On the Bitrate Adaptation of Shared Media Experience Services

Argyrios G. Tasiopoulos
University College London
argyrios.tasiopoulos@ucl.ac.uk

Ray Atarashi
IIJ-Innovation Institute
ray@iijlab.net

Ioannis Psaras
University College London
i.psaras@ucl.ac.uk

George Pavlou
University College London
g.pavlou@ucl.ac.uk

ABSTRACT

In Shared Media Experience Services (SMESs), a group of people is interested in streaming consumption in a synchronised way, like in the case of cloud gaming, live streaming, and interactive social applications. However, group synchronisation comes at the expense of other Quality of Experience (QoE) factors due to both the dynamic and diverse network conditions that each group member experiences. Someone might wonder if there is a way to keep a group synchronised while maintaining the highest possible QoE for each one of its members. In this work, at first we create a Quality Assessment Framework capable of evaluating different SMESs improvement approaches with respect to traditional metrics like media bitrate quality, playback disruption, and end user desynchronisation. Secondly, we focus on the bitrate adaptation for improving the QoE of SMESs, as an incrementally deployable end user triggered approach, and we formulate the problem in the context of Adaptive Real Time Dynamic Programming (ARTDP). Finally, we develop and apply a simple QoE aware bitrate adaptation mechanism that we compare against youtube live-streaming traces to find that it improves the youtube performance by more than 30%.

CCS CONCEPTS

• **Information systems** → **Multimedia streaming**; • **Software and its engineering** → **Synchronization**; • **Networks** → *Network control algorithms*; *Network performance modeling*; • **Theory of computation** → *Online learning algorithms*;

KEYWORDS

Shared Media Experience Services (SMESs), QoE Assessment Framework, Bitrate Adaptation

ACM Reference format:
Argyrios G. Tasiopoulos, Ray Atarashi, Ioannis Psaras, and George Pavlou. 2017. On the Bitrate Adaptation of Shared Media Experience Services. In *Proceedings of Internet QoE '17, Los Angeles, CA, USA, August 21, 2017,* 6 pages.
https://doi.org/10.1145/3098603.3098608

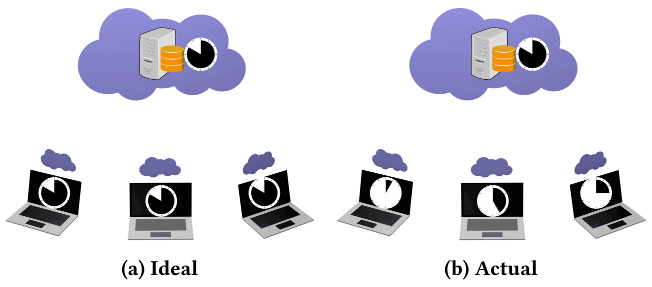

Figure 1: Shared Media Streaming Services Synchronisation, Ideal vs. Actual

1 INTRODUCTION

The recent growth of services requiring the simultaneous media consumption by a group of users has been unprecedented. An increasing number of interactive streaming, video calls, as well as cloud gaming, indicate a movement in video streaming consumption from a passive activity, that takes place in isolation, to both a dynamic and interactive scenery. These developments demand the revision of media delivery approaches since under this paradigm the end user Quality of Experience (QoE) is affected by how synchronised the media reception is between the other users. We refer to these services as Shared Media Experience Services (SMESs). In SMESs, centralised designs [10] should take into account the diverse network conditions, experienced by each end user, for delivering media simultaneously; however, the involved overhead questions the scalability of such a system. Clearly as a first step, SMESs delivery must investigate decentralised, scalable, and incrementally deployable methods to address the problem of synchronised media delivery. Here, we argue that the *end user media quality adaptation* seems like the most promising approach.

End user media quality adaptation refers to the dynamic, user triggered media chunk quality change, in terms of bitrate [12]. Utilising different bitrate qualities is an old concept [6, 8] which tries to capture the fundamental tradeoff between media quality and smooth playback. This tradeoff exists since higher bitrates increase the chances of playback freezing/disruption, caused by the volatile network conditions which inherently impose restrictions on the smoothness of content delivery.

The aim of this work is to take full advantage of the end users' media quality adaptation, in terms of bitrate, for improving their content delivery QoE in the context of SMESs. We argue that the approach under investigation is an attractive, incrementally deployable, and scalable solution, that is based on a previously proposed mature concept. SMESs can be considered as an extreme case of multimedia streaming where *i*) the synchronisation dimension of media consumption is crucial to QoE, and *ii*) the content might be *almost* real time generated. This means that always selecting a *low* bitrate does not maximise the buffered content, since the content may not have been generated yet, forcing the end-user to remain idle, wasting bandwidth resources. On the other hand, a higher bitrate, than the one that can be supported by the network conditions, does not only cause playback disruptions but also leads to end users' desynchronisation. For these reasons, the globally established DASH approach [14] is not fit for purpose.

Clearly, at first we need a notion of *what* is considered to be a successful bitrate adaptation mechanism before developing one. To this end, the main contributions of this paper are as follows:

(1) We create a QoE assessment framework for SMESs that can be used for comparing different SMESs content delivery mechanisms.
(2) We develop a simple QoE aware bitrate adaptation mechanism after formulating the bitrate adaptation problem as a real-time adaptive dynamic programming.
(3) We collect Youtube live streaming traces and we compare the performance of our adaptive bitrate mechanism against youtube.

2 SYSTEM MODEL & QUALITY ASSESSMENT FRAMEWORK

2.1 System Model

We consider a SMES environment where a group of users, G, is interested in consuming, in a synchronised way, content that is produced and/or stored at a source, *src*. Let t_0 be the moment that the session between group G members and the source *src* is established. Then for each moment $t > t_0$ we denote the set of available chunks at the source as $X^t \triangleq \{x_1, x_2, ..., x_q\}$. Evidently, when the content is created dynamically, like in the case of live streaming, X^t is increasing over time, or more formally, $X^t \supseteq X^{t'}$ for $t \geq t'$; otherwise, all the chunks of the content are available at the source since t_0, *i.e.,* $X^{t_0} = X^t$ for all $t \geq t_0$.

Each chunk has a predefined, by the source, playback duration, of ψ time units, while it is offered in $z + 1$ different bitrate qualities, $\mathcal{B} \triangleq \{b_0, b_1, ..., b_z\}$. Note that set $\mathcal{B}$ is totally ordered in the sense that $b_j \geq b_i \; \forall j \geq i, j \leq z$. Thus, the amount of data associated with a chunk of duration ψ and quality b_i is $s_i = \psi \times b_i$. Furthermore, we assume that there is a different bitrate available in $\mathcal{B}$ for each possible combination of frames per second (fps) and user display resolution. Therefore, in order to take into account the end-user diverse bitrate requirements, we assume that each user $g \in G$, is interested in a bitrate subset $\mathcal{B}_g \subseteq \mathcal{B}$ suitable for her display resolution. We consider $\mathcal{B}_g$ as a totally ordered set too.

The download speed of user g from source *src* over time, $t \geq t_0$, is denoted by $C_g(\cdot)$, which is a continuous positive function defined over $[t_0, +\infty)$. Based on $C_g(\cdot)$ function, the download completion time of a b_i bitrate k-th chunk, x_k, requested at t_k, is estimated by:

$$\tau_{t_k}^{b_i} = \{t' \in [t_k, +\infty) : \int_{t_k}^{t'} C_g(x)dx = s_i\} \tag{1}$$

where s_i is the requested chunk size. Note, that upon the time of request, t_k, chunk x_k has to belong to the set of available chunks, $x_k \in X^{t_k}$. Otherwise, t_k equals the minimum time that the chunk becomes available, t_k^{gen}, or more formally $t_k^{gen} = \arg\min_t \{x_k \in X^t\}$. Each end-user requests chunks in the logical order of media consumption, meaning that $t_k \leq t_{k+1}$.

2.2 A Quality Assessment Framework for SMESs

Let $\mathcal{T}_m \triangleq \{\tau_{t_1}, \tau_{t_2}, ..., \tau_{t_m}\}$ be the vector of delivery times for the first m requested chunks of a user. Next we describe a quality assessment framework, based on $\mathcal{T}_m$, capable of capturing the QoE over time of an end-user. Generally, quantifying the QoE of adaptive bitrate is the subject of ongoing research [1]. However, recent works have identified playback disruption, when the media player buffer gets empty, and frequent video quality changes, as major causes of impairing user's engagement [4, 11]. In the case of SMESs, we include the media consumption desynchronisation on top of the traditional adaptive bitrate QoE factors. Therefore, we express the produced QoE as a function of the received bitrate quality, playback disruptions, and *src* desynchronisation as we explain next in detail. Since QoE is subjective, the presented framework maps measurement metrics, traditionally used for describing the Quality of Service, to the QoE in a representative way, similarly to the framework described in [15, 16] in the context of crowdsourcing media delivery.

CONJECTURE 1. *The total dissatisfaction produced upon receiving the first m chunks of content,* $\widetilde{\mathcal{V}}^m(\mathcal{T}_m)$, *is the sum function of the bitrate quality cost,* $\mathcal{V}_b^m(\cdot)$, *the playback disruptions cost,* $\mathcal{V}_d^m(\mathcal{T}_m)$, *and the desynchronisation cost to the source,* $\mathcal{V}_\delta^m(\mathcal{T}_m)$.

We assume that the total cost is the sum of these different cost functions since an end user most probably would complain about these features separately. In other words, we assume that the dissatisfaction caused by a low bitrate quality is independent of how smooth the content playback is as well as of how desynchronised the user might be compared to the source.

We start by defining the cost impact of different bitrate choices function to the first m delivered chunks. Let $\mathcal{V}_b : \mathcal{B}_g \longmapsto [\underline{v}_b, \overline{v}_b]$ be a non-increasing per chunk cost function from the totally ordered set of available bitrates, $\mathcal{B}_g$, to the cost interval of $[\underline{v}_b, \overline{v}_b]$. The reasoning is that the bitrate cost of a chunk starts from a maximum cost $\overline{v}_b$ for the lowest available bitrate b_0, $\mathcal{V}_b(b_0) = \overline{v}_b$, and continues declining as the bitrate increases up to a satisfactory bitrate quality, corresponding to the minimum cost of $\underline{v}_b$. Therefore, the total bitrate cost produced can be expressed as:

$$\mathcal{V}_b^m = \sum_{k=1}^{m} \mathcal{V}_b(b^k) \tag{2}$$

Where b^k is the vector of the first k bitrates, $b^k \triangleq (b_i : i = 1, ..., k)$. Cost function $\mathcal{V}_b(\cdot)$ requires vector b^k instead of just the k-th chunk's bitrate, in order to capture the cost of bitrate changes frequency with respect to the previous choices. The lowest possible bitrate we consider is the $b_0 = 0$, denoting the case where a user prefers to skip a chunk rather than download it, known as aggressive content playback adjustment [9].

Next, regarding the aspects of playback disruption, we denote by β_k the buffered content, in terms of playback time units, of an end user upon receiving chunk k. Assuming that a chunk becomes available at the user media player as soon as it is downloaded, the buffered time upon receiving the first chunk equals the chunk's playback duration, $\beta_1 = \psi$. Therefore the smooth playback at the end user will be disrupted if the time required for downloading the second chunk exceeds the buffered content, $\tau_{t_2} - \tau_{t_1} \geq \psi$; otherwise, the buffered time will be increased by $\psi - (\tau_{t_2} - \tau_{t_1})$. In the general case, for $\Delta\tau_k = \tau_{t_k} - \tau_{t_{k-1}}$, the buffered content is updated upon the reception of chunk k according to:

$$\beta_k = \begin{cases} \psi & \text{if } \Delta\tau_k \geq \beta_{k-1} \\ \psi + \beta_{k-1} - \Delta\tau_k & \text{otherwise} \end{cases}$$

The derivation of the buffered time is associated with the playback disruption time caused upon receiving the k-th chunk, d_k, as follows:

$$d_k = \begin{cases} \Delta\tau_k - \beta_{k-1} & \text{if } \Delta\tau_k \geq \beta_{k-1} \\ 0 & \text{otherwise} \end{cases}$$

Let $\mathcal{V}_d : \mathbb{R}_+ \longmapsto \mathbb{R}_+$ be a non-decreasing per chunk playback disruption, $d_k \in \mathbb{R}_+$, cost function. Then, given a delivery time vector $\mathcal{T}_m$, the total playback disruption cost will be:

$$\mathcal{V}_d^m(\mathcal{T}_m) = \sum_{k=1}^{m} \mathcal{V}_d(d_k) \tag{3}$$

Note that for a bitrate $i_k = b_0 = 0$ we have that $\Delta\tau_k = 0$, meaning that in our quality assessment framework, skipping a chunk is not considered as causing a playback disruption since it does not *freeze* the media player of the end user.

Finally, in the context of SMESs the rate of content consumption is dictated by the source, *src*, in order to keep the end-users synchronised. Let τ_k^* be the indicative time of downloading chunk k, then the desynchronisation caused by the k-th chunk is $\delta_k = \tau_{t_k} - \tau_k^*$. Hence, $\mathcal{V}_\delta : \mathbb{R}_+ \longmapsto \mathbb{R}_+$ is defined as a non-decreasing per chunk desynchronisation cost function, meaning that the cumulative desynchronisation cost can be expressed as a function of $\mathcal{T}_m$ as:

$$\mathcal{V}_\delta^m(\mathcal{T}_m) = \sum_{k=1}^{m} \mathcal{V}_\delta(\delta_k) \tag{4}$$

Therefore, the total dissatisfaction can be expressed function to $\mathcal{T}_m$ as:

$$\widetilde{\mathcal{V}}^m(\mathcal{T}_m) = \sum_{k=1}^{m} \Big(\mathcal{V}_b(b^k) + \mathcal{V}_d(d_k) + \mathcal{V}_\delta(\delta_k) \Big) \tag{5}$$

Hence, we can estimate the playback disruption as well as the user desynchronisation over time, resulting in estimating the total dissatisfaction over time instead of per chunk reception basis.

Next, let $\tilde{C}_g(\cdot)$ be the over time downloading speed function of user g for chunk x_m when the user requests chunks x_m and x_{m+1} at the same time, *i.e.,* $t_m = t_{m+1} \leq \tau_{t_m}$, as opposed to function $C_g(\cdot)$ when only chunk x_m is requested. Then from Eq. (1) we see that the download completion time of chunk x_m is expected to be increased, *i.e.,* $\tau_{t_m} \leq \tilde{\tau}_{t_m}$, due to the download speed limitations, *i.e.,* $\tilde{C}_g(t) \leq C_g(t)\ \forall t \in [t_m, \tau_{t_m}]$. A higher completion time might result in playback disruptions as well as desynchronisation, affecting permanently the total dissatisfaction $\widetilde{\mathcal{V}}^m(\cdot)$, despite the fact that the download completion time of the second chunk might be the

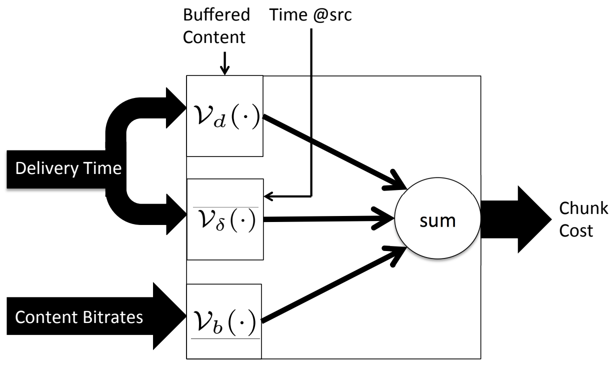

Figure 2: Quality Assessment Framework, Chunk Cost Abstract View

same in both cases, $\tau_{t_{m+1}} = \tilde{\tau}_{t_{m+1}}$. Clearly, the situation worsens when more than 2 chunks are requested at once. Therefore, we assume that *in the context of SMESs each end user requests one chunk at a time.*

Given that each end user downloads one chunk at a time, we have that the request time of a chunk x_{m+1} is taking place after the download completion of chunk x_m, *i.e.,* $\tau_{t_m} \leq t_{m+1}$. Then, let τ be the download time of chunk x_{m+1} when the chunk is requested upon the delivery of chunk x_m, τ_{t_m}. Then from Eq. (1) we have that $\tau \leq \tau_{t_{m+1}}$ for any chunk x_{m+1} request time $t_{m+1} > \tau_{t_m}$, meaning that we may end up with a higher dissatisfaction $\widetilde{\mathcal{V}}^{m+1}(\cdot)$. Since the optimal request time is determined by the previous chunk delivery time, the only decision left for the end user is the requested bitrate of each chunk. Therefore, we assume that *the total dissatisfaction produced depends solely on the bitrate choices of the end user.*

Note that despite the fact that SMESs purpose is to keep a group of users synchronised, in the adaptive bitrate approach presented here the group remains synchronised indirectly by remaining synchronised to the source of the content.

Fig. 2 summarises the per chunk cost of the quality assessment framework, where the content bitrates received so far affect the chunk bitrate cost while the chunk delivery time defines both the playback disruption and desynchronisation cost. The presented framework's purpose is to be utilised as a totally parametrised comparison basis of different approaches concerning SMESs QoE improvement. Furthermore, the derivation of QoE cost function is an independent component of the framework that can be replaced by more sophisticated and service specific models, like [5] for Cloud gaming, and [17] for interactive multimedia environments. Nevertheless, we believe that with the right parametrisation, even a simple approach like the one presented here, can capture a broad scope of SMESs QoE on a satisfying level.

3 PROBLEM FORMULATION & QUALITY AWARE ADAPTIVE BITRATE

3.1 Problem Formulation

Given that a user is interested in the first N chunks of SMESs, the objective of a control mechanism is to find the optimal vector of delivery times, $\mathcal{T}_N^*$, that minimises the total dissatisfaction $\widetilde{\mathcal{V}}^N(\cdot)$.

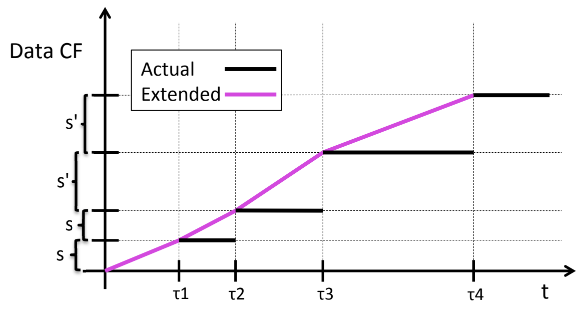

Figure 3: Buffer Monitoring, Actual and Extended Cumulative Function.

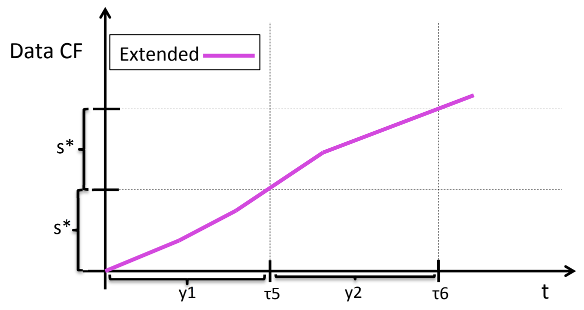

Figure 4: Delivery time measurements derivation for chunk size s^*.

The search for the optimal $\mathcal{T}_N^*$ can be formulated as an Adaptive Real Time Dynamic Programming (ARTDP) [2] problem as we describe next.

Imagine the following dynamic discrete dynamic system, resulting in the downloading time of chunk x_{m+1}:

$$\tau^{i_{m+1}} = f(\tau^{i_m}, i_{m+1}, w_m) \quad (6)$$

where the system state $\tau^{i_{m+1}}$ derives from the optimal request time of x_{m+1}, the previous state τ^{i_m}, the control parameter of bitrate for this chunk $i_{m+1} \in \mathcal{B}_g$, and the random parameter of network conditions, w_m, that affect the chunk download time, $\tau^{i_{m+1}}$. Note that if the $m+1$ chunk is not created yet upon its optimal request time, $t^{gen}_{m+1} > \tau^{i_m}$, the end user remains idle until t^{gen}_{m+1}.

If we knew the network conditions function $C_g(\cdot)$ we would be able to use Eq. (1) for defining the outcome of each possible bitrate chunk choice, making our problem deterministic. Unfortunately, $C_g(\cdot)$ is unknown, rendering our dynamic system a Markovian Decision Problem with incomplete information. That said, our problem is *adaptive* in a sense that it tries to approximate the behaviour of the unknown $C_g(\cdot)$ as a random variable, w_m, and *real time* since our knowledge about w_m increases as the system is controlled, meaning that we need an online/real time decision update method.

3.2 Quality Aware Adaptive Bitrate

Assuming that we know the time required for receiving a chunk of a specific bitrate, the involved chunk cost can be estimated. In order to improve the end user QoE it is crucial to have a mechanism for predicting each chunk's delivery time for each available bitrate choice. Here we derive a *simple to implement* delivery time forecasting approach for each chunk, that is solely based on monitoring the involved media player buffer. The challenge here is to estimate the next chunk's delivery time by using only the information of the chunks that are delivered to the media player buffer over time.

In particular, assuming that we can keep track of the chunks size, *i.e.,* in megabytes, and delivery times at the media player buffer, we form the cumulative function of data up to a monitoring time t. At first we convert the actual stepwise data cumulative function into a continue extended one, by increasing the received data linearly between the chunk delivery times. Then, given that we are interested in a specific bitrate b, associated with a chunk size s, we use the extended cumulative function to estimate the time that would be required to deliver chunks of size s up to the present time t.

For example, in Fig. 3 we see the relationship between the actual measured and the extended data cumulative functions. In particular, the player buffer receives 2 chunks of size s at τ_1 and τ_2, and another 2 chunks of size $s' > s$, *i.e.,* of a higher bitrate, at τ_3 and τ_4. Then in Fig. 4, by targeting a chunk size s^*, we use the extended cumulative function to find that y_1 seconds would be required for delivering the first chunk and another y_2 seconds for the second. In that way, we create artificial historical data regarding a sequence of delivery times for each bitrate, by using only the same extended cumulative function.

More formally, let $y_t^b \triangleq (y_{t,i}^b : i = 1, ..., k)$ be the vector of past delivery times for chunks of bitrate b given the extended data cumulative functions at time t. Let $\bar{y}_t^b(h)$ be the average value produced by the h last elements of vector y_t^b, *i.e.,* $(y_{t,k-h}^b, y_{t,k-h+1}^b, ..., y_{t,k}^b)$, and $\sigma_t^b(h)$ the corresponding variance. Then the expected time required to download the next chunk of a bitrate b will be equal to:

$$\bar{y}_{t,k+1}^b(h, r) = \bar{y}_t^b(h) + r\sigma_t^b(h) \quad (7)$$

where $r\sigma_t^b(h)$ is an r-weighted term that characterises the risk of a chunk to be delivered after its expected time. Evidently, a *risk averse* end user will choose a high r value while a risk taker end user will set $r = 0$.

Then the expected delivery time of the next chunk $m+1$ for a bitrate $i_{m+1} = b$ is $\tau^{i_{m+1}} = t + \bar{y}_{t,k+1}^b(h, r)$. The end user is finally requesting the bitrate that minimises the next chunk's cost. Note that each vector y_t^b is updated in each buffer measurement iteration by including the additional chunks that would have been delivered for the new extended cumulative function.

4 EVALUATION

4.1 Live streaming traces

At first we asked 10 users, all located in Japan, to watch a popular live streaming youtube channel for over 2 days after installing the "Delay Measuring" chrome extension[1] for monitoring the buffer of their media players. In total 480 hours of data have been collected containing information about the chunks received over time, related to their size as well as the involved content duration. The "Delay

[1]https://chrome.google.com/webstore/detail/delay-measuring/

Resolution	Bitrate	Cost, $\mathcal{V}_b(\cdot)$
144p	80Kbps	91.4
240p	350Kbps	66.2
360p	520Kbps	53.0
480p	830Kbps	34.0
720p	1.6Mbps	7.9
1080p	3Mbps	0.0

Table 1: Resolution and Bitrate Cost

Measuring" extension starts a new video session every hour, so the previously buffered content is discarded. From the size of each chunk we were able to approximate the underline bitrate by using the resolution-Kbps association depicted in Table 1. These bitrates were chosen as representative of 24fps normal motion content.

4.2 Quality Evaluation Setting

Following the resolution-bitrate association, for our evaluation we consider the same resolution categories and we map each chunk resolution to a cost, depicted on the third column of Table 1. We set the resolution of 1080p as the acceptable one with cost/dissatisfaction equal to 0. Similarly, we assume that the maximum bitrate cost that can be produced by a chunk when it is skipped equals to 100, *i.e.,* $\mathcal{V}_b(b_0) = 100$. The cost presented in Table 1 derives by associating the 2 extreme costs, $\mathcal{V}_b(b_0) = 100$ and $\mathcal{V}_b(3Mbps) = 0$, with a positive, decreasing, and convex function for the intermediate resolutions' bitrates.

Next, we assume that the end-users are indifferent between skipping a chunk and experiencing a playback disruption equal to the chunk's duration. Therefore, since our live streaming traces indicate a chunk content duration of 5 seconds, we define the playback disruption cost at 5 seconds equal to 100, *i.e.,* $\mathcal{V}_d(5) = 100$. Similarly, we also assume that the desynchronisation cost at 5 seconds equals to 100, $\mathcal{V}_\delta(5) = 100$. Furthermore, we consider that both playback disruption and desynchronisation cost functions increase linearly until the reception of a chunk, that might exceed the chunk's duration, *e.g.,* $\mathcal{V}_d(6) = \mathcal{V}_\delta(6) = 120$.

Then, we combine all the cost functions into a weighted sum and we consider a **desynchronisation first scenario 1**, where the total cost is $\widetilde{\mathcal{V}}^m(\cdot) = 0.15\mathcal{V}_b(\cdot) + 0.15\mathcal{V}_d(\cdot) + 0.7\mathcal{V}_\delta(\cdot)$, and a **desynchronisation second scenario 2**, where the total cost is $\widetilde{\mathcal{V}}^m(\cdot) = 0.4\mathcal{V}_b(\cdot) + 0.4\mathcal{V}_d(\cdot) + 0.2\mathcal{V}_\delta(\cdot)$. In both scenarios we consider the chunk bitrate to be equally important to playback disruptions.

We compare our quality aware adaptive bitrate mechanism against the youtube performance and an *oracle* approach. To begin with, evaluating the youtube performance under each scenario is straightforward. Secondly, for applying our approach we have to construct for each user the extended cumulative function progressively and apply Eq. (7) to forecast the delivery time of each chunk for each resolution of Table 1. Finally, for the oracle approach we implement a branch and bound algorithm while knowing the extended cumulative function, *i.e.,* the network conditions, since the beginning, so the bitrate chunk choices are optimal.

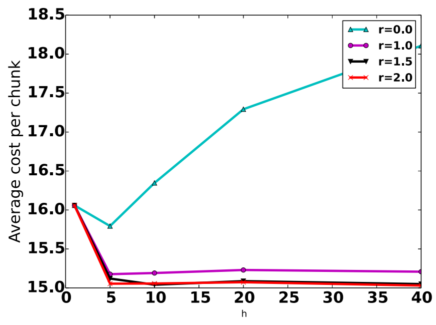

Figure 5: Adaptive Bitrate Parametrisation Scenario 1

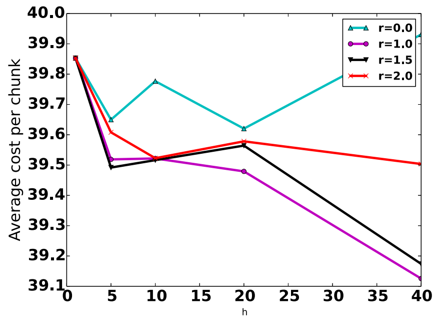

Figure 6: Adaptive Bitrate Parametrisation Scenario 2

4.3 Results

Initially, we investigate our approach in both scenarios for different parameters of previous measurements, h =1, 5, 10, 20, 40, and risk weights, r =0, 1, 1.5, 2. For scenario 1, where the desynchronisation is important, we see in Fig. 5 that if we do not take the risk into account in our forecasting, *i.e.,* $r = 0$, our average cost per chunk will deteriorate as we include more measurements, *i.e.,* h increases. For all other combinations of h and r parameters, the performance of our approach is similar; meaning that when desynchronisation is important the end user has to be risk averse, *i.e.,* $r > 0$. On the other hand, in the second scenario where the desynchronisation is not as important as the other cost factors, the different combinations of h and r result in small performance differences, between 39.1 and 39.9 as it is illustrated in Fig. 6. Generally, we observe that the parametrisation of our approach in both scenarios is quite simple as long as parameter $r > 0$ and parameter $h > 1$.

That said, we set $r = 1$ and $h = 5$ and we compare our approach against the optimal-oracle mechanism and youtube. As shown in Figures 7 and 8 our approach achieves an improvement of 37% and 33%, compared to youtube performance, for scenarios 1 and 2 respectively. At the same time our approach is only worse than the oracle by 25% for scenario 1 and 33% for scenario 2.

5 RELATED WORK

A lot of recent works consider bitrate adaptation approaches for improving the delivery of media streaming. In BOLA [13] a simple

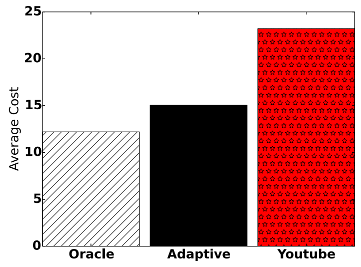

Figure 7: Average Chunk Cost Scenario 1

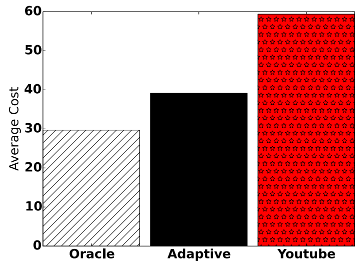

Figure 8: Average Chunk Cost Scenario 2

to implement buffer based algorithm is introduced which efficiency is verified on extensive network traces. ELASTIC [3] aims to keep the buffer occupancy at a constant level. PANDA [7] drops multiplicatively and increases linearly and requested bitrate quality as response to the network bandwidth. In [18] offline optimisation is performed for an exhaustive set of scenarios in order to apply model predictive control for improving QoE related metrics. However, none of these works consider the dimension of synchronisation in media consumption. On the other hand, authors in [9] address explicitly the problem of synchronised media consumption. Nevertheless, their approach relies on the deployment of a new protocol which does not seem promising in being deployed as the bitrate adaptation approach. Unlike previous work, we present a buffer based approach for estimating the network conditions and adapt the requested content bitrate in a SMESs QoE-aware way.

6 CONCLUSIONS

In this work, we created a Quality assessment framework for evaluating the performance of Shared Media Services, in which QoE depends on the synchronised content consumption. Then based on buffered measurements we created a simple to implement and parametrise Quality aware bitrate adaptation approach. Based on real youtube live streaming traces we found that our bitrate adaptation mechanism improves the performance of youtube by more than 30%.

ACKNOWLEDGEMENT

This work was supported by the EC H2020 UMOBILE project (GA no. 645124), the EPSRC COMIT project (grant no. EP/K019589/1), and the EPSRC INSP Early Career Fellowship (no. EP/M003787/1).

REFERENCES

[1] A. Balachandran et al. Developing a predictive model of quality of experience for internet video. In *ACM SIGCOMM*, 2013.
[2] D. P. Bertsekas and J. N. Tsitsiklis. Neuro-dynamic programming: an overview. In *Decision and Control, 1995.* IEEE, 1995.
[3] L. De Cicco et al. Elastic: a client-side controller for dynamic adaptive streaming over http (dash). In *Packet Video Workshop (PV).* IEEE, 2013.
[4] F. Dobrian et al. Understanding the impact of video quality on user engagement. In *ACM SIGCOMM*, 2011.
[5] M. Jarschel et al. An evaluation of qoe in cloud gaming based on subjective tests. In *IMIS.* IEEE, 2011.
[6] D. Le Gall. Mpeg: A video compression standard for multimedia applications. *Communications of the ACM*, 1991.
[7] Z. Li et al. Probe and adapt: Rate adaptation for http video streaming at scale. *IEEE JSAC*, 2014.
[8] J. L. Mitchell. *MPEG video compression standard.* Springer Science & Business Media, 1997.
[9] M. Montagud, F. Boronat, and H. Stokking. Design and simulation of a distributed control scheme for inter-destination media synchronization. In *IEEE AINA, 2013*, 2013.
[10] M. K. Mukerjee et al. Practical, real-time centralized control for cdn-based live video delivery. *ACM SIGCOMM*, 2015.
[11] P. Ni et al. Flicker effects in adaptive video streaming to handheld devices. In *ACM Multimedia*, 2011.
[12] I. E. Richardson. *H. 264 and MPEG-4 video compression: video coding for next-generation multimedia.* John Wiley & Sons, 2004.
[13] K. Spiteri, R. Urgaonkar, and R. K. Sitaraman. Bola: near-optimal bitrate adaptation for online videos. In *IEEE INFOCOM 2016.*
[14] T. Stockhammer. Dynamic adaptive streaming over http–: standards and design principles. In *ACM Multimedia Systems*, 2011.
[15] A. G. Tasiopoulos et al. Tube streaming: modelling collaborative media streaming in urban railway networks. In *IFIP Networking*, 2016.
[16] A. G. Tasiopoulos, I. Psaras, and G. Pavlou. Mind the gap: modelling video delivery under expected periods of disconnection. In *ACM CHANTs*, 2014.
[17] W. Wu et al. Quality of experience in distributed interactive multimedia environments: toward a theoretical framework. In *ACM Multimedia*, 2009.
[18] X. Yin et al. A control-theoretic approach for dynamic adaptive video streaming over http. *ACM SIGCOMM*, 2015.