a model to estimate MOS from stalling frequency and average duration based on user studies for YouTube.

Maki *et al.* [14] discussed the relation between network quality of service (QoS) factors (DASH segment length, buffer length and packet loss rate) and the performance of DASH based on stalling patterns, duration, and frequency. Observed stalling events are further modelled for their relation to QoE using a linear regression model. Our study differs from the presented work by taking into account additional network characteristics, namely latency, TCP CC and adaptation algorithms. Thus, we are able to cover a larger area of involved mechanisms on different layers. Further, we include the dynamic aspects of such video streaming systems by considering the adaptation in DASH playback and analysing configurations using dynamic bandwidth traces. For the QoE evaluation in our work, we rely on the findings presented by Hoßfeld *et al.* [8][9]. However, our work is not limited to those QoE models.

The performance of different classes of adaptation strategies (throughput-based or buffer-based) under changing network conditions has been researched by Thang *et al.* [21]. Further, novel adaptation strategies have been introduced such as *BOLA* Spiteri *et al.* [20]. For the experiments conducted in our work, we use three throughput-based adaptation algorithms with a varying trade-off between reaction time and adaptation hysteresis (see Section 3). Newly developed adaptation strategies (e.g., as JavaScript DASH player) can be integrated in our emulation environment in the future.

The interactions between TCP transmissions and their impact on HAS sessions were analysed in multiple publications. Esteban *et al.* [4] examined the influence of packet losses for three TCP transmission phases in TCP Cubic and their effect on video segment downloads. A prediction based adaptation algorithm for mobile DASH streaming sessions is proposed by Evensen *et al.* [5]. Here, they present the influence of TCP CC algorithms on the number of packet drops, as well as the influence of the segment size on the congestion window. A mea-

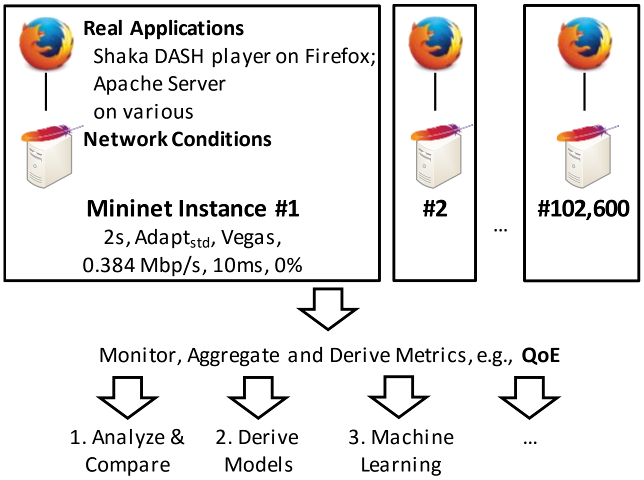


Figure 1: Extensive emulations with real applications on multiple network conditions

surement study by Huang *et al.* [10] examined three major streaming platforms and their performance in relation to effects on TCP having competing clients. For two novel TCP mechanisms, *newCWV* and *newCWV*-pacing, Nazir *et al.* [15] have examined their performance in cross traffic and non-cross traffic scenarios, showing that *newCWV*-pacing improves performance compared to TCP NewReno in DASH streaming scenarios. In contrast to previous work, we examine different existing TCP CC algorithms in relation to other cross-layer aspects and configurations (see Table 1). Thus, we provide an extensive, QoE-driven view on the interdependencies for DASH streaming. Our tests cover a wide range of both static and dynamic network conditions.

3. EXTENSIVE EMULATIONS

In this section, we present our extensive network emulation approach for DASH. Based on an architecture overview, we describe relevant layers and instantiations of their configurations in a top-down fashion (Table 1).
Architecture Overview: The viewers' QoE depends on the configuration of the application and the network conditions in a complex, non-trivial fashion. We propose to evaluate applications, the impact of their configuration and the network conditions on QoE metrics in large-scale emulations. Figure 1 shows the architecture we used to emulate a total of 10,260 combinations of application configurations and network conditions (see Table 1). This approach extends a recent extensive emulation approach, which reduced page load times for traditional non-video HTTP traffic by optimizing application configurations [6].

For the emulations, we rely on Mininet [7], as it allows the use of real applications, i.e. the Shaka DASH player[1] in Firefox and an Apache Web Server, on specified network conditions. The collection of all monitored and derived metrics allows three major steps: *1)* a detailed analysis and comparison to investigate the impact of configurations and network conditions on QoE, *2)* the derivation of models which describe and explain these dependencies, *3)* finally, we envision to use machine learning for understanding results and predicting future behaviour. We implemented a framework to support our methodology and execute all emulations parallelized on multiple server instances. Each configuration is executed 10 times to ensure statistically meaningful results, resulting in a total number of 102,600 emulations. The metrics of each emulation are stored and further aggregated, e.g., to QoE metrics.

Furthermore, we extended Mininet for easy control of applications such as Firefox and Apache as well as mechanisms for replaying bandwidth traces. As Mininet runs on a single Linux kernel, the emulation is sensitive to high CPU loads. Our framework ensures reproducible results by tracking metrics such as the CPU usage.

QoE Metrics: The performance evaluation of all emulated configurations is based on three QoE measures. First, we employ a model introduced by Hoßfeld *et al.* [9] to derive a MOS score based on stalling frequency and length, denoted as $\text{MOS}_{\text{Stal.}}$. Next, as a means to evaluate the video quality, we apply the model presented by Hoßfeld *et al.* [8], providing a MOS based on the portion during which the session stayed in the highest quality representation. We will refer to this metric as $\text{MOS}_{\text{Rep.}}$. To show the combined influence of both stalling and representation, we derive $\text{MOS}_{\text{Avg.}}$ as the average of the two metrics (see Section 4.1).

As the proposed models are based on 30 second test sequences, we derive the average of four independent scores for each 30 second fragment of our 120s test intervals. For reproducibility, we use the same dataset as originally used to derive the $\text{MOS}_{\text{Rep.}}$ model in [9] (c.f. Section 3). Unlike the original work presenting this model, we also include quality representations exceeding 0.807 Mbps (having a resolution of 640×360 pixels), classified as high-quality layers, to explore potential for high bandwidth configurations. The fact that we include this higher bitrate layers is addressed by an additional analysis of results based on the more fine granular average bit rate ($\text{Bitrate}_{\text{Avg.}}$) QoS metric.

Video Content: The video content in our tests is based on the open movie *Tears of Steel*[2] provided in a dataset prepared by [13]. It consists of nine H.264-AVC encoded video representation layers having bitrates between 0.253 and 10 Mbps with resolutions between 480×270 and 1920×1080 pixels in 2s and 10s segment size configurations, offering similar resolutions and bitrates as used in practice by YouTube, Netflix and Apple [11].

DASH Player: We used the JavaScript based Google *Shaka* DASH player in the Firefox web browser (Version 42.0). The player has a default playback buffer size of 15 seconds and uses a throughput-based adaptation algorithm. The default adaptation algorithm ($\text{Adapt}_{\text{Def.}}$)

[1] https://github.com/google/shaka-player

[2] https://mango.blender.org/

estimates conservatively by using the minimum of the segment download rates of the last 3 and 10 segments, respectively, as the input for an exponentially weighted moving average (EWMA). To evaluate this design decision, we added two EWMA-based adaptation algorithms, which only rely on 3 (Adapt_3) or 10 (Adapt_{10}) segments for calculation. We modified parameters in the `SimpleAbrManager`-class `MIN_SWITCH_INTERVAL=5`, default 30; `MIN_EVAL_INTERVAL=1`, default 3) to allow for faster adaptation, given the 120 second test sequences.

Transport Protocol: As DASH uses HTTP and TCP, the TCP congestion control algorithm influences the achievable throughput and interacts with DASH's adaptation loop. We investigate three TCP congestion control algorithms: First, we include TCP Cubic that adjusts the congestion window size (cwnd) based on a cubic function for high bandwidth utilization. In contrast, TCP New Reno is based on the additive increase, multiplicative decrease concept for cwnd control, adding the concept of fast recovery to its preceding implementation TCP Tahoe. Last, we include TCP Vegas because it is sensitive to changes of the round trip time (RTT) and adjusts the cwnd by detecting changes in the RTT. [1] provides a detailed discussion of TCP CC algorithms.

Network Conditions: The choice of a wide range of realistic network conditions is fundamental for allowing insights into the dependencies between configurations. We consider two approaches: *i) static conditions* as combinations of bandwidth, delay and packet loss probability, and *ii) dynamic bandwidth trace replay* using real-world traces, e.g., collected from mobile devices [17], and replay these by changing the traffic shaper accordingly. We argue that meaningful insights require both approaches. The reduced complexity of static conditions allows a systematic comparison, whereas dynamic bandwidth traces allow to analyze the impact of fluctuating bandwidth conditions as they are experienced in reality.

4. EVALUATION AND SYNTHESIS

This section presents a first evaluation of the QoE driven extensive emulation approach for DASH. In particular, we present a comparative QoE analysis of configurations depending on network conditions. Based on this, we use the huge data set to derive a model for QoE based on network conditions. Finally, we provide evidence for performance gains by changing configurations depending on the network characteristics.

4.1 Analyse and Compare

Congestion Control, Segment Size and Static Bandwidth: Figure 2 illustrates the $\text{MOS}_{\text{Stal.}}$ variance aggregated over repetitions of the same configuration for common delay and packet loss scenarios. As we observed no significant performance differences for varying adaptation strategies, we only show the results of the standard algorithm $\text{Adapt}_{\text{Def.}}$. The range of observed $\text{MOS}_{\text{Stal.}}$ values indicates significant performance differences for static bandwidth conditions depending on the TCP CC

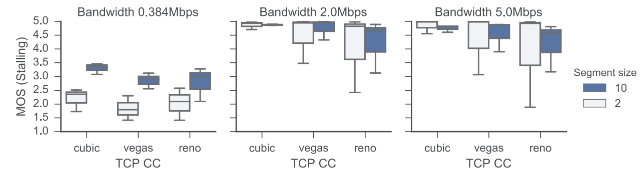


Figure 2: $\text{MOS}_{\text{Stal.}}$ for usual network QoS aggregating delays of 35, 50 and 100ms and a loss of 1%

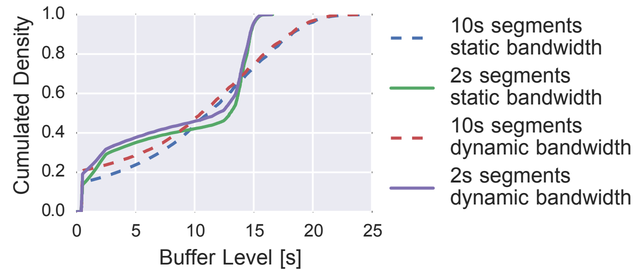


Figure 3: Distribution of the DASH player buffer level depending on the segment size

algorithm and segment sizes. Here, 10s segment sizes clearly dominate in terms of $\text{MOS}_{\text{Stal.}}$ across all observations. For bandwidths higher than 0.384 Mbps, TCP Cubic provides a performance advantage over Vegas and Reno, showing nearly no reduction in the $\text{MOS}_{\text{Stal.}}$.

Player Buffer Comparison: Figure 3 compares the distribution of buffer fill levels across all configurations between 2s and 10s segment sizes. In both cases, the buffer fill state is strongly cumulated close to zero, which is to be expected given that a large part of the tested network settings lead to significant stalling, e.g., when bandwidths are low or the packet loss rate is high. A second bend can be seen for the 2s segment configuration caused by the player having loaded exactly one segment from the initial or buffer under-run state. For full buffers, there is a clear distribution difference past the 60% density between 2s and 10s configurations, having the range of buffer fill states spread out in the 10s case whereas 2s segments have a high cumulation rate around the fixed 15s upper bound for the buffer level in Shaka player. For 10s segments this difference can be explained based on their higher segment length compared to 2s segments in relation to the total buffer size of 15s, which allows them to overfill the players' buffer. However, the median buffer level is still higher for 2s segments which is somewhat counter-intuitive given this observation.

Investigating the dependencies between the buffer level and stalling events, we found that the total stalling duration (frequency $\times$ duration) for 2s and 10s segments is roughly equal. 2s segments, however, lead to four times more stalling events, whereas 10s segments have on average four times higher stalling durations. The model for $\text{MOS}_{\text{Stal.}}$ penalizes a higher stalling frequency, and therefore prefers 10s segments.

QoE Metrics Comparison: A QoE-driven development of applications depends on the choice of the QoE metric, respectively the aggregation of multiple metrics. To evaluate the impact of the aggregation depending on

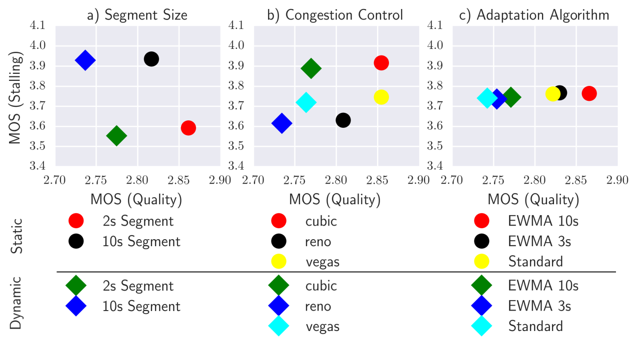


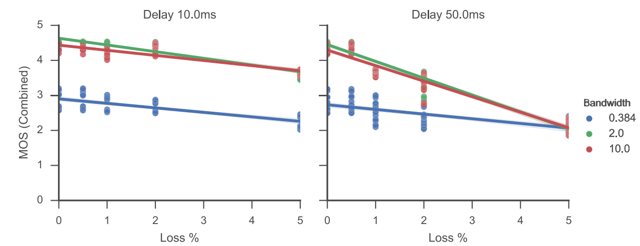
Figure 4: QoE metric comparison for different configurations and static and dynamic network conditions.

the application configuration, Figure 4 presents different configurations and their QoE metrics. The comparison for different segment sizes (see Figure 4 a) shows that 10s segments increase the $\text{MOS}_{\text{Stal.}}$, whereas 2s segments slightly increase the $\text{MOS}_{\text{Rep.}}$ (note the different scales). For the congestion control and adaptation algorithm (see Figure 4 b and c), one configuration dominates all other configurations. Therefore, the choice of the configuration does not depend on the aggregation of the metrics. Static and dynamic network conditions show similar relations. Based on these results, the configuration (Cubic, 10s segments, single bandwidth estimation with 10 segments) leads to the highest average $\text{MOS}_{\text{Avg.}}$. An analysis of the variances is difficult given the huge variety of network conditions.

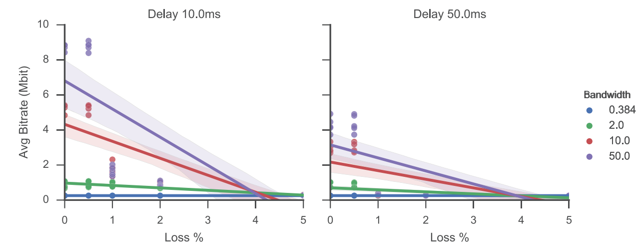
4.2 QoE Model Derivation based on QoS

In Figure 5 the relationship between loss, $\text{MOS}_{\text{Avg.}}$, and $\text{Bitrate}_{\text{Avg.}}$ is shown by applying a generalized least squares linear regression model. First, we will discuss the observations for Figure 5a. Using the bandwidth of 2 Mbps as an example, for low delays of 10ms, the packet loss provides a high confidence ($R^2 = 0.855$; F-Measure < 0.01) for the achievable $\text{MOS}_{\text{Avg.}}$. With larger delays of 50ms, the negative slope decreases, thus showing that packet loss leads to more severe loss of playback quality given a higher delay ($R^2 = 0.929$; F-Measure < 0.01).

Given that a strong linear correlation between loss, delay and the $\text{MOS}_{\text{Avg.}}$ measure can be observed for all bandwidth configurations except 0.384 Mbps, we exclude this configuration for building the following regression model and present their dependency with $\text{MOS}_{\text{Rep.}}$ for all TCP CC and adaptation algorithms. Using two independent variables, a fit of ($R^2 = 0.631$; F-Measure < 0.01) can be achieved. The resulting regression model based on both independent variables, given that the bandwidth is ≥ 1.2 Mbps, can be defined as $MOS_{\text{Stalling}} = 5.8 - 0.4 * \text{loss} - 0.02 * \text{delay}$. To further examine the relationship between bandwidth, delay and video quality, Figure 5b shows a linear regression model fitted to $\text{Bitrate}_{\text{Avg.}}$. The comparison between bandwidth configurations shows a clear influence to $\text{Bitrate}_{\text{Avg.}}$. However, with delays of 50ms, they show the most severe loss in



(a) Linear regression showing the relation between $\text{MOS}_{\text{Avg.}}$ and loss comparing bandwidth configurations as well as delays between facets for $\text{Adapt}_{\text{Def.}}$



(b) Linear regression showing the relation between $\text{Bitrate}_{\text{Avg.}}$ and loss comparing bandwidth configurations as well as delays between facets for $\text{Adapt}_{\text{Def.}}$ to illustrate a *low* fit of the regression model

Figure 5: Regressions

video quality as well as a very *low fit* for the derived linear regression model ($R^2 = 0.445$; F-Measure < 0.01). There is no linear relation between the network conditions and the $\text{Bitrate}_{\text{Avg.}}$, but for high level QoE metrics.

4.3 Potential for Improved QoE

So far, we used the extensive emulation results for analysing dependencies between configurations and network conditions. The overall goal, however, is providing a higher QoE. To analyse the potential for an improved QoE, Figure 6 shows an aggregated comparison of all tested mechanism configurations for both static and dynamic bandwidth conditions. In accordance with the analysis so far, the configuration (TCP Cubic, 10s segment size, single 10s bandwidth estimation, ◆) is superior to the other configurations for static conditions. A solution, however, which always uses the configuration (◆) that leads to the best achievable QoE per network condition outperforms all other configurations. This shows that DASH applications which are tuned to provide constantly high performance in changing conditions fail to provide the best possible QoE in static network conditions. For dynamic bandwidths, the QoE difference between the best configuration (◆) and the best configuration per network condition (◆) is even higher, showing a huge improvement potential.

5. CONCLUSION

In this work, we propose a QoE-driven extensive emulation approach to analyse cross-layer dependencies for DASH. For a first evaluation, we emulated 18 DASH configurations on 570 network conditions, containing both static and dynamic bandwidth traces. In contrast to examining adjustments on a single configuration pa-

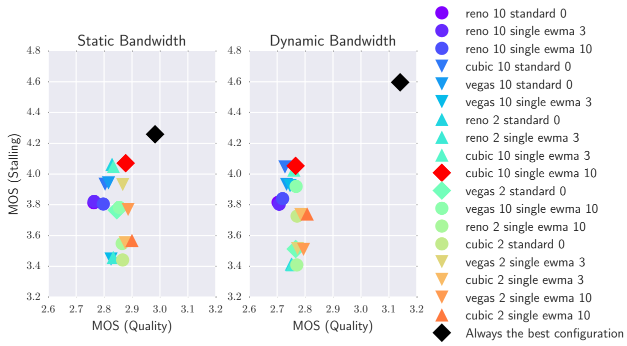


Figure 6: The QoE metrics for all combinations of bandwidth estimators, segment sizes and congestion controls compared with an adaptive solution which uses the best mechanism combination per network condition

rameter or network condition, this approach with 10,280 emulated combinations in 102,800 emulations and a total video playback time of 142 days provides the basis to *understand complex dependencies* between configurations on different layers and network conditions.

In particular, our results show that *i)* adaptive video streaming fails to provide constantly high video quality for static bandwidth conditions; *ii)* the impact of available bandwidths on the QoE of DASH is generally very low—especially with rising delays and loss rates; *iii)* delay and loss rates allow the retrieval of a MOS score based on stalling; *iv)* larger segment sizes are generally favourable for improving the MOS, in particular in combination with TCP Cubic.

Discussion: As no single existing DASH configurations outperforms all other configurations, there is a significant performance potential. Our approach helps to develop a new solution, which outperforms existing configurations by applying transitions between configurations based on experienced network conditions. Furthermore, given a large bandwidth, its influence on the QoE is surpassed by delay and loss. This shows the importance of reducing latency for the content providers, e.g., using content delivery networks (CDNs).

Future work: For the DASH scenario, we will broaden the scope, e.g., we will add network condition variations by introducing cross traffic. Furthermore, we will vary additional configuration parameters, such as the target buffer size or the `BANDWIDTH_UPGRADE_TARGET`, which is used in Shaka to avoid oscillations. Finally, we will include more adaptation algorithms (e.g., buffer-based strategies) by using additional DASH players, such as the DASH-IF player [3].

6. ACKNOWLEDGMENT

This work has been funded by the German Research Foundation (DFG) as part of the projects A02 and C03 in the Collaborative Research Center (SFB) 1053 MAKI.

7. REFERENCES

[1] C. Callegari, S. Giordano, *et al.* “A survey of congestion control mechanisms in linux tcp.” in *Commun. Comput. Inf. Sci.*. vol. 279 CCIS. 2014, pp. 28–42.

[2] *Cisco visual networking index, 2014-2019 white paper.* http://www.cisco.com/c/en/us/solutions/collateral/service-provider/ip-ngn-ip-next-generation-network/white_paper_c11-481360.pdf.

[3] *Dash if client.* https://github.com/Dash-Industry-Forum/dash.js.

[4] J. Esteban, S. A. Benno, *et al.* “Interactions between http adaptive streaming and tcp.” in *NOSSDAV ’12*. Toronto, Ontario, Canada: ACM, 2012, pp. 21–26.

[5] K. Evensen, T. Kupka, *et al.* “Adaptive media streaming to mobile devices: challenges, enhancements, and recommendations.” *Adv. Multimed.*, pp. 1–21. 2014.

[6] A. Frömmgen, P. Wagner, and A. Buchmann. “Simulation-based retrieval of adaptation knowledge.” in *CoNEXT Student Workshop*. ACM, 2015.

[7] N. Handigol, B. Heller, *et al.* “Reproducible network experiments using container-based emulation.” in *CoNEXT ’12*. Nice, France: ACM, 2012, pp. 253–264.

[8] T. Hoßfeld, M. Seufert, *et al.* “Assessing effect sizes of influence factors towards a qoe model for http adaptive streaming.” *QoMEX ’14*, pp. 111–116. 2014.

[9] T. Hoßfeld, R. Schatz, *et al.* “Internet video delivery in youtube: from traffic measurements to quality of experience.” *Lecture Notes in Computer Science*. vol. 7754. E. Biersack, C. Callegari, and M. Matijasevic, Eds., pp. 264–301. 2013.

[10] T.-Y. Huang, N. Handigol, *et al.* “Confused, timid, and unstable.” in *IMC ’12*. ACM Press, 11/2012, p. 225.

[11] C. Kreuzberger, B. Rainer, *et al.* “A comparative study of dash representation sets using real user characteristics.” in *NOSSDAV ’16*.

[12] S. S. Krishnan and R. K. Sitaraman. “Video stream quality impacts viewer behavior: inferring causality using quasi-experimental designs.” *IEEE/ACM Trans. Netw.*. vol. 21. no. 6, pp. 2001–2014. 12/2013.

[13] S. Lederer, C. Müller, and C. Timmerer. “Dynamic adaptive streaming over http dataset.” in *MMSys ’12*. New York, NY, USA: ACM Press, 02/2012, p. 89.

[14] T. Maki, M. Varela, and D. Ammar. “A layered model for quality estimation of http video from qos measurements.” in *SITIS ’15*. 2015, pp. 591–598.

[15] S. Nazir, Z. Hossain, *et al.* “Performance evaluation of congestion window validation for dash transport.” in *NOSSDAV*. Singapore: ACM, 2014, pp. 67–72.

[16] *Netflix : Quarterly earnings.* http://ir.netflix.com/results.cfm. (Accessed on 02/24/2016).

[17] H. Riiser, P. Vigmostad, *et al.* “Commute path bandwidth traces from 3g networks.” in *MMSys ’13*. ACM.

[18] D. Z. Rodríguez, Z. Wang, *et al.* “The impact of video-quality-level switching on user quality of experience in dynamic adaptive streaming over http.” *EURASIP J. Wirel. Commun. Netw.*. vol. 2014. no. 1, p. 216. 2014.

[19] M. Seufert, S. Egger, *et al.* “A survey on quality of experience of http adaptive streaming.” *IEEE Communications Surveys Tutorials*, pp. 469–492. 2015.

[20] K. Spiteri, R. Urgaonkar, and R. K. Sitaraman. “BOLA: near-optimal bitrate adaptation for online videos.” *CoRR*. vol. abs/1601.06748. 2016.

[21] T. C. Thang, H. T. Le, *et al.* “An evaluation of bitrate adaptation methods for http live streaming.” *IEEE J. Sel. Areas Commun.*. vol. 32. no. 4, pp. 693–705. 2014.