Modelling Video Rate Evolution in Adaptive Bitrate Selection

Yusuf Sani [*], Andreas Mauthe[†] and Christopher Edwards[‡]
School of Computing and Communications
InfoLab2l, Lancaster University
Lancaster LA1 4WA, UK
Email: [*] *y.sani@lancaster.ac.uk,* [†]*a.mauthe@lancaster.ac.uk,* [‡]*c.edwards@lancaster.ac.uk*

Abstract—**Adaptive bitrate selection adjusts the quality of HTTP streaming video to a changing context. A number of different schemes have been proposed that use buffer state in the selection of the appropriate video rate. However, models describing the relationship between video quality levels and buffer occupancy are mostly based on heuristics, which often results in unstable and/or suboptimal quality. In this paper, we present a QoE-aware video rate evolution model based on buffer state changes. The scheme is evaluated within a real-world Internet environment, where it is shown to improve the stability of the video rate. Up to 27% gain in average video rate can be achieved compared to the baseline ABR. The average throughput utilisation at a steady-state reaches 100% in some of the investigated scenarios.**

Keywords-**HTTP Adaptive Streaming, Adaptive Bitirate Selection, Buffer modelling**

I. Introduction

Nowadays a typical video streaming service is expected to serve a variety of platforms e.g., smart phones, web browsers, TVs, etc. Each of these platforms has specific requirements with respect to transmission and video quality. Nonetheless, regardless of the access device, users want the best viewing experience possible. HTTP Adaptive Streaming (HAS) is the most successful technology so far that allows content providers to cater for the requirements of these multitude of devices and different contexts. The process through which a HAS client chooses a video rate is called *Adaptive Bitrate Selection* (ABR). The first generation of ABRs relied on throughput estimation and selected the highest video rate lower than the measured throughput. It later became clear that throughput estimation alone is not a sufficient parameter for designing efficient ABR. This is because an accurate bandwidth estimation above the HTTP layer is difficult to achieve [1]. Consequently, any video rate selection algorithm that solely depends on such a relatively inaccurate estimate results in unnecessary rebufferings [2], undesirable variability of video rates [1] and sub-optimal video quality [1].

Various attempts have been made to improve some of the identified issues of throughput-based ABRs by supplementing throughput measurements with information about the playback buffer [3]. However, in the absence of a systematic model of the relationship between buffer occupancy and available video rates, researchers resorted to heuristic according to which the playback buffer is simply divided into a number of segments separated by thresholds. The problem with this design, is that, many of these algorithms attempt to maintain the buffer at the so-called optimal level, but since the buffer size, the segmentation and the thresholds are at best conjectured, the ABRs are forced to make a number of unnecessary trade-offs.

It has been shown in [2] that by basing ABR solely on playback buffer occupancy, a client can choose the highest quality level without the fear of an increase in rebuffering. However, a continuous increase in the video rate may not always enhance the quality of experience (QoE). For example, when the video quality is relatively high, an increase in the current rate does not necessarily translate into an improvement in the user-perceived quality [4], [5].

To ensure that the video rate evolves in a way that optimises QoE, there is a need for a *rate evolution map* that captures the desirable pattern of video quality transition. This paper will concentrate on the following research question: *If we have a QoE aware model of the relationship between the playback buffer state changes and the available video rates, how much improvement in user-perceived video quality can be achieved?*

In order to answer this question, this paper first identifies the patterns of video quality changes that are known to affect QoE, and then develops a QoE-aware model of the rate map that combines all stages of video rate evolution, while at the same time incorporating an optimal number of patterns that improve user-perceived video quality. We demonstrate how the proposed model can be used in practical systems by modifying selected ABRs. Experimentation over the Internet using both wired and wireless connections, based on objective QoE metrics, shows the performance of the scheme.

The rest of the paper is structured as follows: Section II presents background and the related work; Section III discusses the QoE aware evolution trajectory and the system model; Section IV details the methodology and experimental set-up used; Section V covered result presentation and finally the paper is wrapped with conclusion in section VI.

II. Background and Related Work

HTTP adaptive streaming services usually divide a video file into a number of chunks of equal temporal size with each chunk encoded in multiple bitrates. A client progressively requests a relevant chunk. The quality of a request is based on the client's measurement of the available resources. The ABRs that use throughput estimation as their main factor are called *throughput-based* ABRs, while those that solely rely on buffer occupancy are called *buffer-based* ABRs.

Due to short-term throughput fluctuations, as a result of the TCP congestion control mechanism and the difficulty in accurately estimating throughput above the HTTP layer, throughput-based algorithms use a weighted average to smooth-out the estimated network capacity [3]. However, using historical data impedes the responsiveness of an algorithm [3]. A number of measurement studies have shown that throughput-based algorithms are unstable [6], unnecessarily rebuffer [1], request sub-optimal video rates [3], and are unfair [1].

A significant amount of research is focused on how to improve the throughput measurement for ABR. The authors of [6] propose a probe and adapt technique. The algorithm mimics the congestion control of TCP but at the application layer. It uses TCP throughput as an input only when it is an accurate indicator of the fair-share of bandwidth. In the same vein, the authors of [7] use machine learning techniques to predict the achievable throughput by using network state information.

In order to improve some of the downsides of the throughput-based services, various researchers have used the buffer level as a feedback signal to complement the throughput estimation [8]. Tian and Lui [7] went further by using the playback buffer state change as the key feedback signal. Usually, most of the existing ABRs divide the whole buffer into segments: $S_0, S_1, ... < S_n$, which are separated by thresholds: $B_1 < B_2 < ... < B_{max}$. This forced ABR designers to use heuristic in deriving almost everything regarding buffer, for instance, the maximum buffer size, the respective sizes of the buffer segments, how much buffer is required for the ramping-up period, etc. Consequently, the ABRs are forced to make a number of, perhaps, false trade-offs. An oft-cited trade-off is between video quality and the amount of rebuffering.

Huang et al. [2] propose an algorithm that completely relies on buffer occupancy for video rate selection. The model employed separates the buffering from the steady-state phase, which obviously creates a disconnected flow. Furthermore, at the start-up period — called reservoir, the lowest available video rate is downloaded, hence, there is a substantial loss in video quality. During the ramping-up period, quality was linearly incremented. However, in [9] it has been shown that the probability of buffer starvation decreases exponentially with respect to the initial buffer level. Therefore, a linear evolution of the video quality when ramping-up will unnecessarily prolong the convergence time. It is also worth noting that an uncontrolled increase in video rate may not always enhance QoE. In [4] it was demonstrated that when video quality is high, an increase in the current rate does not necessarily translate into an improvement in the user-perceived quality. Nevertheless the paper made an important observation: when buffer is used as the main factor of an ABR, the trade-off between video quality and the amount of rebuffering is unnecessary.

Earlier, Mok et al. [5] have studied the effect of video quality transition on QoE. They found that a sudden drop in video rate has a negative impact on user experience. To improve QoE, they opted to switch down the video rate to an intermediate level even when the target video level is lower. The problem with this design is that the user will be downloading higher bitrate than the download rate, hence increasing the risk of buffer starvation, especially since both the intermediate level and the maximum buffer size are heuristically determined. While this work narrowed its investigation to a pattern, in this paper we pay attention to the whole sequence of the trajectory through the space of all possible system states.

III. System Modelling and Implementation

A. Quality Evolution Trajectory

At any given time t after the video streaming started the buffer may contain an array of chunks of different quality levels. However, chunks of different video rates generally have different sizes in bytes. We shall assume that all chunks contain an equal amount of video time V in seconds. Since there is no direct mapping between buffer size in bytes and video time, we calibrate buffer in time i.e. by the second. This has also been assumed in [2], [7].

At the beginning of a streaming session ($t = 0$), a server presents to a client a set of different video rates $Q = \{q_0, q_1, q_2...q_n\}$, with $|Q| = n + 1$. Let us suppose $q_0 < q_1 < ... < q_n$, therefore q_0 is the minimum quality level (referred here also as q_{min}) and q_n is the maximum available quality level (called q_{max}). Suppose B_t is the buffer occupancy at time t and B_{max} is the maximum buffer. Let $\hat{c}_t$ denote the estimated throughput at time t with $C(t)$ being the system capacity (i.e., $\hat{c}_t \leq C_t$).

Usually, after the receipt of the media description file at t_0, playout buffer is often empty ($B_{t_0} = 0$), a client starts requesting a chunk with quality level q_{min} so as to minimise the start-up period. However, a prolonged download of q_{min} will negatively affect the user experience. Hence, the client, immediately, starts a gradual improvement of the quality of the requested chunks as soon as it receives the initial chunk. Therefore, the download of chunk $i \geqslant 1$ with video rate q_k , where $\{k \in n : \ 0 \leq k \leq n\}$, starts at time t_k^s and finishes at t_k^e using a video rate selection function $R(t)$, this continues until the last chunk is requested. Let us also assume that the rate at which the client's requested video rate evolves with respect to time $dR(t)/dt$ is $g\prime(R)$. Assuming that $C(t) \geqslant R(t) = q_{max}$, therefore $g\prime(R)$ is positive at any time after the start of streaming except when $R(t) = q_{max}$ and $B_t = 0$ in which case $g\prime(R) = 0$. To ensure that the client gets its fair share of the available bandwidth, we rely on the recommendation of [2], which states that highest rate is selected only when buffer is full or nearly full (i.e., $R(t) = q_{max}$ when $B_t \to B_{max}$).

To avoid high amplitude variations (e.g. an abrupt drop of the video quality), which are known to be detrimental to QoE [5] and to minimise the negative impact of *Recency Effect* [10], transition decision to q_{k+1} should depend on q_k. Furthermore, since users are not known to be appreciative of increase in video quality when video rate is relatively high [4] we recommend a non-linear $g\prime(R)$. However, since it is not yet clear at what point users begin to be less receptive to the quality increase, we suggest that after reaching $q_{max}/2$ a client should start reducing the rate at which it increases its video quality. The trajectory of $g\prime(R)$ that we deduced from the foregoing discussion is a concave path pinned at two points $q = 0$ and $q = q_{max}$ with the amplitude at $q_{max}/2$, this can easily be described by a quadratic function with $q = 0$ and $q = q_{max}$ and a positive constant a.

$$g\prime(R) = aq(q_{max} - q) \tag{1}$$

B. Modelling

Knowing that at time t_0 $R(t) = q_0$ and $B_{t_0} = 0$, how can we predict video quality (q_k) to be streamed at t_i, where $i \geqslant 1$, in conformance with the desired trajectory, given that $B_{t_i} = B_i$ for $\{B_i : \ B_{min} \leq B_i \leq B_{max}\}$.

1) Continuous Rate: We first look at a case where $R(t)$ results in any value between q_{min} to q_{max}. With this assumptions we can model $R(t)$ as a continuous function [1].

Clients usually infer $C(t)$ from $\hat{c}(t)$ for the purpose of rate selection. Now, suppose $c(t_i)$ is the estimated throughput when $t = t_i$ derived from the average of h number of chunks, we have

$$c(t_i) = \frac{1}{t_i - t_{i-h}} \int_{i-h}^{h} \hat{c}(t) dx.$$

Let us assume that a HAS client requests chunk i immediately after chunk $i - 1$ is completely downloaded except when the buffer is full. In this case it waits for V seconds (chunk size) before sending a request. Except during the off period, the playback buffer drains at the one buffer second every real time second and fills at $C(t)/R(t)$, therefore the rate at which buffer changes is

$$\frac{dB(t)}{dt} = \frac{C(t)}{R(t)} - 1 \tag{2}$$

In most contexts, $C(t)$ is time-varying, therefore, if the client is to avoid buffer starvation, the output of $R(t)$ has to adapt to this changing environment with time.

$$\frac{dR(t)}{dt} = \frac{dR(t)}{dB} . \frac{dB}{dt} \tag{3}$$

We want $R(t)$ to closely match $C(t)$, therefore $\frac{dB(t)}{dt} \approx 0$. From equation 1 and 3

[1]This is without loss of generality, in fact, in the next section we drop this assumption.

$$\frac{dR(t)}{dt} = \frac{dR(t)}{dB} = aq(q_{max} - q) \tag{4}$$
$$\frac{dR(t)}{q(q_{max} - q)} = adB$$

after simplification using partial fraction method and using $R(t) = q$ we have

$$\int \frac{1}{q} dq + \int \frac{1}{q_{max} - q} dq = \int aq_{max} dB \tag{5}$$

By integrating Equation 5 we have

$$\ln q - \ln |q_{max} - q| = aq_{max}B + e \tag{6}$$

Recall we start streaming with a minimum quality level, therefore $q = q_{min}$ and $B = B_{t_0}$. Using this information to evaluate e.

$$e = \ln \frac{q_{min}}{q_{max} - q_{min}} - aq_{max}B_{t_0} \tag{7}$$

Substituting equation 7 into 6 and simplifying we have

$$\ln \frac{q}{q_{max} - q} - \frac{q_{min}}{q_{max} - q_{min}} = aq_{max}(B_t - B_{t_0}) \tag{8}$$

finally solving for q and ($B_t - B_{t_0} \approx B_t$), since $\{B_{t_0} : 0 < B_{t_0} \leq V\}$

$$R(t) = \frac{q_{max}}{1 + [\frac{q_{max}}{q_0} - 1] \mathrm{e}^{-aq_{max}B_t}} \tag{9}$$

$$\lim_{B \to \infty} R(t) = q_{max} \tag{10}$$

From Equation 10 it can be deduced that the limiting factor of $R(t)$ is q_{max}, in other words the maximum value q can reach assuming an infinite buffer size is q_{max}. This means after reaching q_{max} any increase in the buffer size does not result in a commensurate rise in q. Therefore, at this point we have the B_{max}.

2) Discrete Rate : By dropping our assumption about the continuous nature of video rates, the video quality has to be chosen from a finite discrete set. Furthermore, q can only move from one valid value to another. We assume the quality level change is done only between adjacent video rates, that is, q_k can only move either to q_{k-1} or q_{k+1}.

The model is now modified to reflect this. To change a video rate a buffer must have grown or contracted by a certain buffer distance. Precisely, to change quality level we need $\Delta B_k = R^{-1}(q_{k+1}) - R^{-1}(q_k)$. When ΔB_k is positive the quality level is going to be increased and when it is negative the quality level going to be reduced.

C. Implementation

As a proof of concept, we apply the proposed model within the following two selected rate adaptation algorithms: the buffer-based algorithm proposed by Huang et al. [2], and the throughput-based ABR by Miller et al. [8].

When modifying the implementation of the algorithm proposed in [2], we do not change anything except that we drop the *reservoir*. Hence from the start, the algorithm now

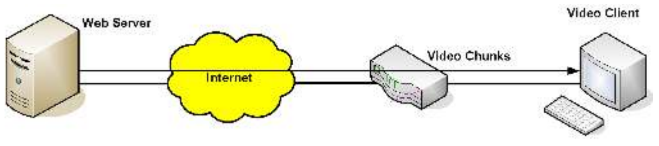


Figure 1. Experimental Set-up

relies on our model (for a detail discussion of the algorithms see [2]).

To retrofit the model into [8], we had to slightly modify the algorithm. Though none of the changes affect the throughput related logic. In order to closely map the original buffer dynamics, we divide the playback buffer into three phases. The first phase is when video rate change is slow, with a threshold at B_{qt_1}. The next phase is when the video rate grows exponentially, which ends at B_{qt_2}. The third is when video quality level increase reaches saturation, which starts at B_{max}. The threshold can be calculated thus:

$$B_{qt_x} = R^{-1}(q_{min} + \beta(q_{max} - q_{min}))$$

For $x = 1$ the $\beta = 0.1$ and for $x = 2$ the $\beta = 0.73$.

IV. Performance Evaluation

This section presents the experimental set-up and the performance evaluation metrics

A. Experimental Set-up

The test-bed set-up is shown in Fig. 1. The client is connected to the Internet either via an Ethernet switch or EE's 3G networks. The web server is located at the Alpen-Adria-Universität Klagenfurt, which hosts the Big Buck Bunny dataset [11].

All the players used are implemented in Python, and run on top of Ubuntu 12.04.2 LTS. The host that runs the players also hosts: Dummynet, tcpdump, lsof, and Wget.

Throughout the wireline experimentation, we limit the maximum downstream available bandwidth to $6mbps$. While for the wireless, we conduct a "blue-sky" test. For all players, we set $B_{max} = 240s$, and for the player running the Huang et al. [2] original algorithm, we set its reservoir to $40s$. The player using the original throughput-based algorithm retain the same configurations as in [8]. We found through experiments the values between 0.05 to 0.1 are appropriate for the growth constant a of the proposed model, therefore, we use $a = 0.05$ throughout. Each experiment was conducted 10 times and the average result is used.

B. Evaluation Metrics

Research in the field of QoE is still ongoing, and no definite model has so far been established. However, the following objective QoE metrics are used to evaluate the model.

- Rebuffers: this is the total number of video freeze per streaming session.
- Average video rate: is calculated as $\frac{t_1q_1+t_2q_2...t_nq_n}{t_n-t_1}$ and measured in kb/s

Table I
Adaptation for Variable Bandwidth

Players	Maximum Video rate (kb/s)	Average Video rate(kb/s)	Throughput Utilisation (%)
Original[2]	4000	2982	67
Modified[2]	6000	3827	100
Original[8]	4000	2212	67
Modified[8]	5000	2645	85

- Instability: is the fraction of successive chunk requests by a player in which the requested video rate changes [12], measured at the steady-state.
- Utilisation of available network resource: is calculated by dividing of average video rate by the average network capacity [7].
- Convergence time: is the time taken to settle at the sustainable video rate.

V. Results

This section discusses the result of the various test-bed experiments conducted in both wired and wireless environments.

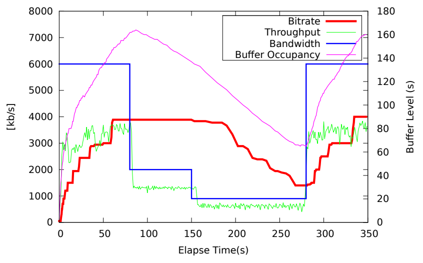
A. Client with Wireline Access

1) Variable Bandwidth: these experiments are aimed at demonstrating the elasticity of the proposed model, i.e. how it adapts to a rapidly changing bandwidth. As can be seen from Fig. 2, the streaming started with a maximum available bandwidth of $6mb/s$, after $80s$ the bandwidth dropped to $2mb/s$ at $150s$ it dropped again to $900kb/s$, finally at $270s$ it rose back to $6mb/s$ and stayed until the end.

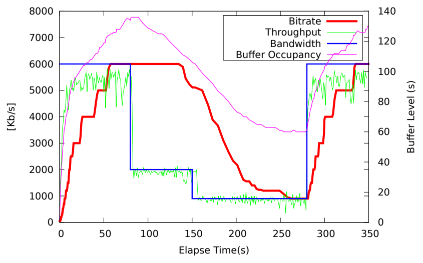
The first thing to note is that the video rate of segments downloaded by the player employing the proposed model (Fig. 2(b) and 2(d)), are able to converged at a relatively higher video rate, in fact, for the modified buffer-based player, it converged at exactly the system capacity (see Fig. 2(b)). Table I shows that the modified players achieved a maximum video rate of $6mb/s$ and $5mb/s$ against the $4mb/s$ for the unmodified players. This translated to 100% and 85% throughput utilisation, which is an improvement of 33% and 18% utilisation for the original buffer-based and throughput-based players respectively. Furthermore, we recorded an improvement in the video rate of $854kb/s$ in the case of the buffer-based player and $433kb/s$ in the case of the throughput-base player.

2) Video Rate Convergence: we investigated two scenarios: when bandwidth suddenly rises and when it instantly drops and both cases stay there for the remaining duration of the streaming session. Fig. 3 shows the former, while Fig. 4 presents the later scenarios.

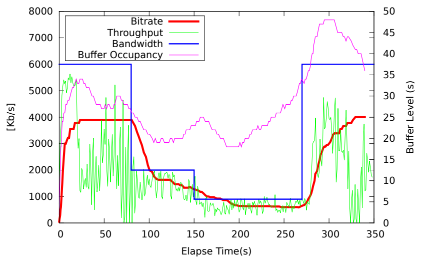
Fig. 3(a) shows when the bandwidth suddenly increases at $120s$ both players running the buffer-based algorithm are able to converge at the right quality level, albeit at different times. While it only took the player using the proposed model $65s$ to reach the convergence state, it took the original player three times longer (i.e. $165s$). Furthermore, from Fig. 3(b) it can be seen that by using the proposed model we can



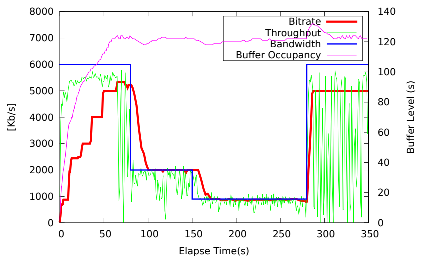
(a) Original [2]



(b) Modified [2]



(c) Original [8]



(d) Modified [8]

Figure 2. Video quality change, for both the original and the modified algorithms, operated in an environment with changing bandwidth.

reduce the convergence time by up to $80s$ in comparison to the original throughput-based player.

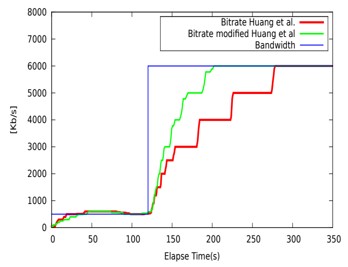
However, a player needs not to always converge to a high video quality level. It can as well converge to a lower quality level. Fig. 4 presents such a scenario. When the bandwidth suddenly drops, it takes the player using the original buffer-based logic longer to converge even though it is coming from a lot lower quality level (see Fig. 4(a)). That is $102s$ against $146s$ for the unmodified buffer-based algorithm.

Furthermore, Fig. 4(b) shows when the bandwidth suddenly drops, the player running the unmodified throughput-based algorithm was so aggressive in its reduction of the video rate that the player had to reach the lowest available video quality before it later stabilises at a sub-optimal rate. Such a large amplitude in video switch in detrimental to QoE. However, the player running the proposed model was much more conservative in its reduction and was able to converge at the appropriate quality level.

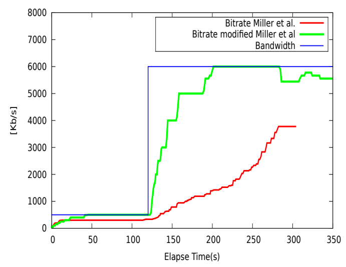
B. Client with Wireless Access

As can be seen from Fig. 5 the throughput of the wireless network is highly fluctuating, which makes it difficult to ascertain the actual capacity of the link. Therefore, we leave out any test on utilisation. From Fig. 5(a) and 5(c) the maximum quality level attained by the original players are $600kb/s$ and $700kb/s$ respectively. However, the modified versions of the players are able to achieve $1500kb/s$ each (see Fig. 5(b) and 5(d)). Importantly, this helps the modified players achieve 45% increase in the average video quality. The summary results are presented in Table II.

Finally, a stability test for the players is conducted (see Table II). Both the original throughput-based and buffer-based players suffer a high degree of instability, at the steady-state the players are respectively 12.6% and 11.8% unstable. However, the instability is significantly reduce,



(a) Original [2] Vs Modified [2]



(b) Original [8] Vs Modified [8]

Figure 3. Video quality convergence, for both the original and the modified algorithms, when bandwidth increases.

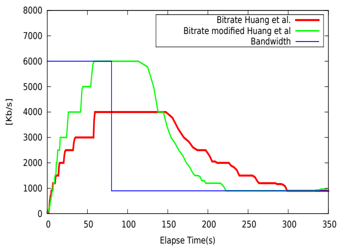
Table II
ADAPTATION IN WIRELESS ENVIROMENT

Players	Maximum Video rate (kb/s)	Average Video rate(kb/s)	stability (%)
Original[2]	600	567	12.6
Modified[2]	1500	1247	2.6
Original[8]	700	536	11.8
Modified[8]	1500	1239	4.0

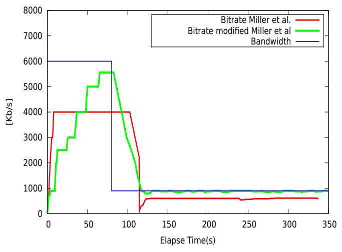
when the proposed model is used, to 2.6% for the buffer-based player and 4.0% for the throughput player.

VI. Conclusion

The task of an adaptive bitrate selection module of HTTP adaptive streaming is to ensure that the quality of video



(a) Original [2] Vs Modified [2]



(b) Original [8] Vs Modified [8]

Figure 4. Video quality convergence, for both the original and the modified algorithms, when bandwidth.

is appropriate to its context. Selecting which resource is used as a situational indicator, in video rate adaptation, is context dependant, however, it is difficult to build an ABR that maximises QoE without the knowledge of buffer state. Relying on heuristic for such an important aspect of ABR, as is currently the practise, is not the best option. In this paper, we propose a QoE-aware model of the relationship between video quality and buffer state changes. The model is able to find the optimal buffer requirement for any given set of video quality levels. The scheme is evaluated within a real-world Internet environment and the result is encouraging. In future we plan to conduct more tests and incorporate more factors into the model, for example power level.

References

[1] T.-Y. Huang, N. Handigol, B. Heller, N. McKeown, and R. Johari, "Confused, timid, and unstable: Picking a video streaming rate is hard," in *Proc of the ACM IMC*, 2012, pp. 225–238.

[2] T.-Y. Huang, R. Johari, and N. McKeown, "Downton abbey without the hiccups: Buffer-based rate adaptation for http video streaming," in *Proc. of the FhMN*, 2013, pp. 9–14.

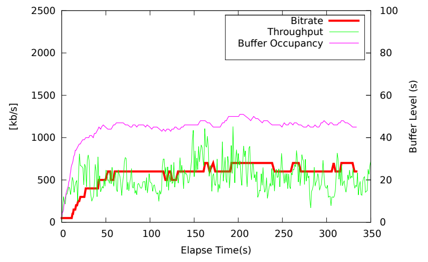
[3] S. Akhshabi, L. Anantakrishnan, A. C. Begen, and C. Dovrolis, "What happens when http adaptive streaming players compete for bandwidth?" in *Proc. of the NOSSDAV*, 2012, pp. 9–14.

[4] N. Cranley, P. Perry, and L. Murphy, "User perception of adapting video quality," *International Journal of Human-Computer Studies*, vol. 64, no. 8, pp. 637–647, 2006.

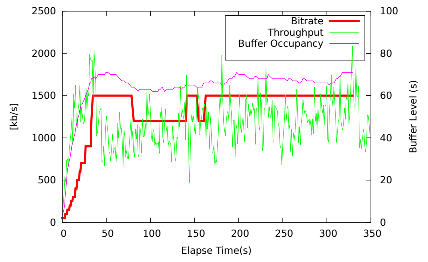
[5] R. K. Mok, X. Luo, E. W. Chan, and R. K. Chang, "Qdash: a qoe-aware dash system," in *Proc. of the MMsys*, 2012, pp. 11–22.

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

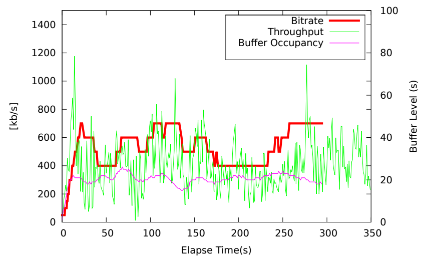
[7] G. Tian and Y. Liu, "Towards agile and smooth video adaptation in dynamic http streaming," in *Proc. of the CoNEXT*, 2012, pp. 109–120.



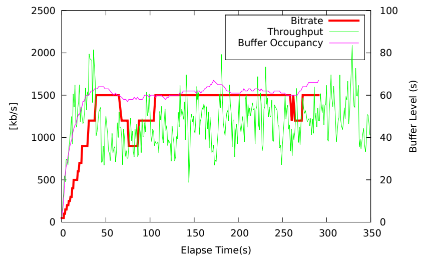
(a) Original [2]



(b) Modified [2]



(c) Original [8]



(d) Modified [8]

Figure 5. Video quality change, for both the original and the modified algorithms, operated in a wireless environment.

[8] K. Miller, E. Quacchio, G. Gennari, and A. Wolisz, "Adaptation algorithm for adaptive streaming over http," in *Proc. of the PV*, 2012, pp. 173–178.

[9] Y. Xu, Y. Zhou, and D.-M. Chiu, "Analytical qoe models for bit-rate switching in dynamic adaptive streaming systems," *Mobile Computing, IEEE Transactions on*, vol. 13, no. 12, pp. 2734–2748, Dec 2014.

[10] T. Hossfeld, M. Seufert, C. Sieber, and T. Zinner, "Assessing effect sizes of influence factors towards a qoe model for http adaptive streaming," in *Proc. of the QoMEX*, 2014, pp. 111–116.

[11] S. Lederer, C. Muller, and C. Timmerer, "Dynamic adaptive streaming over http dataset," in *Proc. of the MMsys*, 2012, pp. 89–94.

[12] S. Akhshabi, L. Anantakrishnan, C. Dovrolis, and A. C. Begen, "Server-based traffic shaping for stabilizing oscillating adaptive streaming players," in *NOSSDAV*, 2013, pp. 19–24.