An Adaptive Bitrate Algorithm Based on Estimation and Video Adaptation for Improving QoE in DASH

Saba Qasim Jabbar[1,*], Dheyaa Jasim Kadhim[2] and Yu Li[1]

[1]School of Electronics and Information, Huazhong University of Science and Technology, Wuhan National Laboratory for Optoelectronics, Division of Communication and Intelligent Networks, Wuhan, Hubei, 430074, P.R. China.
[2]Electrical Engineering Department, University of Baghdad, Baghdad, Iraq
[*]Corresponding author

Abstract—**Dynamic Adaptive streaming over HTTP (DASH) is widely used by content providers for video delivery and dominates traffic on cellular networks. The variation in both video bitrate and network bandwidth badly impacts on the user Quality of Experience (QoE), so recent works is going to submitting better design of DASH adaptation algorithms. In this work, an adaptive bitrate algorithm is proposed which incorporates the network state, the application state and the video variety conditions to adapt video quality under time varying wireless system. The proposed algorithm consists of two main units: estimation unit and video adaptation unit. During the estimation unit, a new attribute is included that scales the buffer occupancy estimation and the throughput estimation based on the variations in the current buffer level and measured throughput of previous download segments. While during the video adaptation unit, the selection of video bitrate is done based on both the measurements at estimation unit and a target level. The simulation results show that the proposed algorithm significantly outperforms the other rate adaptation algorithms in terms of providing maximum download bit rates, minimum number of bitrates switches and the maximum utilization of available bandwidth while maintaining the playback buffer level within limits without any playback interruptions.**

Keywords—video streaming; DASH, video buffer; adaptive rate algorithm; QoE

I. INTRODUCTION

Video content constitutes a dominant fraction of Internet traffic today. Furthermore to ride the Internet bandwidth requirements, current video streaming systems adopt some adaptive and scale-able video coding techniques to facilitate the transmission [1]. Dynamic adaptive streaming over HTTP (DASH) is an HTTP-based technique in which each video is divided into segments encoded at multiple rates [2]. Each segment is encoded into different quality levels that may vary in resolution, quality and bitrates. This approach provides TCP fairness in the network as well as ordered lossless delivery. However, the adaptive algorithm at client side does not be defined by DASH system for choosing which segment to request or how to adapt the video bitrates so as to get good Quality of Experience (QoE). One of the commercial solutions for adaptive streaming is the adaptive algorithm “Adobe Open Source Media Framework" (OSMF) [3], which considers the download time of last video segments and decides a suitable bitrate for the next segment to be fetched. The state-of-the-art for adaptive bitrates algorithms can be generally divided as buffer-based [4], rate-based [5] and hybrid strategies [6][7].

One of the main features in wireless system is the time variations due to its multi-path nature, user mobility, and medium sharing. Moreover, the changing nature of the bitrates in video traffic contributes to the system's natural change. Such variable operating conditions may significantly affect the performance of different adaptation algorithms that degrade the QoE [8]. In this approach, there are some related works such as [9], which the authors utilized the physical layer information to enhance the available throughput estimation and then enhancing the streaming performance in cellular networks. In contrast, the proposed algorithm only bases on application level information and it can be used with any access technology. In [10], authors introduced an adaptation scheme based on a Markov Decision Process (MDP) framework. The authors suggested three heuristics that can use online or offline estimation for available bandwidth in statistic way to absorb MDP complexity. In [11] A. Zahran et al. proposed OSCAR algorithm which it is an optimized algorithm controlled by a probabilistic constraint on stalls using a detailed optimization framework to improve the quality of experience. In [12] the authors studied the problems of bitrate adaptation and determined the causes of many unwanted interactions that arise as a result of modifying the video bitrate over HTTP. In [13] the author investigates the delivery of on demand video streaming under variable bit rate (VBR) coding. The adaptation method is based on detailed instant bitrates of future segments in order to understand the fluctuation traits of the VBR video.

In this work, an adaptive bitrate algorithm is proposed which incorporates the network state, the application state and the video variety conditions to adapt video quality under time varying wireless system. Essentially, the proposed algorithm consists of two main units: estimation unit and video adaptation unit where the download throughput and the buffer occupancy are measured through estimating the available network bandwidth according to the current buffer occupancy of last downloaded segments. During the estimation step, a new attribute is included that scales the buffer occupancy estimation and the throughput estimation based on the variations in the current buffer level and measured throughput of previous download segments. The variation is taken at short term level. At video adaptation unit, the selection of video bitrate is done based on both the measurements at estimation unit and a target level. The target level is defined for checking the new level if where it is near or far from the target in order to prevent buffer overflow/underflow situations and achieve high quality of experience at user aspect. During the simulation results, the proposed algorithm significantly outperforms the other rate adaptation algorithms in terms of providing maximum download bit rates, minimum number of bitrates switches and the maximum utilization of available bandwidth while maintaining the playback buffer level within limits without any playback interruptions.

The work of this paper is organized as follows: Section II describes our system design and construction of the video streaming model that we will consider in this work, also this section will explain the proposed algorithm which it consists mainly from two parts (estimation unit and video adaption unit). Section III presents our simulation results of experimental tests for our proposed approach in this work. Finally section IV gives the main conclusions that can be drawn from our work.

II. Video Streaming Model Design

We consider a client streaming video divides into *M* segments each of which contains τ seconds of video. We assume a continuous HTTP connection over which segments are sequentially requested using HTTP GET requests. Each segment is encoded into *L* video quality representations Q_v with each representation having an average encoding rate V_R, where $Q_v \epsilon \{1, 2, ..., Q_{VL}\}$. For each i^{th} segment, the client selects the video bitrate V_{Ri} from $V=\{V_{R1}, V_{R2}....,V_{RL}\}$. The main goal of the propose algorithm is to improve the quality of experience (QoE) by minimizing QoE metrics such as number of interrupts, duration of interrupts, average quality rate, average number of quality switches, and average level of quality switching. The streaming system which it is considered in this work is consisting from main two parts (estimation unit and video adaptation unit. Figure 1 below shows the client/ server system that it is considered for our work, where the network bandwidth represents the data producing rate and video content's bitrate represents the data consuming rate to the video buffer of the client. At the beginning of each download stage, an adaptation algorithm at client side will select the suitable bitrate of the next segment to be downloaded. The rates of the different video representation levels are transferred to the client by the server in the Media Presentation Data (MPD) file.

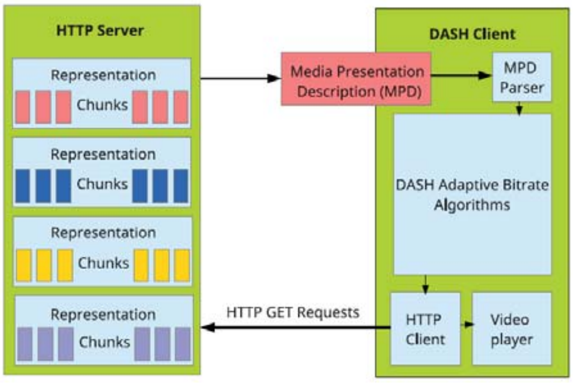

FIGURE I. HTTP STREAMING SYSTEM

A. *Estimation Unit*

The estimation unit includes the estimate for both buffer occupancy and download throughput for i^{th} segment. At t_i the video player starts downloading the i^{th} segment. The segment download time is $\tau V_{Ri} / T_i$ which is the ratio between the size of chosen segment i.e. τV_{Ri} according to quality level and the average download throughput T_i. Immediately after the download process is completed for i^{th} segment, the video player then starts download the $(i+1)^{th}$ segment, if the download throughput at time *t* is $T_i(t)$ hence:

$$t_{i+1} = t_i + \frac{\tau V_{Ri}}{T_i} \tag{1}$$

$$T_i = \frac{1}{t_{i+1} - t_i} \int_{t_i}^{t_{i+1}} T_i(t)\,dt \tag{2}$$

When segments are downloaded and video is being playback, the video buffer occupancy is developed. The buffer occupancy is changed by τ sec after i^{th} segment is downloaded and minimizes as the user watches the video. The dynamic changes for buffer occupancy could be formulated at time t as:

$$B_{i+1}(t) = \left(\left(B_i - \frac{\tau V_{Ri}}{T_i} \right)_+ + \tau \right) \tag{3}$$

Here, the notation $(x)_+ = max\{x,0\}$ to ensures that the term can never be negative as well as if $B_i < \tau V_{Ri} / T_i$, the buffer occupancy becomes empty while the video player is still downloading i^{th} segment, leading to stopping events. Now, let W_B be the weighted mean estimation of buffer occupancy which is based on an exponentially weighted changing estimation window that absorbs S_b samples with *i* be the index of the last downloaded segment then after i^{th} segment download the buffer occupancy would be assigned with a weight, denoted as *s* while the weight of previous segments in the buffer estimation window would be assigned as $s\,(1-s)^{i-j}$ where *j* is the segment index. Then these weights are normalized and the final weights are expressed as:

$$s_j = \frac{s\,(1-s)^{i-j}}{1-(1-s)^{S_b}}, \forall j \in \{1,.....,S_b\} \tag{4}$$

The weighted mean of buffer occupancy can be expressed as:

$$W_B = \sum_{j=1}^{S_b} s_j B_{i-j} \tag{5}$$

where B_i the measured buffer occupancy at the download of i^{th} segment and $s_j \in (0,1)$. The response of the estimation unit would enhance the changes in buffer occupancy if *s* be with large value but at the same time might lead to frequent disturbing switches in video quality level. The proposed algorithm adapts the weighted mean of buffer occupancy based on information on network and application states. Usually when the buffer level is low it means a bad network condition that might lead to buffer underflow and impact on video quality of experience. While a high buffer level means the network condition is good and the buffer occupancy would reach the maximum level size B_{max}, the streaming client stops requesting new segments until there exists enough buffer space for one or more segments which leads to ON-OFF behavior and hence effects on QoE [14]. The OFF interval affects on the adaptation process for the bitrate algorithm since the client would not have an accurate estimate for the available network condition during OFF periods. The proposed algorithm uses a scaling factor for buffer level to avoid stalls and construct the video buffer. The scaling factor for buffer level f_B is given as:

$$f_B = f_B^L + (f_B^H - f_B^L)\frac{B_i}{B_{max}} \tag{6}$$

where f_B^H and f_B^L is the higher and lower bound of scaling factor and B_{max} is the maximum buffer size. The above equation scales up the rate of estimation to avoid OFF periods and improve the video. The estimation process also includes the network throughput since any fluctuation in the available bandwidth would impact on the buffer occupancy, so we use a smoothing factor ρ that reflects the changing in network conditions as:

$$\hat{T}_i = \rho T_{i-1} + (1-\rho)\hat{T}_{i-1} \tag{7}$$

A large value of ρ means the network conditions has changed abruptly and so the upcoming bandwidth is calculated by giving more weight to the last available throughput. Contrarily, bandwidth is calculated by giving more weight to the last estimated throughput. Finally, our adjusted throughput estimate *T* is calculated as:

$$\tilde{T}_i = f_B \times W_B \times \hat{T}_i \tag{8}$$

B. Video Adaption Unit

The last step of adaptive algorithm is to specify a suitable representation quality for *(i+1)*th segment such that $V_{Ri} \leq \check{T}_i$ and buffer level is at the steady region where $0 < B_L < B_i < B_H < B_{max}$. B_L and B_H are two operating thresholds for preventing buffer underflow/overflow, measured in seconds, B_i is the current buffer occupancy after download *i*th segment and B_{max} is the buffer maximum size. Let B_{aim} represent the target for keeping the buffer level between these thresholds as:

$$B_{aim} = \frac{B_L + B_H}{2} \tag{9}$$

The HTTP adaptive streaming clients download the segments without any wait until the video buffer level reaches the target level B_{aim}.

The following steps represent our proposed adaptive algorithm procedure:

```
input : $\hat{T}_{i+1}, s, S_b, f_B^L, f_B^H, B_{max}, B_i, L, V, \tau$
output : $V_{Ri+1}$
estimate $W_B, f_B, \tilde{T}_{i+1}$
$I = J = k = c = e = x = 1;$
while$(I \leq L)$ do
if$(V[I] \leq \tilde{T}_{i+1})$ then    $A[J] = V[I];$  $J+=1;$  $I+=1;$
end;
if$(\tilde{T}_{i+1} = V_{Ri})$ then $V_{Ri+1} = V_{Ri}$ ; else
while $(k < J)$ do
$B_{i+1}^{expected} = B_i + (\frac{\hat{T}_{i+1} - A[J]}{\hat{T}_{i+1}}) \times \tau;$  $C[k] = B_{i+1}^{expected}$; $k+=1;$ end;
 while$(c < k)$     do
$D[c] = |C[c] - B_{aim}|;$   $c+=1;$   end;
$Y = D[x];$
while$(e < c)$  do
if$(D[e] < Y)$  then $Y = D[e]; x = e;$ $e+=1;$  end;
$V_{Ri+1} = V[x];$
if $\left(\frac{B_L + B_{aim}}{2}\right) < Y < \left(\frac{B_H + B_{aim}}{2}\right)$ then  $V_{Ri+1} = V_{Ri};$
else $V_{Ri+1} = V_{Ri+1};$
```

III. Simulation Results

The proposed adaptive algorithm is evaluated using ns-3 network simulator and the topology implemented in this work is shown in Figure 2 below. The topology consists of an HTTP server, HTTP clients and a network element. To achieve adaptive streaming, the HTTP server offers the client seven levels of representations to adapt the video rates these are V={ 356, 500, 800, 1200, 1500, 2400 and 4500Kbit/s}. The length of video segment and video buffer is chosen to be 4s and 40s respectively. The values of other parameters B_L, B_H are 15s, 35s respectively. The bandwidth traces is varied according to network traces [15]. The values for *s*, S_b, f_B^L, f_B^H and ρ are 0.4,10, 0.5,1.5 and 0.8 respectively. The proposed algorithm is compared with other algorithms such as: the smooth video adaptation algorithm (SVAA) [16] and Buffer-Based Approach algorithm [4].

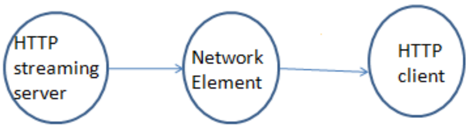

FIGURE II. NETWORK TOPOLOGY FOR SIMULATION

Figure 3 explains the video bitrate achieved by each of the rate adaptation algorithms, while Figure 4 indicates the corresponding client's buffer occupancy under a wireless network environment. In Figures 3 and 4, the aim of these algorithms (proposed , SVAA , BBA) is to improve the quality of experience (QoE) by minimizing the oscillations between the highest and lowest video qualities since the oscillation would increase the number of interruptions during the video playback. From these figures, SVAA has low number of bitrates switches with low average downloaded bit rates, because it follows a conservative approach in adapting the video bitrates under time varying condition in order to keep the client buffer occupancy away from underflow level. For BBA algorithm, the number of bitrates switches is moderate since the algorithm depends on the current playback buffer level in making the decision of rate adaptation. Finally, for our proposed algorithm, the number of quality switches is less than BBA with high average video bitrates because the bitrate adaptation process of the proposed algorithm is done according to the estimation of the available throughput and the current buffer occupancy. Also from Figure 4, the buffer level is kept away from underflow/overflow level because the selection for next bitrate is made according to the target level hence maintaining high video quality.

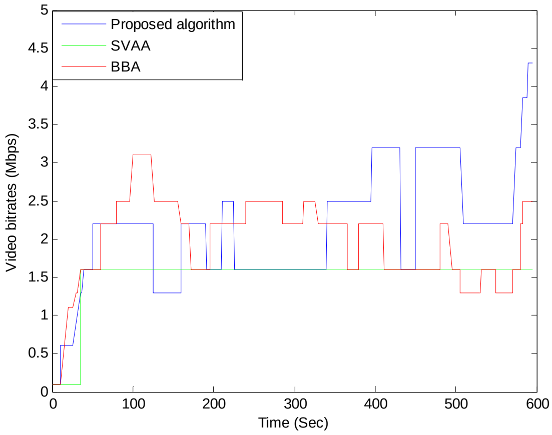

FIGURE III. VIDEO BITRATES VS. TIME

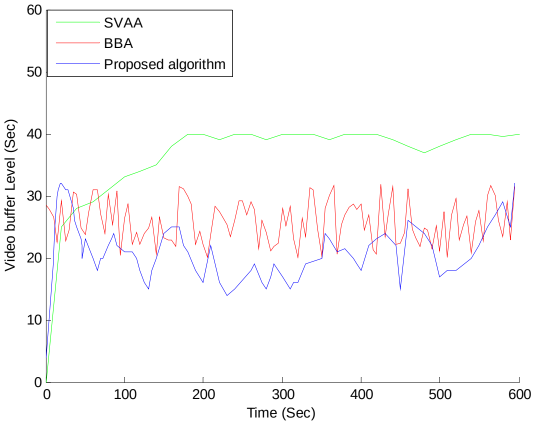

FIGURE IV. VIDEO BUFFER LEVEL VS. TIME

Figures 5 and 6 draw the cumulative distribution function (CDF) for the algorithms with different QoE metrics. In Figure 5, SVAA achieves low average bitrates because it selects lowest video quality compare with BBA and the proposed algorithm. With changing the buffer size, all algorithms tend to select the higher bitrates when minimize the video buffer size for many reason BBA select the higher bitrate to download the next segments because the buffer be full faster and hence the client starts to request higher video quality while the proposed algorithm tends to select higher bitrates because the scaling factor for buffer occupancy is activated earlier as the buffer occupancy increases. Figure 6 plots the CDF of number of interrupts encountered by different algorithms. This figure observes that doubling the buffer size from 60 sec to 120 sec helps increasing the number of interrupt free sessions from 85% to 95% for BBA, from 88% to 96.5% for the proposed and from 90% to 97% for SVAA. Increasing the buffer size also mitigates the maximum number of interrupts for all algorithms to 5 interrupts from 8, 13, and 17 interrupts for SVAA, proposed algorithm, and BBA, respectively.

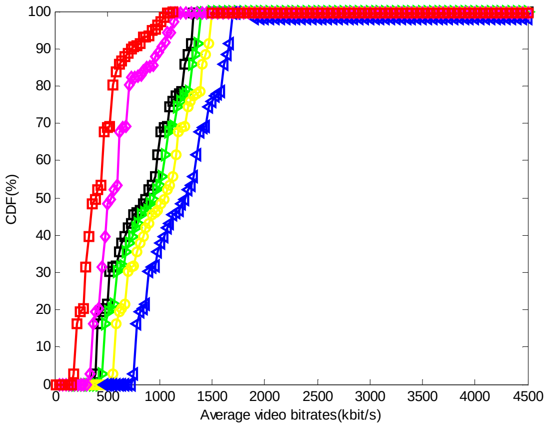

FIGURE V. THE CUMULATIVE DISTRIBUTION FUNCTION OF AVERAGE VIDEO BITRATES

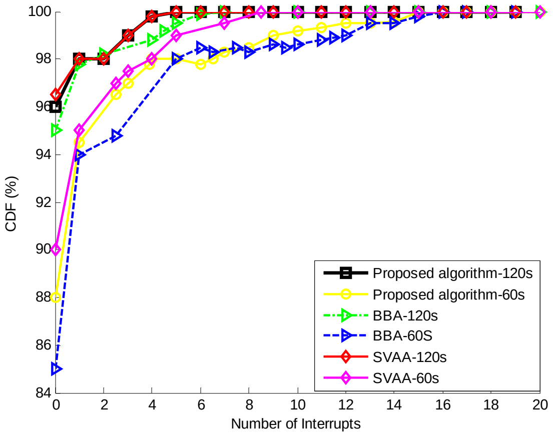

FIGURE VI. THE CUMULATIVE DISTRIBUTION FUNCTION FOR THE NUMBER OF INTERRUPTS

IV. CONCLUSION

In this work, adaptive algorithm for obtaining a smooth video bitrate and improve the QoE relies on estimating the available network throughput and current buffer occupancy. The simulation shows that the proposed algorithm outperform other algorithms in avoiding buffer underflow/overflow, high average bitrates and minimum number of interruptions due to use the scaling factor for buffer occupancy in that changing the buffer size effects on the video quality level i.e. minimizes the buffer size let's all the compared algorithms pick the high bitrates for avoiding the risk of playback interruption while increases the size improving the video quality with free number of interruptions.

REFERENCES

[1] N. Pengpeng, R. Eg, A. Eichhorn, C. Griwodz, and P. Halvorsen. "Flicker effects in adaptive video streaming to handheld devices." In Proceedings of the 19th ACM international conference on Multimedia, pp. 463-472. ACM, 2011.

[2] T. Stoekhammer. "Dynamic adaptive streaming over HTTP-design principles and standards." In Proceedings of the Second Annual ACM Conference on Multimedia Systems, vol. 2014, pp. 2-4. New York, USA: ACM, 2011.

[3] M. Ricky, X. Luo, E. Chan, and R. Chang. "QDASH: a QoE-aware DASH system." In Proceedings of the 3rd Multimedia Systems Conference, pp. 11-22. ACM, 2012.

[4] H. Te-Yuan, R. Johari, N. McKeown, M. Trunnell, and M. Watson. "A buffer-based approach to rate adaptation: Evidence from a large video streaming service." ACM SIGCOMM Computer Communication Review 44, no. 4 (2015): 187-198.

[5] J. Junchen, V. Sekar, and H. Zhang. "Improving fairness, efficiency, and stability in http-based adaptive video streaming with festive." IEEE/ACM Transactions on Networking (TON) 22, no. 1 (2014): 326-340.

[6] S. Yi, X. Yin, J. Jiang, V. Sekar, F. Lin, N. Wang, T. Liu, and B. Sinopoli. "Cs2p: Improving video bitrate selection and adaptation with data-driven throughput prediction." In Proceedings of the 2016 conference on ACM SIGCOMM 2016 Conference, pp. 272-285. ACM, 2016.

[7] D. Luca, V. Caldaralo, V. Palmisano, and S. Mascolo. "Elastic: a client-side controller for dynamic adaptive streaming over http (dash)." In Packet Video Workshop (PV), 2013 20th International, pp. 1-8. IEEE, 2013.

[8] X. Renchao, F. Richard Yu, T. Huang, J. Liu, and Y. Liu. "Joint user association and rate allocation for HTTP adaptive streaming in heterogeneous cellular networks." In Communications (ICC), 2016 IEEE International Conference on, pp. 1-6. IEEE, 2016.

[9] X. Xiufeng, X. Zhang, S. Kumar, and L. Erran Li. "pistream: Physical layer informed adaptive video streaming over lte." In Proceedings of the 21st Annual International Conference on Mobile Computing and Networking, pp. 413-425. ACM, 2015.

[10] B. Ayub, M. Hassan, and S. Kanhere. "Http-based adaptive streaming for mobile clients using markov decision process." In Packet Video Workshop (PV), 2013 20th International, pp. 1-8. IEEE, 2013.

[11] A. Zahran, J. Quinlan, D. Raca, C. J. Sreenan, E. Halepovic, R.K. Sinha, R. Jana, and V. Gopalakrishnan. "Oscar: an optimized stall-cautious adaptive bitrate streaming algorithm for mobile networks." In Proceedings of the 8th International Workshop on Mobile Video, p. 2. ACM, 2016.

[12] Y. Xiaoqi, M. Bartulović, V. Sekar, and B. Sinopoli. "On the efficiency and fairness of multiplayer HTTP-based adaptive video streaming." In American Control Conference (ACC), 2017, pp. 4236-4241. IEEE, 2017.

[13] Y. Li, T. Tillo, and J. Xiao. "QoE-Driven Dynamic Adaptive Video Streaming Strategy With Future Information." IEEE Transactions on Broadcasting (2017).

[14] A. Saamer, L. Anantakrishnan, A. Begen, and C. Dovrolis. "What happens when HTTP adaptive streaming players compete for bandwidth?." In Proceedings of the 22nd international workshop on Network and Operating System Support for Digital Audio and Video, pp. 9-14. ACM, 2012.

[15] R. Haakon, P. Vigmostad, C. Griwodz, and P. Halvorsen. "Commute path bandwidth traces from 3G networks: analysis and applications." In Proceedings of the 4th ACM Multimedia Systems Conference, pp. 114-118. ACM, 2013.

[16] T. Guibin, and Y. Liu. "Towards agile and smooth video adaptation in dynamic HTTP streaming." In Proceedings of the 8th international conference on Emerging networking experiments and technologies, pp. 109-120. ACM, 2012