An Elastic DASH-based Bitrate Adaptation Scheme for Smooth On-Demand Video Streaming

Mohammed Amine Togou and Gabriel-Miro Muntean

School of Electronic Engineering, Dublin City University, Ireland

{mohammedamine.togou, gabriel.muntean}@dcu.ie

***Abstract*—The Video traffic has seen a surge in the last decade due to the widespread use of smartphones and the abundance of video streaming applications in the market. Considering the time varying characteristics of today's networks, ensuring high quality of experience (QoE) to all video traffic users has become a daunting challenge for most service providers. The dynamic adaptive streaming over HTTP (DASH) standard enables the adjustment of video bitrates to match the network conditions, therefore guaranteeing smooth video playback. Different DASH-based approaches have been proposed. Nonetheless, most of these schemes incur substantial bitrate oscillations due to their quick reactions to changes in bandwidth, which negatively impact the users' QoE. In this paper, we propose EDRA, a DASH-based bitrate adaption solution that aims at averting video playback interrupts while reducing the number of bitrate switches. EDRA dynamically adjusts the bounds of available video bitrates based on bandwidth estimations. It then selects the most suitable bitrate for each video segment taking into consideration the current and previous bandwidth measurements, the buffer level and the bitrate variation with respect to the previously downloaded segments. Simulation results show that EDRA outperforms existing commercial schemes as it incurs between 6% and 22% higher accumulated played utility and between 30% and 77% lower bitrate switches, ensuring a smooth video streaming experience at high throughput levels.**

I. Introduction

According to Cisco's Visual Networking Index [1], Internet video traffic (*i.e.,* 56% in High Definition while 20% in Ultra High Definition) will reach 282.3 Exabytes per month by 2022, accounting for 80% of all Internet traffic. Given the limited and highly fluctuated bandwidth of today's networks, video delivery adaptation techniques have been used by video service providers to ensure high quality of experience (QoE) for users of video streaming applications. These techniques adapt the bitrate of the video streams to match the available bandwidth of the network in order to avoid video stalling while ensuring a rapid video startup time. They are based on different protocols and standards, including Microsoft Smooth Streaming (MSS), HTTP Dynamic Streaming (HDS), HTTP Live Streaming (HLS), and Dynamic Adaptive Streaming over HTTP (MPEG-DASH).

MPEG-DASH employs a client-server architecture where videos are stored in the server in different qualities called representations, each of which is encoded in a different bitrate. Every representation is divided into multiple equal length segments. To stream a specific video from the server, the DASH client first requests the Media Presentation Description (MPD), containing the list of all the segments making the video along with their encoded bitrates. Using an adaptation algorithm, the client then starts requesting segments with specific bitrates considering the network condition. Segments are requested sequentially so that no new segment will be requested unless the current one is fully downloaded.

Numerous DASH-based adaptive bitrate (ABR) algorithms [2]–[19] have been proposed in the literature. They differ in terms of the parameters considered when selecting segment bitrates. While some solutions only consider bandwidth and others just consider the buffer levels, most of them are hybrid techniques that consider both bandwidth and buffer level, along with other parameters such as energy consumption, device characteristics, user interest in content, location, etc. Most of these schemes have good performances. Yet, they still face a significant challenge when it comes to maintaining a smooth video quality. This is primarily due to the significant bitrate oscillations incurred as a result of highly dynamic changes of available bandwidth, which might affect the users' QoE.

In this paper, we propose an **E**lastic **D**ASH-based video bit**R**ate **A**daptation approach, labelled **EDRA**, that reduces bitrate fluctuations during on-demand adaptive video streaming. In a two stage process, based on available bandwidth estimations, EDRA first adjusts dynamically the bounds of the list of available video bitrates. Afterwards, EDRA selects within these bounds the most suitable bitrate for transmission of each video segment. It considers various parameters, including the previous and current bandwidth measurements, the buffer level and the video quality variation with respect to previously downloaded segments. To the best of the authors' knowledge, no existing work has proposed a similar solution for smooth video streaming adaptation.

The rest of this paper is organised as follows. Section II surveys existing DASH-based ABR algorithms, classified into three categories: bandwidth-based, buffer-based, and hybrid. Section III describes the system model. Section IV introduces the proposed elastic DASH-based bitrate adaptation solution. Section V describes the simulation setting and discusses the results. Finally, Section VI concludes the paper.

II. Related Work

From a bandwidth and buffer-centric perspective, existing DASH-based ABR schemes can be divided into three broad categories: bandwidth-based, buffer-based and hybrid.

Throughput-based algorithms use the estimated available throughput in order to select the bitrate of the next video segments. Jiang *et al.* [12] proposed FESTIVE, an ABR having two key modules. The *harmonic bandwidth estimator* computes the harmonic mean over k last segments and sends it to the *stateful and delayed bitrate* module, which deploys a gradual bitrate switching strategy to compute the bitrate for next segments while considering the received bandwidth estimations. Li *et al.* [20] proposed a probe-and-adapt approach that aims at circumventing playback stalls along with a conservative and a more responsive approaches to deal with sudden spikes and drops in bandwidth, respectively. Wang *et al.* [13] proposed a light-weight QoE tailored bitrate adaptation algorithm that selects the most suited bitrate for next segments considering QoE metrics such as average bitrate and bitrate variation.

Buffer-based algorithms use the buffer level to decide the bitrate for next segments. When the buffer level is high, buffer-based algorithms would select a high bitrate to avoid buffer overflow. When the buffer level is low, these algorithms choose a low bitrate to avoid buffer underflow, which can lead to playback interruptions. Spiteri *et al.* [14] formulated the video bitrate adaptation as a utility maximization problem and proposed BOLA, an online algorithm that uses Lyapunov optimisation techniques, to select the bitrate for next segments based solely on the amount of data in the buffer. Huang *et al.* [15] proposed BBA, a buffer-based approach that defines two states: startup and steady. To select the bitrate for next segments, BBA uses the bandwidth estimation and the buffer level during the startup state, and exclusively uses the buffer level information during the steady state.

Hybrid algorithms consider both the estimated available bandwidth and the buffer occupancy when selecting the bitrate of the next segments. In [16], Spiteri *et al.* proposed DYNAMIC, an ABR algorithm that uses bandwidth estimation when the buffer level is low and switches to BOLA when the buffer level is high to minimise rebuffering and bitrate oscillations while maximising the average video bitrate. Yaqoob *et al.* [3] proposed TBOA, a throughput and buffer occupancy-based adaptation scheme which downloads the first few segments with the lowest bitrate and adjusts the bitrates of the subsequent segments based on bandwidth estimations and buffer level. Zhou *et al.* [10] proposed a Markov decision-based rate adaptation scheme that takes into account video playback quality, bitrate switching frequency and amplitude, buffer level and buffer underflow/overflow events. Yin *et al.* [19] proposed a model predictive control algorithm that considers bandwidth estimation and buffer level to make optimal bitrate decisions for QoE maximisation. Finally, Garcia *et al.* [11] proposed a bitrate adaptation solution based on Stochastic Dynamic Programming that selects the best suited bitrate through a cost function considering the buffer level along with the estimated bandwidth and the bitrate difference.

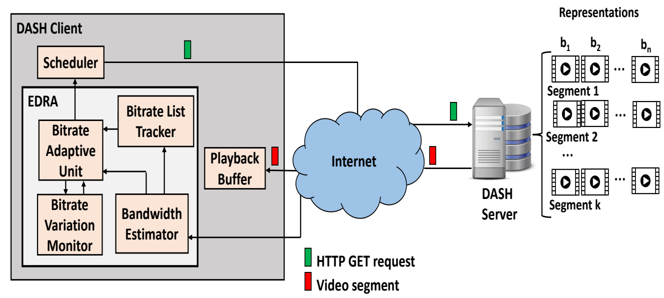


Fig. 1. Block architecture of the elastic DASH-based bitrate adaptation solution

Despite their performance, the aforementioned schemes still suffer from considerable bitrate oscillations due to their rapid reactions to changes in network conditions. Therefore, we propose EDRA, a DASH-based bitrate adaptation scheme that dynamically adjusts the bounds of the list of available bitrates while slowly reacting to bandwidth changes in order to provide smooth playback quality experience.

III. System Model

We assume that a video is split into k segments, each of which is T seconds long and is encoded in n different bitrates, *i.e.,* $b = \{b_1, b_2, \ldots, b_n\}$. The DASH-client selects b_i, the bitrate of the i^{th} segment of the video, considering the available bandwidth, the playback buffer level and the quality variation with respect to previous segments. The next segment (*i.e.,* $i+1$) is requested once segment i is fully downloaded. Segment i is then decoded and placed in the playback buffer to be played when needed.

Fig. 1 illustrates the block architecture of EDRA. It consists of four modules. The *Bitrate List Tracker* controls the bounds of the bitrate list based on bandwidth estimations. The *Bandwidth Estimator* computes the network's available bandwidth while the *Bitrate Variation Monitor* stores the difference in bitrates among the previously downloaded segments. The *Bitrate Adaptive Unit* selects the bitrate of the next segment to be requested based on the information received from the aforementioned modules along with the buffer level information received from the *Playback Buffer*. Once the bitrate is selected, the *Bitrate Adaptive Unit* informs the *Bitrate Variation Monitor* and the *Scheduler*. The latter sends the HTTP GET request to the DASH server to start downloading the segment.

IV. Elastic DASH-based Bitrate Adaptation

This section explains how the available bandwidth, the buffer level and the bitrate variation are computed and how they are used to select the bitrate for the video segments.

A. Available Bandwidth

The main challenge of any video bitrate adaptation scheme is to accurately estimate the available bandwidth given its stochastic nature over best effort networks. A possible way to address such a problem is to use a smoothed bandwidth estimation which stabilizes the predictions over time to reduce the impact of frequent bandwidth fluctuations [3], [21]. Let BW_i be the required bandwidth for segment i encoded using bitrate b_i. BW_i can be expressed as:

$$BW_i = \frac{T \times b_i}{d_i} \tag{1}$$

where d_i is the download time of segment i. We adopt a smoothed prediction approach that uses a moving average to estimate the available bandwidth. It is expressed as follows:

$$BW_i^e = \begin{cases} BW_{i-1}, & i = 1 \\ \alpha_1 BW_{i-1} + \alpha_2 BW_{i-1}^e, & i > 1 \end{cases} \tag{2}$$

where $\alpha_1, \alpha_2 > 0$ are smoothing coefficients. Note that Eq. (2) takes into consideration the short spikes (*i.e.,* drops as well) in bandwidth that may happen over time and which may influence the bandwidth estimations, yielding high bitrate variability for video segments.

B. Buffer Level

The DASH client has one playback buffer in charge of storing the segments once they are downloaded and decoded. We define the buffer level in this paper as the number of segments contained in the playback buffer. It expands when new segments are added to the buffer and shrinks when segments are played (*i.e.,* removed from the buffer). Let $B_i \in [0, B_{max}]$ be the buffer level after downloading segment i. We can express the dynamics of the buffer level as follows:

$$B_i = B_{i-1} + 1 - \left[\frac{d_i}{T}\right] \tag{3}$$

The first term in Eq. (3) represents the buffer level at the start of segment i download while the second term represents the increment of the buffer level once segment i is fully downloaded. The third term represents the number of segments played while downloading segment i. [.] is used for rounding the third term to the closest integer. By analyzing Eq. (3), we can observe that it is dominated by the third term. Indeed, when d_i is long, the playback buffer may run dry (*i.e.,* buffer underflow) before fully downloading segment i, leading to playback interruptions. In this case, a rebuffering period is triggered in which the buffer is filled with the video streaming paused. On the other hand, when d_i is short, segments will be continuously downloaded into the playback buffer, which might induce buffer overflow. To mitigate these problems, we introduce two thresholds: B_l and B_h. In case the buffer level is less than B_l, segments will be downloaded at low bitrates to quickly fill in the buffer, avoiding therefore video stalling. In case the buffer level is greater than B_h, segments will be downloaded with higher bitrates to allow for the buffer level to be reduced, averting the buffer overflow problem.

C. Quality Variation

Numerous studies [10], [11], [22] have shown that high bitrate variation among video segments can significantly decrease the user's QoE. Therefore, to reduce the frequent bitrate switches, we use a moving average approach to keep track of the bitrate variation. It is computed as follows:

$$q_i = \begin{cases} b_1, & i = 1 \\ (1-\beta)(b_i - b_{i-1}) + \beta q_{i-1}, & i > 1 \end{cases} \tag{4}$$

where $\beta \in [0, 1]$ is a smoothing coefficient. Note that Eq. (4) captures the short-term bitrate variation and gives higher weight to bitrate changes of recent segments as they are more likely to influence the user's perceived QoE.

D. Bitrate Selection Algorithm

Let b_{min} and b_{max} be the dynamic bounds of the bitrates list b computed by the *Bitrate List Tracker* based on bandwidth estimation. Initially, b_{min} and b_{max} are both set to b_1 as no bandwidth estimation exists. After downloading following segments, the bounds are adjusted based on the bandwidth estimation according to Algorithm 1. To this end, the bandwidth status is checked. If it tends to increase, b_{max} is set to the highest bitrate that is lower than the estimated bandwidth while b_{min} is set to the next highest bitrate in the list (lines $9-13$). If it decreases, b_{min} is set to the bitrate having an index that is two decrements from the index of b_{max} (lines $14-18$). This is to limit the number of bitrates from which EDRA should choose in order to avert substantial bitrate oscillations when sudden short bandwidth spikes/drops occur. Consequently, this will help enhance the users' QoE.

Algorithm 1: Dynamic adjustment of b_{min} and b_{max}

Result: b_{min} and b_{max}
1 **while** $i \neq k$ **do**
2   **if** $i = 1$ **then**
3     $b_{min} \leftarrow b_1$ and $b_{max} \leftarrow b_1$
4     $BW_{last} \leftarrow 0$
5   **else**
6   **if** $i > 2$ **then**
7       $BW_{last} \leftarrow BW_{i-2}$
8   **end**
9   **if** $BW_{i-1} - BW_{last} > 0$ **then**
10   **if** $b_{max} \leq BW_{i-1}$ **then**
11         $b_{max} \leftarrow \max\{b_j \in b \mid b_j \leq BW_{i-1}\}$
12         $b_{min} \leftarrow \min\{b_j \in b \mid b_j > b_{min}\}$
13   **end**
14   **else**
15   **if** $b_{min} > BW_{i-1}$ **then**
16         $b_{max} \leftarrow \max\{b_j \in b \mid b_j \leq BW_{i-1}\}$
17         $b_{min} \leftarrow \max\{b_j \in b \mid max - j = 2\}$
18   **end**
19   **end**
20   **end**
21 **end**

Algorithm 2 illustrates the bitrate selection process. At the start of the streaming session, the DASH client is not aware of the network condition. Most adaptive schemes adopt a conservative approach where the first few segments of the video are downloaded at the lowest supported bitrate. This is to fill-in the buffer quickly, reducing therefore the startup delay. Yet, several studies [23], [24] have shown that users are willing to tolerate larger startup delays; others [25] have also shown that a low startup bitrate followed by a slow quality increase clearly degrades the user's QoE. Unlike existing solutions, EDRA adopts a more liberal approach where the first segment is downloaded at the lowest supported bitrate (*i.e.,* b_{min}). The bitrate of the next segment is selected based on the estimated download time while ensuring that the buffer level is higher than 0. The same bitrate is used for the following segments till the buffer level exceeds B_l. In case the buffer level is between B_l and B_h, EDRA selects the bitrate for segment i that meets the following conditions:

$$\begin{cases} b_i \leq BW^e_{i-1} \\ \dfrac{|b_i - b_{i-1}|}{q_i} \leq 1 \\ (T \times B_i) - d_i \geq T \times B_l \end{cases} \tag{5}$$

The first condition indicates that the bitrate of segment i should not exceed the estimated bandwidth while the second specifies that the bitrate of segment i should be the next element in the bitrate list with respect to b_{i-1} in either ascending or descending order. The third condition implies that the buffer level should be higher than the threshold B_l after downloading segment i. Finally, in case the buffer level exceeds B_h, EDRA waits for a period of time before requesting the next segment. This is to avoid the buffer overflow problem. Still, chances of the buffer underflow occurring during this period cannot be ignored particularly in the case of a sharp drop in bandwidth or a low number of segments in the buffer once the this period expires. As a result, the waiting period is computed as follows:

$$\tau = T\left(B_{i-1} - \left[\frac{B_l + B_h}{2}\right]\right) \tag{6}$$

V. Performance Evaluation

To assess the performance of EDRA, we used Sabre [16], an open-source tool for simulating ABR environments, that takes three inputs:

- **Network trace:** contains a sequence of records, each of which includes the time duration, network throughput, and latency. We used two network traces in our simulations: nt_1 [26] and nt_2 [27] nt_1 has four periods, each one lasts 30 seconds. The first period allows a bandwidth of 5000Kbps, with a round-trip latency of 75ms. When the four periods come to an end, Sabre restarts at the top. nt_2 has multiple periods with the same round-trip latency (*i.e.,* 100ms) and variable bandwidth.
- **Video description:** analogous to the DASH manifest and includes the segment length T, the encoded bitrates, and the segment size matrix C where $C[i, j]$ represents the size of the i^{th} segment of the video encoded at the j^{th} bitrate. The video description file [28] used in the simulations is of the Big Buck Bunny Movie, a 10 minutes movie encoded in ten different bitrates (Kbps), b = {230, 331, 477, 600, 991, 1427, 2056, 2962, 5027, 6000}.
- **ABR algorithm:** refers to the algorithm invoked before downloading a new segment. EDRA, THROUGH (*i.e.,* bitrate selected based only on estimated bandwidth), BOLA [14], and DYNAMIC [16] are used in turn. The last two are part of the DASH reference player dash.js [29] and are used commercially by many content providers (*e.g.,* akamai, edgeware, brightcove).

Algorithm 2: Bitrate Selection

Result: b_i, bitrate of segment i

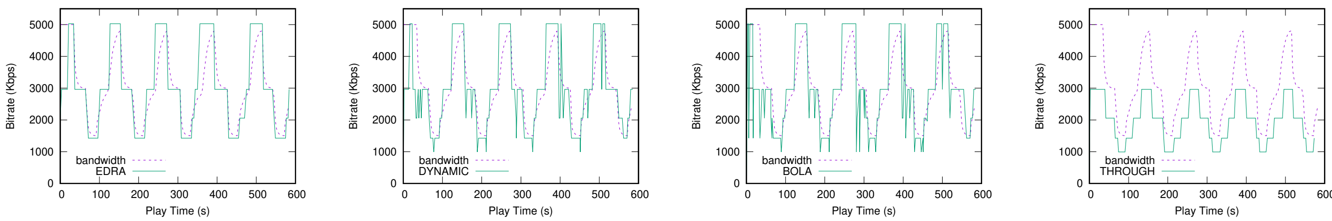
while $i \neq k$ **do**
  if $i = 1$ **then**
    $b_i \leftarrow b_{min}$
  else
    if $B_{i-1} \leq B_l$ **then**
      $b_i \leftarrow \max\{b_j \in b \text{ s.t. } (T \times B_i) - d_i > 0\}$
    end
    if $B_l < B_{i-1} \leq B_h$ **then**
      $b_i \leftarrow \max\{b_j \in b \text{ s.t. Eq. (5) holds}\}$
    end
    if $B_{i-1} > B_h$ **then**
      wait for τ seconds
    end
  end
end

TABLE I
SIMULATION PARAMETERS

Parameter	Value
T	3s
B_{max}	25s (8 segments)
B_l	10s (3 segments)
B_h	22s (7 segments)

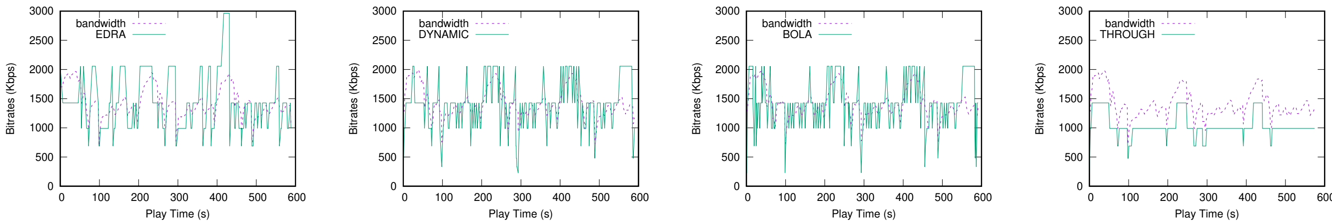
We evaluate comparatively four ABR algorithms in terms of the total number of bitrate switches (BTS), the average incurred throughput (ATH), the total reaction time (TRT), *i.e.,* defined as the time it takes to start rendering at the highest sustainable bitrate after the network bandwidth increases, and the accumulated played utility (AU). The latter is used by Sabre to reflect the users' QoE and computed as $AU = \sum_{i=1}^{k} log(b_i)$. Other simulation parameters are included in Table I. Note that $\alpha_1 = 3$ and $\alpha_2 = 8$ are the default values used in Sabre while β was set to 0.3 in these simulations.

Fig. 2 and Fig. 3 3 show the bitrate of the video playback as a function of the video play time using nt_1 and nt_2, respectively. We observe that both EDRA and THROUGH provide a smoother video play quality with fewer bitrate switches compared to DYNAMIC and BOLA. Indeed, using nt_1, both EDRA and THROUGH incur a number of bitrate



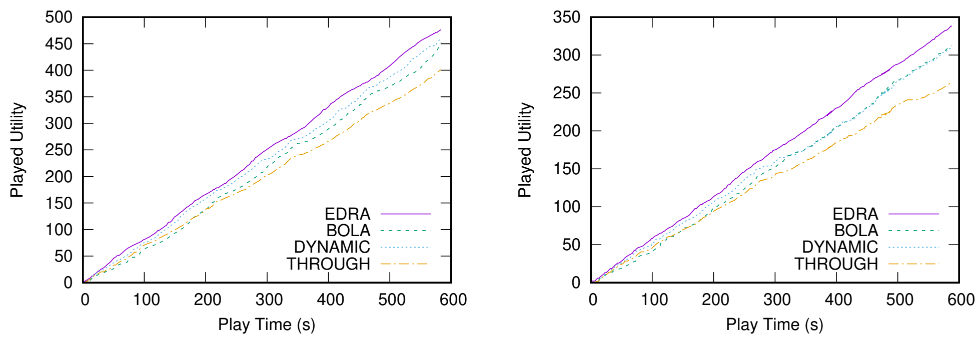
(a) EDRA (b) DYNAMIC (c) BOLA (d) THROUGH

Fig. 2. Bitrate of the video playback as a function of the video play time for the various ABR algorithms using nt_1



(a) EDRA (b) DYNAMIC (c) BOLA (d) THROUGH

Fig. 3. Bitrate of the video playback as a function of the video play time for the various ABR algorithms using nt_2



(a) nt_1 (b) nt_2

Fig. 4. Total played utility as a function of the video play time using nt_1 and nt_2

switches that is 77% and 67% lower than BOLA and DYNAMIC, respectively (see BTS in Table II). When using nt_2, THROUGH incurs the lowest number of bitrate switches with EDRA incurring a number of bitrate switches that is 40% and 30% lower than BOLA and DYNAMIC, respectively (see BTS in Table III). Fig. 4 illustrates the accumulated played utility as a function of the video play time using nt_1 and nt_2. We observe that EDRA provides the highest AU compared to the other ABR algorithms for both traces. Indeed, using nt_1 EDRA provides an average AU which is 6%, 13%, and 20% higher than DYNAMIC, BOLA, and THROUGH, respectively. When using nt_2, EDRA provides an average AU which is 11%, 13%, and 22% higher than DYNAMIC, BOLA, and THROUGH, respectively.

The reason EDRA outperforms the remaining ABR schemes is twofold. First, EDRA limits the number of available bitrates by dynamically adjusting the values of b_{min} and b_{max} along with maintaining the selected bitrate as long as the buffer level allows it to avoid rebuffering events. Second, EDRA is slow to react to the changes in bandwidth to avoid bitrate oscillations that may occur due to sudden and short spikes/drops in bandwidth (see TRT in Tables II and III). As a result, EDRA generates the lowest number of bitrate switches (*i.e.,* as opposed to BOLA and DYNAMIC) and the highest average throughput (see ATH in Tables II and III), providing higher AU (*i.e.,* perceived QoE) in comparison to the remaining ABR algorithms. Note that THROUGH only considers bandwidth estimations when selecting the bitrate of segments and deploys a conservative approach that quickly reacts to bandwidth changes to evade the buffer underflow problem. This leads to low average throughput and high reaction time. Note also that none of the simulated ABR schemes incurred any buffering events when using both network traces.

TABLE II

QoS METRICS USING nt_1

	EDRA	**BOLA**	**DYNAMIC**	**THROUGH**
BTS	29	65	58	29
TRT (s)	86	82	82	225
ATH (Kbps)	2921	2877	2905	1964

TABLE III
QoS Metrics using nt_2

	EDRA	BOLA	DYNAMIC	THROUGH
BTS	78	117	106	22
TRT (s)	21	9.83	11.82	50.7
ATH (Kbps)	1370	1353.4	1323.1	1028.2

VI. Conclusions

In this paper, we proposed EDRA, a new ABR algorithm that aims at ensuring a smooth video playback quality through reducing bitrate oscillations. It starts by dynamically adjusting the bounds of the available bitrates list based on bandwidth estimations; then, it selects the most suitable bitrate for every video segment considering the buffer level, the available bandwidth and the bitrate difference with respect to previously downloaded segments. Simulation results demonstrates that EDRA incurs low bitrate switches compared to well known commercial ABR schemes (*e.g.,* BOLA, DYNAMIC) while providing higher throughput.

Future work will focus on deploying EDRA in a production setting, particularly the DASH reference player. In addition, EDRA will be integrated in the Co-Creation Stage, a novel artistic co-creation tool that is one of the results of the EU Horizon 2020 project TRACTION, to enable the creation and delivery of collaborative remote Opera performances.

Acknowledgments

This work was supported by the European Union's Horizon 2020 Research and Innovation programme under Grant Agreement no. 870610 for the TRACTION project. The support of the Science Foundation Ireland (SFI) Research Centres Programme Grant Numbers 12/RC/2289_P2 (Insight SFI Centre for Data Analytics) and 16/SP/3804 (SFI ENABLE Spoke) is gratefully acknowledged.

References

[1] "Cisco Annual Internet Report (2018–2023) White Paper," San Jose, CA, Tech. Rep. c11-741490, March 2020. [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html

[2] G.-M. Muntean, "Efficient delivery of multimedia streams over broadband networks using qoas," *IEEE Transactions on Broadcasting*, vol. 52, no. 2, pp. 230–235, 2006.

[3] A. Yaqoob, T. Bi, and G. Muntean, "A DASH-based Efficient Throughput and Buffer Occupancy-based Adaptation Algorithm for Smooth Multimedia Streaming," in *2019 15th International Wireless Communications Mobile Computing Conference (IWCMC)*, 2019, pp. 643–649.

[4] L. Zou, T. Bi, and G.-M. Muntean, "A dash-based adaptive multiple sensorial content delivery solution for improved user quality of experience," *IEEE Access*, vol. 7, pp. 89 172–89 187, 2019.

[5] A.-N. Moldovan and C. H. Muntean, "Dqamlearn: Device and qoe-aware adaptive multimedia mobile learning framework," *IEEE Transactions on Broadcasting*, vol. 67, no. 1, pp. 185–200, 2021.

[6] A. Zhang, Q. Li, Y. Chen, X. Ma, L. Zou, Y. Jiang, Z. Xu, and G.-M. Muntean, "Video super-resolution and caching—an edge-assisted adaptive video streaming solution," *IEEE Transactions on Broadcasting*, vol. 67, no. 4, pp. 799–812, 2021.

[7] A. Yaqoob and G.-M. Muntean, "A combined field-of-view prediction-assisted viewport adaptive delivery scheme for 360° videos," *IEEE Transactions on Broadcasting*, vol. 67, no. 3, pp. 746–760, 2021.

[8] D. Yun and K. Chung, "Dash-based multi-view video streaming system," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 8, pp. 1974–1980, 2018.

[9] S. Q. Jabbar, D. J. Kadhim, and Y. Li, "Proposed an Adaptive Bitrate Algorithm based on Measuring Bandwidth and Video Buffer Occupancy for Providing Smoothly Video Streaming," *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 2, 2018.

[10] C. Zhou, C. Lin, and Z. Guo, "mDASH: A Markov Decision-Based Rate Adaptation Approach for Dynamic HTTP Streaming," *IEEE Transactions on Multimedia*, vol. 18, no. 4, pp. 738–751, 2016.

[11] S. García, J. Cabrera, and N. García, "Quality-Control Algorithm for Adaptive Streaming Services Over Wireless Channels," *IEEE Journal of Selected Topics in Signal Processing*, vol. 9, no. 1, pp. 50–59, 2015.

[12] J. Jiang, V. Sekar, and H. Zhang, "Improving Fairness, Efficiency, and Stability in HTTP-Based Adaptive Video Streaming With Festive," *IEEE/ACM Transactions on Networking*, vol. 22, no. 1, pp. 326–340, 2014.

[13] C. Wang, A. Rizk, and M. Zink, "SQUAD: A Spectrum-Based Quality Adaptation for Dynamic Adaptive Streaming over HTTP," in *Proceedings of the 7th International Conference on Multimedia Systems (MMSys '16)*, 2016, pp. 1–12.

[14] K. Spiteri, R. Urgaonkar, and R. K. Sitaraman, "BOLA: Near-optimal bitrate adaptation for online videos," in *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*, 2016, pp. 1–9.

[15] T.-Y. Huang, R. Johari, N. McKeown, M. Trunnell, and M. Watson, "A Buffer-Based Approach to Rate Adaptation: Evidence from a Large Video Streaming Service," in *Proceedings of the 2014 ACM Conference on SIGCOMM*, 2014, p. 187–198.

[16] K. Spiteri, R. Sitaraman, and D. Sparacio, "From Theory to Practice: Improving Bitrate Adaptation in the DASH Reference Player," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 15, no. 2s, Jul. 2019.

[17] D. J. Vergados, A. Michalas, A. Sgora, D. D. Vergados, and P. Chatzimisios, "FDASH: A Fuzzy-Based MPEG/DASH Adaptation Algorithm," *IEEE Systems Journal*, vol. 10, no. 2, pp. 859–868, 2016.

[18] A. Beben, P. Wiundefinedniewski, J. M. Batalla, and P. Krawiec, "ABMA+: Lightweight and Efficient Algorithm for HTTP Adaptive Streaming," in *Proceedings of the 7th International Conference on Multimedia Systems*, 2016, pp. 1–11.

[19] X. Yin, A. Jindal, V. Sekar, and B. Sinopoli, "A Control-Theoretic Approach for Dynamic Adaptive Video Streaming over HTTP," *SIGCOMM Comput. Commun. Rev.*, vol. 45, no. 4, p. 325–338, Aug. 2015. [Online]. Available: https://doi.org/10.1145/2829988.2787486

[20] Z. Li, X. Zhu, J. Gahm, R. Pan, H. Hu, A. C. Begen, and D. Oran, "Probe and Adapt: Rate Adaptation for HTTP Video Streaming At Scale," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 4, pp. 719–733, 2014.

[21] T. Su, A. Sobhani, A. Yassine, S. Shirmohammed, and A. Javadtalab, "A DASH-based HEVC Multi-View Video Streaming System," *Journal of Real-Time Image Processing*, vol. 12, pp. 329–342, 2016.

[22] N. Barman and M. G. Martini, "QoE Modeling for HTTP Adaptive Video Streaming–A Survey and Open Challenges," *IEEE Access*, vol. 7, pp. 30 831–30 859, 2019.

[23] N. Staelens, J. De Meulenaere, M. Claeys, G. Van Wallendael, W. Van den Broeck, J. De Cock, R. Van de Walle, P. Demeester, and F. De Turck, "Subjective Quality Assessment of Longer Duration Video Sequences Delivered Over HTTP Adaptive Streaming to Tablet Devices," *IEEE Transactions on Broadcasting*, vol. 60, no. 4, pp. 707–714, 2014.

[24] M.-N. Garcia, F. De Simone, S. Tavakoli, N. Staelens, S. Egger, K. Brunnström, and A. Raake, "Quality of experience and HTTP adaptive streaming: A review of subjective studies," in *2014 Sixth International Workshop on Quality of Multimedia Experience (QoMEX)*, 2014, pp. 141–146.

[25] Liu Yitong, Shen Yun, Mao Yinian, Liu Jing, Lin Qi, and Yang Dacheng, "A study on Quality of Experience for adaptive streaming service," in *IEEE International Conference on Communications Workshops (ICC)*, 2013, pp. 682–686.

[26] K. Spiteri, "sabre/network.json," 2019. [Online]. Available: https://github.com/UMass-LIDS/sabre/blob/master/example/network.json

[27] K. Spiteri, "sabre/report2010-09-13_1003cet.json," 2019. [Online]. Available: https://github.com/UMass-LIDS/sabre/blob/master/example/tomm19/3Glogs/report.2010-09-13_1003CEST.json

[28] K. Spiteri, "sabre/movie.json," 2019. [Online]. Available: https://github.com/UMass-LIDS/sabre/blob/master/example/movie.json

[29] DASH Industry Forum, "dash.js JavaScript Reference Client," 2021. [Online]. Available: https://reference.dashif.org/dash.js/