YouTube Can Do Better: Getting the Most Out of Video Adaptation

Christian Moldovan‡, Christian Sieber*, Poul Heegaard†, Wolfgang Kellerer*, Tobias Hoßfeld‡

‡*Universität Duisburg-Essen, Germany*
{christian.moldovan,tobias.hossfeld}@uni-due.de

**Chair of Communication Networks, Technical University of Munich, Germany*
{c.sieber, wolfgang.kellerer}@tum.de

†*Norwegian University of Science and Technology, Trondheim, Norway*
{poul.heegaard}@item.ntnu.no

***Abstract*— YouTube, as one of the major HTTP Adaptive Streaming video services, accounts for a large fraction of today's Internet traffic. Therefore, it is important to understand how efficiently YouTube uses available network resources. Previous work observed that the YouTube player replaces previously buffered segments with higher quality segments. This is good for the user as it increases the average quality level. However, the lower quality level segments are discarded and their traffic is redundant and therefore wasted. In this paper, we use two independent approaches to evaluate the efficiency of YouTube's quality adaptation algorithm. The first approach performs regression based on previously collected video views from a large experimental data set. In the second approach we formulate a mixed integer linear program and calculate the optimal video quality adaptation. The results show that the simplistic regression approach gives an accurate estimation of the optimal adaptation. Furthermore, the optimization shows that the Quality of Experience (QoE) can be significantly improved compared to the actual average quality level observed in the real-world experiments, demanding for better video quality adaptation mechanisms by YouTube.**

I. Introduction

Today, HTTP Adaptive Streaming (HAS) is the dominant way of video delivery in the Internet. HAS is based on the wide-spread HTTP protocol and takes over its properties such as easy traversal of NAT-devices, firewall-friendliness, encryption in the shape of HTTPS and close-to-customer caching through content-delivery networks (CDNs). In HAS the video content is split into short chunks (e.g. 2 seconds) and each chunk is encoded into different quality levels. The individual chunks are then made available on standard HTTP servers. The location of the chunks and their encoding are given to the client through a manifest file. At the beginning of the playback, the streaming client first requests the manifest file. Afterward, it chooses the chunks according to its internal adaptation logic, for example based on the current throughput or buffer level. Dynamic Adaptive Streaming of HTTP (DASH) is an ISO standard which defines the structure and content of such a manifest file and is deployed by major video service providers such as YouTube or Netflix.

As users are shifting away from traditional video broadcast consumption to individual content selection through streaming services, user expectations are growing. Users expects the content to be available on all their devices and wherever they go. It is well known that stalling events and the video encoding bit-rate, i.e. the video resolution, have a significant impact on the acceptance rate and the Quality of Experience [1]. Therefore, it is important for the service provider to develop a sophisticated adaptation logic which can prevent stalling events even when faced with bottlenecked or unstable Internet connections, such as cellular access or congested links during after-work hours.

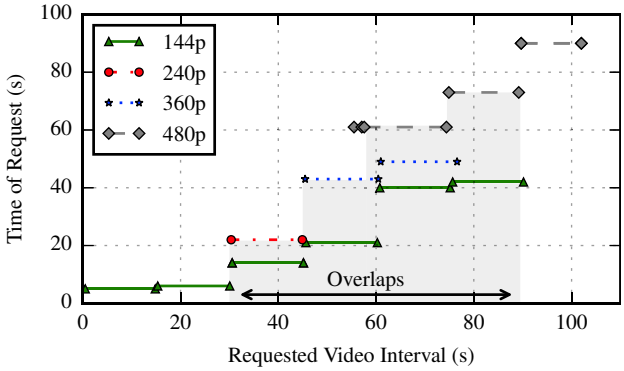

Figure 1. Example request schedule from one of the experiment runs [2]. From 30 s to 90 s overlaps can be observed where low quality (144p) is replaced by higher quality levels (240p, 360p and 480p).

In this paper we take a closer look at the behavior of the adaptation logic of YouTube. In previous work [2] we showed that YouTube's adaptation logic focuses strictly on the user, at the expense of network efficiency. In particular, we observed that the YouTube player sometimes discards its currently buffered content to re-download it in a higher quality level. In this way, the player can increase the average quality level shown to the user. However, the overall efficiency decreases as the previously downloaded segments are discarded.

Figure 1 shows a request schedule for one of the experiment runs. The x-axis shows the request video interval in playback time, e.g. the first 16 s of the video. The y-axis shows the time of request based on the experiment time, with 0 being the time the first HTTP GET request was sent

978-0-9883045-1-2/16 $31.00 © 2016 ITC

to the server. At first, the player requests one minute of the lowest quality level. Then, 20 seconds into the experiment, the player revises its previously made decision, discards two of the low quality segments (i.e. 30 s of playback time) and starts to download a higher quality level instead. The shaded areas in the figure illustrate where lower quality segments were discarded. The figure illustrates that in this video view out of 105 s of video, approximately 60 s were available in more than one quality level at the player. As we show in our previous study [2], this is not an isolated incident, but happens on a regular basis. Therefore, in this work we take a closer look at the effectiveness of this approach.

The evaluation in this work is based on an experimental data set with over 10.000 video views of about 30 different videos. The videos were played in a testbed where the connection was throttled to $\{0.4, 0.5, \ldots 3.0\,\mathrm{Mbps}\}$. A proxy was used to decrypt the HTTPS connection. The dataset and testbed is described in detail in [2], [3] and the experimental dataset is freely available online at [4]. Over 70 QoE-relevant metrics such as average quality level, cumulative stalling times and number of quality switches were collected.

Our main contribution lies in analyzing how much the used adaptation algorithm can be optimized. Even if we completely avoid stalling events, a higher mean video quality is achievable in most cases. Further, it is possible to reduce the number of resolution switches and start the video after a shorter initial delay.

The paper is structured as follows. The next section discusses the related work in this area of research. In Section III we discuss the methodology and the two approaches used in this work. In Section IV we present the results and in Section V we conclude this work and outline future work.

II. Related Work

First, we describe the related work in the area of user perception of HAS video streaming services. In [5], [6], Hoßfeld et al. conclude that avoiding stallings is the first priority when optimizing a HAS service for user experience. The second and third priority are the average video quality shown to the user and minimizing the number of switches and the amplitude of the switches. In [7], Nam et al. conduct a large scale study on YouTube and confirm the high (harmful) impact of re-bufferings and quality switches on the user's QoE. Further related work in the area of HAS QoE and on HAS in general can be found in [8].

Next we describe the related work specific to YouTube's adaptation strategy and in particular regarding observed redundant traffic. In [1], Casas et al. conclude that the ratio between video bit-rate and downlink bandwidth significantly influences YouTube's adaptation. They show that YouTube's adaptation is not robust in bottle-necked scenarios. Yao et al. show in [9] that the iOS YouTube player uses overlapping segments to smoothen the playback. Rao et al. [10] and Ito et al. [11] evaluate YouTube's traffic pattern during video playback. They show a dependency of the behavior on the viewing device. In [12], Añorga et al. show that YouTube uses a large playout buffer of 13 s to 40 s and therefore can only adapt slowly to changing bandwidth conditions. In [13], Alcock et al. describe YouTube's initial burst phase in detail. They show that 32 s of playback time in a low quality level is transferred to the client as fast as possible before the transfer is throttled. We account this for a major source of redundant traffic as the low quality level is replaced later by higher quality segments. In [14], Mansy et al. evaluate YouTube's adaptation behavior in terms of redundant traffic, playback behavior and bandwidth utilization. In a wireless scenario with one video and one bandwidth pattern they quantify the redundant traffic to 16%. They also show that the adaptation strategies of other content providers behave in a similar way. Lui et al. [15] conclude that YouTube's buffer level on mobile devices is based on the amount of data buffered, not on the amount of playback seconds. They observe redundant traffic when segments at the beginning of the video are re-downloaded and quantify the redundancy to 15%. In [16], the authors identify additional redundant traffic on the transport layer of YouTube in a mobile scenario. They quantify the redundant traffic to 35% due to frequent termination of TCP connections and in-flight packets.

III. Methodology

In this section we discuss the two approaches we use to evaluate the observed adaptation from the experimental data set. This first approach is based on regression and uses previously observed video sessions to create an estimation on how much non-redundant traffic relates to a specific average playback quality. This has the advantage of being fast, scalable and not computationally expensive. Furthermore, as it is based on actual observed data, it captures the dynamics of the deployed system. We use this estimation then to calculate the maximum achievable average quality level based on the total amount of downloaded Bytes in a playback session. The second approach is based on a mixed integer linear program (MILP) formulation. For this optimization problem we take the actual video segment sizes, the observed bandwidth and cumulative stalling times from the experimental data set as an input. This gives us the optimal adaptation considering the stalling times. In a second step, we remove the cumulative stalling times and force the optimization problem to instantly play the video.

A. Heuristic Approach

To describe the heuristic, we first have to define redundant traffic. The redundant traffic ratio is defined as in the subsequent equation, where B_T is the total amount of data downloaded during the playback session and B is the sum of the segments' sizes shown to the user.

$$\rho = \frac{B_T - B}{B} \tag{1}$$

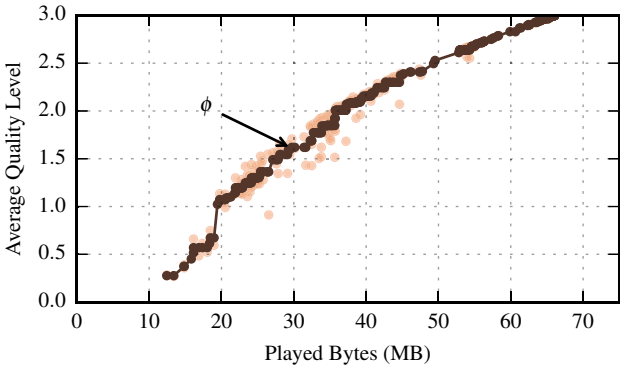

Figure 2. Isotonic regression result showing the relationship between Bytes shown to the user B and resulting average playback quality for video vbLLqaa9ksw. 336 video views are used in this regression.

The heuristic approach uses isotonic regression [17] to deduce a video-dependent relationship between the data shown to the user and the resulting average quality level based on previously recorded playback sessions. This gives an estimate of how much non-redundant data is necessary to reach a certain quality level. Furthermore, it allows us to estimate the difference in terms of average quality between two different amounts of data. The advantage of the approach is, as previously described, that it captures the dynamics of the overall system as it is based on actual observations.

Let $\phi(B)$ be the functional relationship between the quality level ϕ and the Bytes B. Figure 2 illustrates the function ϕ for one of the videos in the data-set. The x-axis gives the amount of Bytes B played back by the player. The y-axis gives the resulting average playback quality. Each (brown) dot represents one playback session. The connected (black) dots are the isotonic regression result. Multiple observations can be made from the figure. First, a specific amount of played bytes can result in different average quality levels at the end. This is due to the combinatorial problem which arises due to the different quality levels and bit-rate variations inside a quality level. Second, there is a jump at $20\,\mathrm{MB}$ from 0.7 to 1.1 average quality level of unknown origin. Third, there are outliers, e.g. at $27\,\mathrm{MB}$, where significant more data does not increase the average quality level.

Based on ϕ we determine the loss in average quality level, or *possible gain*, due to the redundant traffic as:

$$\phi(B_T) - \phi(B) \tag{2}$$

This is the difference between the average quality level we could have reached with the total Bytes downloaded in the session ($\phi(B_T)$) and the average quality level based on the Bytes shown to the user $\phi(B)$.

B. Optimal Adaptation

In order to determine how much potential there is for optimization, we formulate a MILP for this problem. The solution to the MILP will return an optimal adaptation with respect to available bandwidth, video segment sizes, and cumulative stalling times.

A given video is available in r resolutions and consists of n segments, i.e. each segment can be played in exactly one resolution. Furthermore, each segment i that is played in resolution j has a size S_{ij}. We assume that all segments have the same duration τ and are downloaded in order. The total data that has been downloaded at the point in time t is $V(t)$. Before a segment can be played, it has to be downloaded. This means there is a deadline D_i until which the segment must be downloaded to avoid stalling. Since there is an initial delay T_0 before the first segment can be played, according to [5] the deadline is

$$D_i = T_0 + i \cdot \tau. \tag{3}$$

The goal is to optimize the downloading process so that the video may be played with the highest average resolution. This leads us to a MILP which is a special case of *Optimization Problem 2* from [5].

$$\begin{aligned}
\text{maximize} \quad & \sum_{i=1}^{n}\sum_{j=1}^{r_{\max}} j x_{ij} \\
\text{subject to} \quad & \\
& x_{ij} \in \{0,1\} \\
& \sum_{j=1}^{r_{\max}} x_{ij} = 1, \qquad \forall i = 1,\ldots,n \\
& \sum_{i=1}^{k}\sum_{j=1}^{r_{\max}} S_{ij} x_{ij} \le V(D_k), \quad \forall k = 1,\ldots,n.
\end{aligned}$$

This is a Multiple-Choice Nested Knapsack Problem which is NP-hard. However, there exist polynomial time algorithms that return an approximation for the optimal solution that is sufficiently good for most practical purposes. The MILP was implemented in Gurobi [1] with MATLAB.

C. Data Sets

In total, there are four data sets used in this evaluation as listed in Table I. The three data-sets starting from the second are calculated based on the first (experimental) data set.

First, we have the initial observations which shall serve as the *baseline* in the following analysis. These measurements were originally recorded in [2] where the measurement methodology and measurement set-up is described in more detail: 35 videos $\times 27$ bandwidth values $\times 15$ replications. Four quality level representations were observed: $144p, 240p, 360p, 480p$. In the following, we refer to these video quality levels as $0, 1, 2, 3$. Please note that stalling events did occur in 56% of these runs.

Based on this data set, we used the *heuristic approach* described in III-A to estimate the average resolution that is reachable if there was no redundant traffic, i.e. when no video segment is downloaded multiple times. Please

[1] http://www.gurobi.com/

Table I
OVERVIEW OF THE DATA SETS USED IN THIS WORK.

Data Set	Identifier	Description
Measurements from [2]	measurement	The experimental data set recorded in a testbed.
Heuristic estimation	heuristic	The heuristic estimation which gives us the possible gain without redundant traffic.
Optimization with stalling	opt (prebuffering)	The MILP solition with stalling times.
Opimtization without stalling	opt (instant play)	The MILP solution without stalling times.

note that it was assumed that the same amount of stalling would occur.

As a new contribution, we use the optimization problem, described in Section III-B to exactly calculate the highest mean resolution that was optimally obtainable. As a second step, the number of switches is minimized as first proposed in [18]. For both steps, we limit the execution time of the Gurobi Optimizer to $1\,\mathrm{s}$ in order to process the complete data in a timely manner. Increasing the execution will most likely lead to slightly better values than presented in the following. For this *two-step approach*, we consider the same video files, the same duration of the viewing session and the same average network throughput as was used in the baseline scenario to make it comparable. However, instead of having stalling events interrupt the replaying process, we add an initial delay to the replaying process. The duration of this delay is equal to the sum of the observed stalling events. This leads to the same duration of the viewing session and the same replay time and the same amount of data that was totally downloaded. In the following, we refer to this scenario as *opt (prebuffering)*.

Lastly, we present a data set that is obtained in the same fashion as *opt (prebuffering)* with one major difference: the video starts to play immediately after the first segment has been downloaded. To achieve this, we consider the exact same network throughput as in the baseline scenario, while having a shorter session duration since the stalling times are omitted. This means that the amount of data that is downloaded in this case is lower than in the baseline scenario. In the following, we refer to this scenario as *opt (instant play)*.

IV. RESULTS

In this section, we present our results on how much YouTube's current adaptation algorithm could be improved. As key metrics, we analyze the average quality, the frequency of stalling events, the frequency of quality switches and the initial delay of the video playback. As a first step, we discuss how the video quality and stalling events are related to each other in the experimental data set.

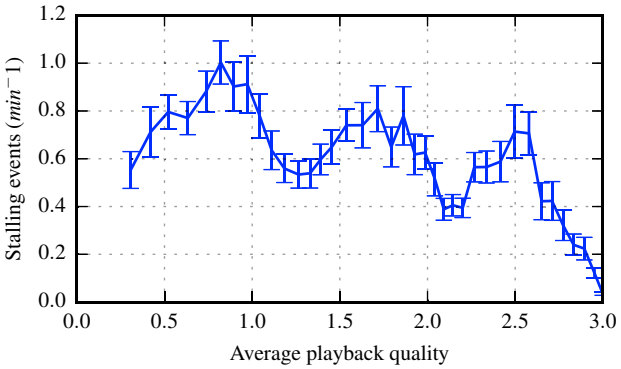

Figure 3. Average playback quality compared to stalling events per minute as observed in the experimental data set. Average playback quality is clustered using k-means with 40 bins. On average, every 0.6 minutes a stalling event can be observed. Average playback quality and stalling events are highly correlated and show osculating behavior.

A. Relationshop Between Quality and Stalling

Figure 3 illustrates the relationship between the average quality level and the stalling events in the experiment result set. For the average quality level, 0 is defined as 100% of the segments are shown to the user in $144p$. Quality 3 is defined as 100% of the segments are shown to the user in $480p$. The average quality levels are clustered using k-means (40 bins) and the error bars indicate the $95\,\%$ confidence interval of each cluster. Two observations can be made from the figure. First, the lowest average quality level is 0.3 with about 0.5 quality switches per minute. From this it follows that the player risks one stalling per two minutes in order to avoid showing only the lowest quality level in low bandwidth scenarios. Second, the buffering events exhibit an oscillating behavior. The oscillating behavior is consistent with observations made in [1], [2]. The studies show that the performance of YouTube's adaptation algorithm depends on the ratio between video bit-rate and available bandwidth. For certain ratios, the algorithm is able to efficiently use the available bandwidth, i.e. there is only a low amount of redundant traffic and buffering. Other ratios exhibit a high amount of redundant traffic and buffering ratio. In total, the pearson correlation shows a high correlation (-0.774) between average quality level and buffering events. To summarize, in the experimental data, YouTube's adaptation exhibits 0.4 to 1 stalling events per minute.

B. Optimal Adaptation

Next, we discuss the potential gain in average quality as estimated by the heuristic and the two optimization problem formulations.

Figure 4 displays the distribution of the difference between the observed mean video quality and the optimally achievable mean video quality. We observe that about 30 percent of runs are already at maximum quality and can therefore not be improved. The data set *opt (prebuffering)* leads to the highest mean quality. However, the results for the three data sets are very close to each other, e.g. the median of all three data sets is within 0.15 of a quality of

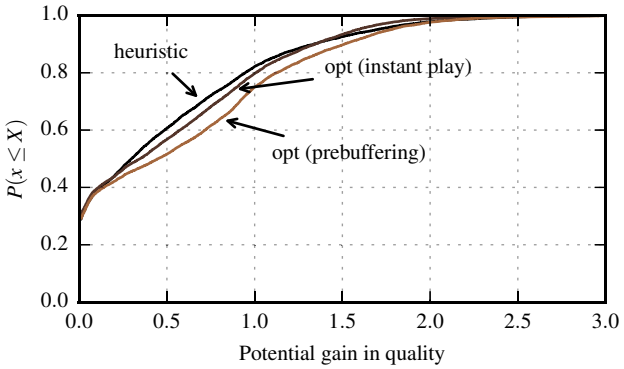

Figure 4. Distribution of the difference between the observed mean video quality and the optimally achievable mean video quality according to the optimization problem from Section III-B and the heuristic from [2].

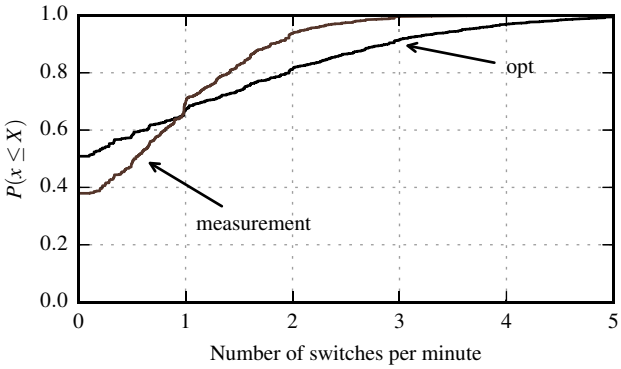

Figure 6. Distribution of the number of switches per minute for the heuristic and the optimization. Very similar results for both data sets that were created by the optimization approach.

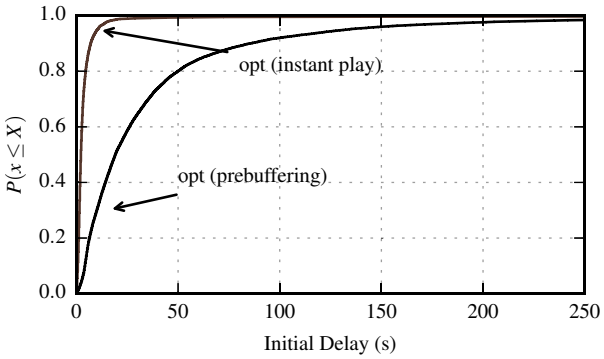

Figure 5. Initial delay for the two optimization data sets

each other.

If we take Figure 5 into consideration, it becomes clear that this minor difference in quality comes at a price: *opt (instant play)* demonstrates that it would have been possible to avoid stalling and a high initial delay in 93 percent of cases while increasing the quality in 30 percent of cases. While *opt (prebuffering)* shows that the mean quality could have been increased by adding an initial delay, the improvement is not particularly high. Lastly, while the heuristic leads to a worse result than *opt (prebuffering)*, it has the advantage of being a less complex problem. This might outweigh the slightly better performance for practical purposes.

Finally, Figure 6 shows the number of quality switches per minute. Here, both data sets that were created with the optimization approach lead to very similar results, which is why we only present the results for *opt (instant play)*. Whereas the number of switches is not of significant importance to the QoE in video streaming according to [8], continuous video quality switches lead to a low QoE [19]. The heuristic approach and the optimization both lead to less than 2 switches per minute in more than 80 percent of cases which are acceptable values. However, the two-step approach for the optimization leads to some very high switching frequencies that might be problematic. This is because the two-step approach puts very high value on the optimization of the quality level and very little emphasis on the number of switches. Luckily, this problem can easily be averted by using a slightly different approach: In [20], a method is proposed that combines both steps into one, allowing the number of switches to be emphasized higher at a negligibly low cost of quality.

V. Conclusion

YouTube is a major source of Internet traffic worldwide and it is important to understand how it uses the available resources in a network. Previous studies revealed that YouTube deploys a user-centric adaptation strategy which allows the player to discard previously downloaded segments and re-download them at a higher quality level. This increases the average playback quality for the user, but at the same time decreases the overall efficiency.

In this paper we use two methods to quantify this decrease in efficiency. The first method is a fast heuristic approach based on historical data. The second method is based on an optimization problem formulation.

Our results show there is still a lot of improvement possible for YouTube. Instead of downloading the same segment multiple times, the wasted traffic could be used to download segments in a higher quality. On average 20% of the videos could have been downloaded in a higher quality. In spite of adaptive mechanisms, stalling usually occurred once per 1 to 2 min. Assuming the future network bandwidth can be predicted, our optimization problem shows that stalling can be prevented in 94% of these cases. At the same time the initial delay can be kept below 10 s in 95% of cases. In future work, other streaming services such as Amazon Instant Video or Netflix may be investigated with the optimization approach described in this paper.

Acknowledgements

This work was partly funded by Deutsche Forschungsgemeinschaft (DFG) under grants HO 4770/1-2 (DFG Ökonet: Design and Performance Evaluation of New Mechanisms for the Future Internet – New Paradigms and Economic Aspects).

References

[1] P. Casas, A. Sackl, S. Egger, and R. Schatz, "YouTube & Facebook quality of experience in mobile broadband networks," in *Globecom Workshops (GC Wkshps), 2012 IEEE*, IEEE, 2012, pp. 1269–1274.

[2] C. Sieber, P. E. Heegaard, T. Hoßfeld, and W. Kellerer, "Sacrificing efficiency for quality of experience: YouTube's redundant traffic behavior," in *IFIP Networking 2016 Conference (Networking 2016)*, Vienna, Austria, May 2016.

[3] C. Sieber, A. Blenk, M. Hinteregger, and W. Kellerer, "The cost of aggressive HTTP adaptive streaming: Quantifying YouTube's redundant traffic," in *IFIP/IEEE International Symposium on Integrated Network Management (IM) 2015*, May 2015, pp. 1261–1267.

[4] *Experimental dataset*. [Online]. Available: https://git.io/vRSSW.

[5] T. Hoßfeld, M. Seufert, C. Sieber, T. Zinner, and P. Tran-Gia, "Identifying qoe optimal adaptation of HTTP adaptive streaming based on subjective studies," *Computer Networks*, vol. 81, pp. 320–332, 2015.

[6] T. Hossfeld, M. Seufert, C. Sieber, and T. Zinner, "Assessing effect sizes of influence factors towards a qoe model for HTTP adaptive streaming," in *2014 Sixth International Workshop on Quality of Multimedia Experience (QoMEX) 2014*, Sep. 2014, pp. 111–116.

[7] H. Nam, K.-H. Kim, and H. Schulzrinne, "Qoe matters more than qos: Why people stop watching cat videos," Apr. 2016.

[8] M. Seufert, S. Egger, M. Slanina, T. Zinner, T. Hobfeld, and P. Tran-Gia, "A survey on quality of experience of HTTP adaptive streaming," *Communications Surveys & Tutorials, IEEE*, vol. 17, no. 1, pp. 469–492, 2015.

[9] L. Yao, W. Qi, G. Lei, S. Bo, C. Songqing, and L. Yingjie, "Investigating redundant internet video streaming traffic on iOS devices: causes and solutions," *Multimedia, IEEE Transactions on*, vol. 16, no. 2, 2014, ISSN: 1520-9210.

[10] A. Rao, A. Legout, Y.-s. Lim, D. Towsley, C. Barakat, and W. Dabbous, "Network characteristics of video streaming traffic," in *Proceedings of the Seventh COnference on emerging Networking EXperiments and Technologies*, ACM, 2011.

[11] M. Ito, R. Antonello, D. Sadok, and S. Fernandes, "Network level characterization of adaptive streaming over HTTP applications," in *IEEE Symposium on Computers and Communication (ISCC)*, Jun. 2014.

[12] J. Añorga, S. Arrizabalaga, B. Sedano, M. Alonso-arce, and J. Mendizabal, "YouTube's DASH implementation analysis," in *19th International Conference on Circuits, Systems, Communications and Computers (CSCC)*, 2015, pp. 61–66, ISBN: 9781618043184.

[13] S. Alcock and R. Nelson, "Application flow control in YouTube video streams," *SIGCOMM Comput. Commun. Rev.*, vol. 41, no. 2, Apr. 2011.

[14] A. Mansy, M. Ammar, J. Chandrashekar, and A. Sheth, "Characterizing client behavior of commercial mobile video streaming services," in *Proceedings of Workshop on Mobile Video Delivery, MoViD'14*, 2014, ISBN: 9781450327077.

[15] Y. Liu, F. Li, L. Guo, B. Shen, and S. Chen, "A comparative study of android and iOS for accessing internet streaming services," in *Passive and Active Measurement*, Springer, 2013, pp. 104–114.

[16] H. Nam, B. H. Kim, D. Calin, and H. G. Schulzrinne, "Mobile video is inefficient: A traffic analysis," 2013.

[17] R. E. Barlow, D. J. Bartholomew, J. Bremner, and H. D. Brunk, *Statistical inference under order restrictions: The theory and application of isotonic regression*. Wiley New York, 1972.

[18] K. Miller, N. Corda, S. Argyropoulos, A. Raake, and A. Wolisz, "Optimal adaptation trajectories for block-request adaptive video streaming," in *Packet Video Workshop (PV), 2013 20th International*, IEEE, 2013, pp. 1–8.

[19] Y. Liu, S. Dey, D. Gillies, F. Ulupinar, and M. Luby, "User experience modeling for DASH video," in *Packet Video Workshop (PV), 2013 20th International*, IEEE, 2013, pp. 1–8.

[20] E. Liotou, T. Hoßfeld, C. Moldovan, F. Metzger, D. Tsolkas, and N. Passas, "Enriching HTTP adaptive streaming with context awareness: A tunnel case study," in *IEEE International Conference on Communications*, ICC, 2016.