Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

Flexible QoE optimized Video Adaptive Streaming over HTTP for sudden bandwidth drops

NGUYEN VIET HUNG[1,3], TRINH DAC CHIEN[2], PHAM NGOC SON[3], PHAM NGOC NAM[2] (Member, IEEE), AND TRUONG THU HUONG[3](Member, IEEE)

[1]Faculty of Information Technology, East Asia University of Technology, 100000 Hanoi, Vietnam (e-mail: hungnv@eaut.edu.vn)

[2]College of Engineering and Computer Science, VinUniversity, 100000 Hanoi, Vietnam (e-mail: nam.pn@vinuni.edu.vn)

[3]School of Electrical and Electronic Engineering, Hanoi University of Science and Technology, 100000 Hanoi, Vietnam (e-mail: huong.truongthu@hust.edu.vn)

Corresponding author: Pham Ngoc Nam, Truong Thu Huong (e-mail: nam.pn@vinuni.edu.vn, huong.truongthu@hust.edu.vn).

ABSTRACT We have observed a boom in video streaming over the Internet, especially during the Covid-19 pandemic, that could exceed the network resource availability. In addition to upgrading the network infrastructure, finding a way to smartly adapt the streaming system to the network and users' conditions to satisfy clients' perceptions is exceptionally critical, too. This paper proposes a new QoE-aware adaptive streaming scheme over HTTP - ABRA - to make flexible adaptations based on the network and the client's current status. Besides, we propose a technique that can keep the buffer at an average high for more than 10s. We were limiting the phenomena of rebuffering due to unexpected and unpredictable bandwidth changes. The algorithm keeps the quality of subsequent versions at a constant level even when the average bitrate decreases, increasing the QoE. Experimental results show that our method can improve QoE from 7.86% to 20.41% compared to state-of-the-art methods. ABRA can provide better performance in terms of QoE score in all buffer conditions compared to the existing solutions while maintaining a minimum secured buffer level for the worst case.

INDEX TERMS Video adaptive streaming, HTTP, Quality of Experience

I. INTRODUCTION

Online video nowadays has been growing rapidly over the Internet, at the moment accounting for 79% of the whole Internet traffic, as reported by Cisco's statistics [1]. Especially, during the Covid 19 pandemic, online video sessions such as virtual classrooms/meetings have become essential to connect people around the world and keep our society go on. Therefore, video services have been much enhanced and developed around the world in recent years. Video is a big challenge as it contains much content transmitted with limited network conditions.

Recently, we have observeed HTTP Adaptive Streaming (HAS) become one of the most common protocols for video streaming in which HTTP is an abbreviation for HyperText Transfer Protocol. In the HAS technique, at first, a video is encoded into many different versions with different video qualities. Each of those versions, then, is divided into smaller units called *Segment*. *Segment* is created and stored at the back-end server. A suitable segment with a specific video version will be sent to a client upon the client's request that is decided based on network conditions. Controlling the streaming system by adjusting segment quality that way can cause a severe quality variation during a streaming session if network bandwidth fluctuates strongly during the session. In turn, users who are watching those streaming sessions may perceive the overall service negatively (i.e. bad Quality of Experience). Therefore, a smart way to maneuvering such an online video system should be based on a QoE model to decide which version should the system adapt to. Applying a QoE measured by its own users, the system attempts to reach the highest possible QoE score to users.

To give the most general adaptive algorithm for different cases of network conditions. We use the QoE model in instance selection to adapt to existing client conditions. Also, client conditions are partitioned based on buffer and instantaneous throughput so that each client machine makes its own different decisions in the most appropriate way.

Recently, although there have been a numerous researches proposed for adaptive streaming such as [2]–[5], they just select segment versions heuristically. The version selection process depends on the current condition of the client's buffer status and network bandwidth and [2], [3], [5]. To the best of our knowledge, employing a real QoE model to decide adaptation version is introduced the first time at research [4]. But, this scheme only uses 2 segments to make decision at a given time. Therefore, the scheme tends to use up all available resource for those 2 segments. In turn, this phenomenon reduces the flexibility of the solution to adapt with the fluctuations of bandwidth for both of the future and the current segments. That situation can get worse in case bandwidth suddenly drops down.

In addition, another strategy based on Scalable Video Coding (SVC) technique to improve the adaptability of HAS is mentioned in [22]. In [22], they took two steps of loading the segment and then smoothing it to increase the quality of the user experience. The above method gives a significant increase in QoE results but proves to be complicated and difficult to implement when it has to incorporate Finite State Machine. It is found out that this algorithm complexity is relatively large, of about 167ms to generate a decision on a 2.5 GHz computing core. The main reason for such high algorithmic complexity is that the system takes a considerable amount of time to smooth after downloading.

To solve the problems of the existing algorithms, the number of next adaptation segments which are selected versions is decided based on the reduction in measured bandwidth of the four previous segments. In particular, if only considering one next segment, the selected version tends to be the highest possible. This leads to buffer level drop-down, causing significant version degradation in long time later.

Based on that fact, in this paper, we propose a rate adaptation strategy on the client-side to improve QoE perceived by users (namely ABRA). This work is also considered the improved version of the adaptation algorithm previously proposed by work [21]. In ABRA, the client's buffer was divided into three levels, and the change in throughput measured from the client-side was divided into two different cases. We thus obtain six different combinations of throughput and buffer. We propose five different solutions to solve the above six cases. Five strategies include lowering the version to the lowest quality level to optimize for the buffer, keeping the version at the same quality level as the previous version to reduce the negative impact of the quality change on the QoE perceived by the user, estimating version for two or three segments in the future based on the effect of a combination of consecutive versions on the QoE value. ABRA solves the problem of considering only one next segment, leading to select the highest possible version, then resulting in buffer level drop-down, causing significant version degradation in long time later.

The remaining of our paper is structured as follows. We will give a review of state-of-the-art in Section II. Then the detail of our proposed adaptive streaming algorithm called ARBA will be elaborated in Section III. The performance results of ARBA obtained from multiple experiments and aspects are discussed in Section IV. And finally, our conclusions are given in Section V.

II. RELATED WORK

Recently, there have been many proposed adaptation algorithms for improving service Quality perceived by clients (i.e. Quality of Experience - QoE. In fact, it is hard to find a clear difference between those solutions, however we could categorize them into 3 main directions: buffer-based, throughput-based, and mixed (i.e. hybrid of buffer and throughput-based) algorithms [6]. Mixed Adaptation combines the external (bandwidth) and internal (buffer, size of segment...) elements of the client to compute the bitrate of the next segment.

In the throughput-based methods, at the client side throughput for the next segment is estimated based on the condition of the previously monitored throughput, which can be computed as the size of the previous downloaded segment being divided by the time required to get it. Finally, based on that estimated throughput, the most appropriate version for the next segment will be chosen. One of the initial studies in the throughput-based direction is solution *Aggressive* [2] which has a very simple principle. In *Aggressive*, throughput is simply estimated as equal to the throughput of the previous segment. Then, the scheme selects the video version with the quality as high as possible, in order to ensure that the bitrate of that version is not higher than the estimated throughput. This is to avoid re-buffering. However, estimating that way is often inaccurate in case the network bandwidth fluctuates strongly. Moreover, *Aggressive* is observed quite sensitive to bandwidth variation. This bandwidth fluctuation intolerance results in strong quality variation, and badly influencing QoE perceived by clients. To solve this challenge, some enhanced solutions are proposed later like [7], [8], which make use of a safety margin in the throughput estimation; or like work [9] which uses the average throughput calculated from multiple previous segments to compute the estimated throughput. In work [14], the authors address to optimize experience of viewers based on a receiver-driven approach subject to changing throughput of a TCP flow. This approach always chooses the lowest representation for the first segment, resulting in the disadvantage that a few first seconds of a video are always downloaded at lowest quality. However, most of the solutions based on throughput knowledge use either the harmonic-mean network capacity estimation or the moving average models. Those models do not capture the time relevance of diverse samples and may not capture the numerous network bandwidth variations accurately.

In the direction of buffer-based schemes, the current and previous buffer statuses are the primary factor to decide the video version for the next segment, as found out in [10]–[12]. For this type of solution, at the client side, the playout buffer is typically divided into multiple ranges. Within each range, a suitable version can be determined by multiple different actions. In general, when the buffer is in a very good condition (i.e. in a high buffer range), the version for the next segment shall be chosen higher than the version of the current segment. But when the buffer stays in the middle range, those schemes prone to keep the version stable. On the contrary, when the buffer stays at the low level, then the version for the next segment will be decreased to the lowest level for avoiding the re-buffering phenomenon in the system. In [10], the authors proposed to consider buffer conditions only for video streaming adaptation in future, provided that capacity estimation is needed. In [11], a buffer-based adaptation logic coordinating with client metrics was proposed to compensate for error in decisions of video adaptation. These errors are generated due to the fact that available network information at clients is insufficient, especially in the context of multiple clients competing through a bottleneck. The authors in work [12] proposed BOLA, that utilizes a Lyapunov optimisation model to consider the buffer occupancy observations only. BOLA achieves near-optimal utility and in many cases significantly higher utility than state of the art such as: MPC, PANDA, ELASTIC and Pensieve. But if the selected bit rate does not match the available bandwidth, BOLA takes long to until convergence. The issue of in-optimized parameters pending in [12] was then solved by work Oboe [36] which overcame the limitations of BOLA by using buffer level to estimate capacity. Research [37] indicated that estimating capacity is not necessary at the steady state; but quite important during the startup phase because buffer grows from empty. So the solution in [37] - BB - decides video rates based on the current buffer occupancy. It applies simple capacity estimation only when the buffer has grown from empty. By doing that work [37] can reduce the re-buffering rate by 10–20 % in comparison with the default ABR algorithm of Netflix, while achieving higher video rates in steady state. However, this solution, BB, becomes unsuitable when the video quality changes continuously. BB tends to generate a large number of version switches that badly affect on the user's quality experience.

For the final category, the mixed (or hybrid) algorithms, every decision made by a client is based on both of the throughput status and buffer occupancy statuses, as well as other parameters such as segment sizes and the QoE perceived by users. The mixed algorithms will take advantages of both buffer-based and throughput-based schemes such as: throughput based scheme helps to choose good bit rates to increase video quality, and the buffer-based scheme helps to adapt to good bit rates to avoid re-buffering. In fact, most of the throughput-based schemes fail to capture the time relevance belonging to different samples; and those methods perhaps do not capture variations in network bandwidth accurately. While, a pure buffer-based strategy could take a long time to converge unless the available system bandwidth matches the selected version. So a hybrid method can take advantage of the strong points, and overcoming the disadvantages of the throughput and buffer-based schemes. In the direction of mixed algorithms, several work can be found in [15], [16], [24], [26], [30]. However, these solutions do not use the QoE-Model for adaptation decisions. Work [15] consider the degradation of DASH performance caused by the rate control loops of DASH and TCP and propose SQUAD to deal with the issue. SQUAD solves the discrepancies of DASH bandwidth estimation at the application layer and rate estimation of the underlying transport protocol. Research [16] introduces a new approach for Adaptation Buffer Management Algorithm, called ABMA+. In principle, ABMA+ make adaptation decisions based on predicted re-buffering probability provided a buffer map is pre-computed in order to avoid heavy computating on the fly. One of the popular approaches to ABR is fuzzy-based Algorithms in [24], [26]. Akshan et al.in work [26] used the moving average of the playback buffer level variations and observed throughput in order to minimize the video rate switches. Since the existing ABR algorithms use fixed control laws and are designed with predefined client/server settings [24], those solutions fail to reach optimal performance for a different cases of video client settings and QoE objectives. In work [24], the authors solved the above problem by proposing a buffer and segment-aware fuzzy-based ABR algorithm that chooses rates for upcoming video fragments, based on segment duration and the client's buffer size in addition to throughput and playback buffer level. The ARBITER+ [30] was proposed employing a combination of a proportional integral controller and a harmonic network throughput estimator to determine the next representation quality. In this category, MPC [29] uses predictive model control, combining buffer occupancy and throughput information. This algorithm proposed to optimize a comprehensive set of metrics. Bitrates for the current segment are chosen based on network bandwidth prediction for the next few segments. Hence it is obvious that the prediction accuracy has huge impact on the performance of MPC. Besides, MPC also requires to compute optimization offline and outside of a client for an exhaustive set of contexts. Similar to solution BB, although MPC can reach quite high average bitrate quality reaches, this solution is unsuitable when the video quality changes continuously. MPC tends to causes more stallings in that case.

Also, in the direction of concerning both throughput and buffer conditions to make adaption decision, we can find a subgroup of using Learning-based algorithms to solve the issue. However, this is another direction different from the QoE-model based approaches. Another approach also uses QoE in adaptive algorithms like our paper, but with a different solution when QoE is used as a value function of the Reinforcement learning (RL) process to improve the quality of traditional algorithms in [31], [32]. In [31], the authors proposed Pensieve using reinforcement learning (RL) for making ABR decisions. The scheme utilizes a neural network to selects bit rate for next video chunks based on observations of performance at the players by the past decisions. In work [32] the authors presented the QoE-oriented DASH framework in which an RL-based ABR algorithm is embedded. This scheme achieves better visual and temporal QoE factors while ensuring fairness at the application-level among multiple clients competing through a bottleneck. Besides, HotDASH in [34] is also another method that uses reinforcement learning to improve QoE, bit rate by prefetching

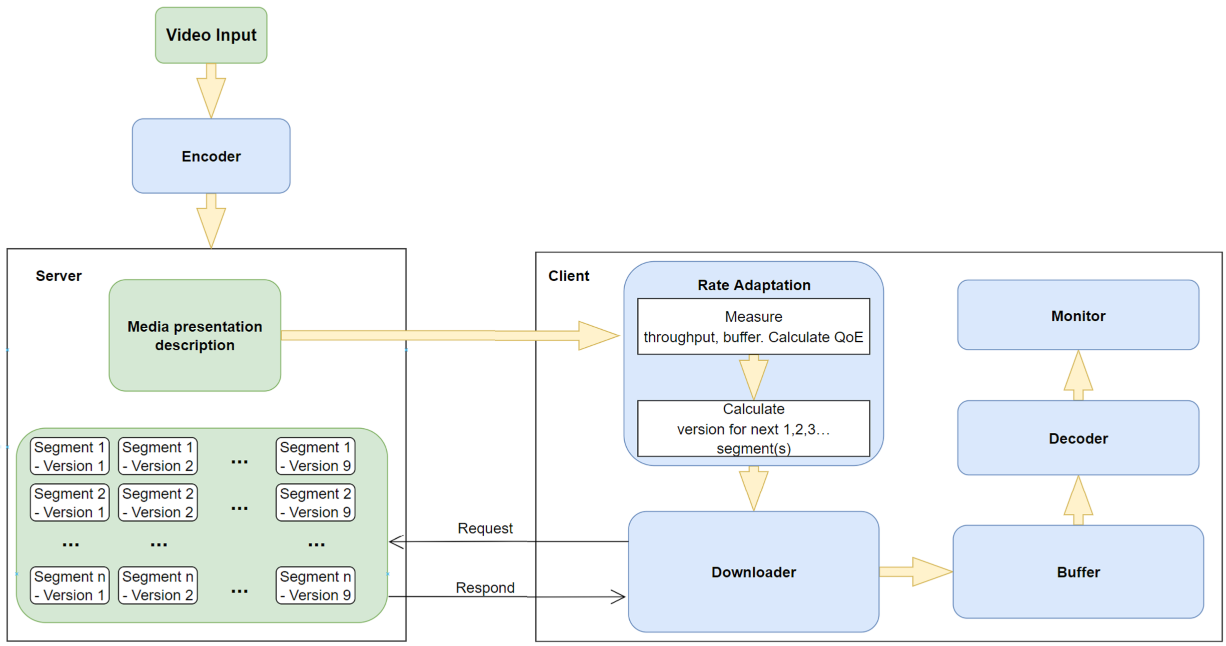


FIGURE 1: Process of Content Preparation at the Streaming Server and Client.

video segments.

Some other adaptive algorithms that use a combination of throughput and buffer with non-QoE parameters are mentioned in [23], [25], [29], [35]. Besides, the authors in work [35] presented a hybrid algorithm named DYNAMIC built on the DASH reference player. In this scheme, BOLA is used when buffer is high as a buffer-based control manner; and a throughput rule is used when buffer is low or empty as a throughput-based manner. Work [23] considers buffer level and level variations to mitigate playback interruption based on the Fuzzy-based DASH adaptation algorithms.

From another side, in the direction of mixed algorithms that take into account the QoE model, we can find several work such as [13] and researches [3]–[5], [21]. The authors in work [13] proposes to use game theory to allocate resource to improve QoE for multiple users.

Research [3] provides *SARA* - an adaptation algorithm that uses the buffer status, the estimated throughput, segment sizes to select the version of the next segment. Based on those metrics, the the most appropriate-size version for the current state at a client will be chosen. But, strong network bandwidth fluctuation can cause selected versions to change frequently, resulting in degradation in viewers' service perception (QoE degradation). Work [5] propses *SATE* which applies a QoE model for better decision. However, both [5] and *SARA* only estimate a version for one next segment, leading to optimization for an instant time but not for the whole streaming session. As the remedy, work [4] proposes an adaptation algorithm that selects versions suitable for the next two segments. However, fixing estimation for 2 segments makes work [4] not work quite well in the case there is a sharp bandwidth drop. Work [21] considers a new adaptive streaming algorithm based on the throughput status, buffer level, and the QoE perceived by users. Therefore, to obtain more stable and high versions and so the QoE, the proposed algorithm took more next segments into account. In comparison with considering three next segments, the decision taking into account two next segments generally gets higher selected versions but less stable. Therefore, when throughput varies sharply, the proposed algorithm considers three next segments in making adaptation decisions to ensure the stable QoE to users. Meanwhile, in the case of steady bandwidths, only two next segments are taken into account. This proposed solution is considered medium-buffer adaptation algorithm. It means that solution is not totally effective in high or low buffer conditions.

To deal with the issue, we propose an upgrade version that can work well in all buffer sizes, which is called ABRA (i.e **A**ll **B**uffer **R**ange **A**daption). In the same throughput conditions, the ABRA algorithm only slightly reduces video bitrates but increases QoE scores compared with the MBA algorithm by 10% and reduces the number of stallings by 3 to 4 times.

III. PROPOSED ADAPTIVE STREAMING ALGORITHM - ARBA

A. SYSTEM ARCHITECTURE

In this part, the overall adaptation architecture between Server and Client. is illustrated in Figure 1.

- At the Server: video is encoded and segmented into

segments of the same length in time, each of which has multiple versions of different quality. Information of components is stored in the Media presentation description file.
- At the Client: the server will send the MPD file to the Client. Based on information obtained from MPD and estimated data from the Client (i.e. throughput, buffer, QoE,...). After that, Rate Adaptation will choose the version for the next 1, 2, or 3 segment(s). The downloader will then request the segments and downloading them to the Client. The component is buffered and then decoded and broadcast to users' screen.

B. ABRA ALGORITHMS

In this section, we will elaborate an adaptation algorithm which is designed to work appropriately with all ranges of buffer (i.e. low, medium, high buffer size). Due to that purpose, the scheme is called All-Range-Buffer Adaptation (ARBA). The ARBA scheme is described as follows:

Assumption:
- ϕ seconds: length of each segment
- N: N encoded versions of different bitrates for each segment in which a better video quality corresponds to a higher video quality version.
- At the client, downloaded segments are placed on the playback buffer to wait for its play time.

To decide appropriate versions for the segments, we divide the buffer into three ranges: dangerous, low, and high, based on 3 determined thresholds of B_{min}, B_{low}, B_{high}, as described in Fig.2. These thresholds are defined by the video duration which is counted by the number of seconds contained in the buffer.

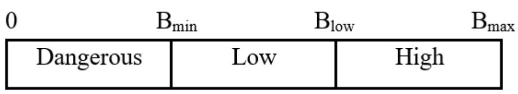


FIGURE 2: Three divided buffer ranges

To make good adaptation decisions in the condition throughput fluctuates, our algorithm ABRA differentiate 2 main variation cases: *downtrend case* and *uptrend case*. The downtrend case is considered when the measured throughput of the previous segment is equal to or greater than the current throughput. Otherwise, it is considered the uptrend case.

In ABRA, we also consider 2 other special cases of throughput called: *Throughput sharp drop* and *Throughput rapid rise*. These 2 conditions are considered based on specific buffer statuses as well.

The goal of ABRA is to select the appropriate versions for next segments based on each specific throughput case and buffer level, in order to maximize overall QoE score of streaming sessions. Each proper decision should be made based on the trade-off between decreasing buffer occupancy and increasing segment versions for avoiding interruptions in playback (or re-buffering events).

- At a specific time, version V_{i+1} is selected for next segment $i+1$ based on the fact that a client has to to capture current buffer B_i^{cur} as well as throughput T_i.
- Later, for each version N that satisfies the condition $N \geq V_{i+1} \geq 1$, the corresponding estimated buffer level $B^e_{i+1,V_{i+1}}$ and throughput T^e and can be calculated.

The corresponding throughput T^e is calculated as follows:

$$T^e = T_i \times (1 - margin) \quad (1)$$

where:
- $margin$: a parameter to reduce the bad influence of throughput estimation errors.

The corresponding buffer level $B^e_{i+1,V_{i+1}}$ is calculated as follows:

$$B^e_{i+1,V_{i+1}} = B_i^{cur} + \phi - \phi \times \frac{R_{i+1,V_{i+1}}}{T^e} \quad . \quad (2)$$

where:
- $R_{i+1,V_{i+1}}$: the bitrate of version V_{i+1} estimated for segment $i+1$.
- $\phi \times \frac{R_{i+1,V_{i+1}}}{T^e}$: the amount of time to download version V_{i+1} for segment $i+1$ completely

In addition, in this paper, we use the QoE model proposed in [17] to calculate the QoE score corresponding to version V_{i+1}. The calculation is based on its quality level $Q_{V_{i+1}}$. This QoE model contains almost all parameters that affect QoE when streaming video via http protocol including: different quality values, quality switching types, and interruptions.

$$QoE_{pred} = Q_{PQ} - D_{IR} - D_{ID} \quad (3)$$

Where:

- QoE_{pred}: overall QoE considering the influence of initial delay, interruptions and varying perceptual quality
- Q_{PQ}: varying perceptual quality of a session, depending on the corresponding quality switching and quality value.
- D_{IR}: distortion function of the interruptions
- D_{ID}: distortion function of the initial delay

This QoE model is found to be capable of predicting QoE perceived by users accurately, from the beginning to any moment during the whole course of a streaming session. Finally, ABRA calculates appropriate versions for next segments based on buffer levels, throughput variations and corresponding QoE score of segment versions. In the ABRA algorithm, QoE scores are continuously measured in every playing video second. However, any existing QoE model can be actually applied after reviewing and investigating the performance and accuracy of those proposals carefully.

This solution is proved to work well in all buffer sizes from low to medium to high buffer. Therefore, we name

Algorithm 1: Calculate 3 next segments

Initiate: $V_{i+1} \leftarrow 1, V_{i+2} \leftarrow 1, V_{i+3} \leftarrow 1,\ QoE^{max} = 0$
for $v_1 \leftarrow 1, 2, ..., V_i$ **do**
  for $v_2 \leftarrow 1, 2, ..., v_1$ **do**
    for $v_3 \leftarrow 1, 2, ..., v_1$ **do**
      Compute B^e_{i+1,v_1} and B^e_{i+2,v_2} and B^e_{i+3,v_3} by (2), (4) and (5)
      Compute the overall quality QoE_{i+3} by the QoE model proposed in [17]
      if $\{(QoE_{i+3} > QoE^{max})$ *and* $(B^e_{i+1,V_{i+1}} > B_{min})$ *and* $(B^e_{i+2,V_{i+2}} > B_{min} + \Delta B_{err})$ *and* $(B^e_{i+3,V_{i+3}} > B_{min} + \Delta B_{err})\}$ **then**
        $QoE^{max} \leftarrow QoE_{i+3}$
        $V_{i+1} \leftarrow v_1, V_{i+2} \leftarrow v_2$ and $V_{i+3} \leftarrow v_3$
      end
    end
  end
end

the algorithm - All-Buffer-Range-Adaptation ABRA. ABRA is an enhanced version of work [21]. ABRA flexibly calculates adapted versions either for the next 2 segments or 3 segments. In case throughput decreases strongly, ABRA calculate adapted versions for next 3 segments, else for next 2 segments. If work [21] focuses more to find a solution for a medium buffer condition, still having a disadvantage of not working very well in the low and high buffer condition. With this ABRA, when buffer is low and throughput increases strongly, ABRA keeps the same version. When buffer is high and throughput decreases strongly, ABRA calculates new adapted versions for the next 3 segments. With this strategy, ABRA can work quite well in 3 ranges of buffer: low - medium - high. In comparison with our previous work [21], ABRA is proved to outperform at the low and high buffer conditions.

In ABRA, the algorithm runs with the input of selecting 1, 2 or 3 next segments to predict versions. when the next number of segments to be calculated is 1, the highest quality level version available is selected. However, this may reduce the stability of the quality of subsequent versions. To overcome this, we consider the next 2 or 3 segments. As a result, the maximum version quality is limited then the version selection is more stable. As the number of segments under consideration increases, the quality of subsequent versions becomes more stable. We consider 1 segment for increasing in quality when throughput increases, 3 segments in the case of optimal stability (e.g. a sharp drop in bandwidth..), and 2 segments in the remaining cases. This way of deciding 2 or 3 segments to make prediction helps the network resource to be used more effectively and flexibly. The resource utilization is, therefore, more efficient than the method of of considering only 1 segment.

Below is a description of how to choose version when the system considers the next 3 segments:

- select 3 next versions $\{V_{i+1}, V_{i+2}, V_{i+3} | 1 \leq V_{i+3} \leq V_{i+2} \leq V_{i+1} \leq V_i\}$
- the estimated QoE for segment 3 - QoE_{i+3} - is greater

Algorithm 2: All Buffer Range Adaptation - ABRA

if $(T_i \leq T_{i-1})$ *//Down trend case* **then**
  if $B^{cur}_i \leq B_{min}$ *//in dangerous range* **then**
    $V_{i+1} \leftarrow 1$ *// switch to the lowest*
  end
  if V_{i+1} *was decided* ***and*** $|B^{cur}_i - B^e_{i,V_i}| \leq \Delta B_{err}$ **then**
    Keep using V_{i+1} *//which is* V_{i+2} *in the previous decision*
  end
  if $B^{cur}_i > B_{low}$ *//in high or safe range* **then**
    ($V_{i+1} \leftarrow V_i$ *// keep the same version*)
    Select versions for 3 next segments by Algorithm 1
  end
  if $max(T_{i-1}, T_{i-2}, T_{i-3}) - T_i > \Delta T_{drop}$ *// sharp throughput drops* **then**
    Select versions for 3 next segments by Algorithm 1
  end
  else
    Initiate: $V_{i+1} \leftarrow 1, V_{i+2} \leftarrow 1, QoE^{max} = 0$
    for $v_1 \leftarrow 1, 2, ..., V_i$ **do**
      for $v_2 \leftarrow 1, 2, ..., v_1$ **do**
        Compute B^e_{i+1,v_1} and B^e_{i+2,v_2} by (2) and (4)
        Compute the overall quality QoE_{i+2} by (1)
        if $\{(QoE_{i+2} > QoE^{max})$ *and* $(B^e_{i+1,V_{i+1}} > B_{min})$, $(B^e_{i+2,V_{i+2}} > B_{min} + \Delta B_{err})\}$ **then**
          $QoE^{max} \leftarrow QoE_{i+2}$
          $V_{i+1} \leftarrow v_1$ and $V_{i+2} \leftarrow v_2$
        end
      end
    end
  end
end
else
  if $B^{cur}_i \leq B_{min}$ *// in dangerous range* **then**
    if $max(T_{i-1}, T_{i-2}, T_{i-3}) - T_i > \Delta T_{rise}$ **then**
      $V_{i+1} \leftarrow V_i$
    end
    else
      $V_{i+1} \leftarrow 1$
    end
  end
  else
    Initiate: $V_{i+1} \leftarrow V_i, QoE^{max} \leftarrow 0$
    for $v_1 \leftarrow V_i, V_i + 1, ..., N$ **do**
      Compute B^e_{i+1,v_1} by (2)
      Compute the overall quality QoE_{i+1} by (1)
      if $\{(QoE_{i+1} > QoE^{max})$ *and* $(B^e_{i+1,v_1} > B^{cur}_i)\}$ **then**
        $QoE^{max} \leftarrow QoE_{i+1}$
        $V_{i+1} \leftarrow v_1$
      end
    end
  end
end

than QoE^{max}

- the estimated buffer for segment 1, given the condition if version $i+1$ is chosen, is greater than B_{min}
- the estimated buffer for segment 2, given the condition if version $i+2$ is chosen, is greater than B_{min} plus ΔB_{err}. ΔB_{err} is the buffer margin taken into account to prevent deviation from the actual bandwidth and the estimated one. In our experiment this buffer margin is set 2 seconds.
- in the same way the estimated buffer for segment 3, given the condition if version $i+3$ is chosen, is greater than the mimimum buffer plus buffer margin ΔB_{err}.

Calculating for 1 or 2 segments is similar to the above case. However considering for 1 segment will be different in terms of the selected version as follows:

- Select 1 next version when $\{V_{i+1}|V_i \leq V_{i+1} \leq 9\}$ where 9 is the maximum version.

In the downtrend case where $T_i \leq T_{i-1}$, ABRA operates as follows:

- when the current buffer B_i^{cur} is in the dangerous range (i.e., $B_i^{cur} \leq B_{min}$), ABRA selects the lowest version to avoid interruptions (i.e., $V_{i+1} = 1$).
- If the current buffer is in the high range (i.e. $B_i^{cur} \geq B_{low}$), ABRA calculates for the next 3 segments with the goal of either reducing to a lower quality version if possible or remaining the video quality version.
- If the current buffer is in the low range (i.e. $B_{min} < B_i^{cur} < B_{low}$), ABRA predicts the version for either 2 or 3 segments, depending situations in bandwidth decrease. If bandwidth encounters a sharp drop, the prediction will cover for 3 segments, otherwise 2 segments.

Since the number of next selected segments is based on the variation on throughput in real time. Especially, if the network condition encounters a sharp drop in throughput, a decision on which versions should be used for the next 3 segments is made also based on the goal of how to keep the video quality mostly stable during a streaming session overall. Otherwise, users would badly perceive the service due to quality up-side-down all the time.

ABRA uses Algorithm.1 to calculate 3 next segments in order to make a suitable decision taking into account keeping the current version to have version stability or decreasing video quality version in case of bad bandwidth conditions.

Otherwise, version prediction for next 2 segments will be carried out based on the degree of throughput variation. To define this degree, we determine ΔT_{drop} - throughput difference threshold. Throughput degradation is considered to be sharp drop if the difference between the current throughput and the max throughput measured at the 3 previous segments is greater than this ΔT_{drop}. Essentially, the goal to select versions for next segments is to maximize QoE at the last adapted segment and to prevent buffer levels from dropping to the dangerous range. In ABRA, the estimated buffer level of segment $i+3$ and $i+2$ are calculated as follows:

$$B^e_{i+2,V_{i+2}} = B^e_{i+1,V_{i+1}} + \phi - \phi \times \frac{R_{i+2,V_{i+2}}}{T^e}. \quad (4)$$

and

$$B^e_{i+3,V_{i+3}} = B^e_{i+2,V_{i+2}} + \phi - \phi \times \frac{R_{i+3,V_{i+3}}}{T^e}. \quad (5)$$

C. UPTREND CASE

In the uptrend case, adaptation decisions are made based on different conditions as follows. If the buffer level B_i^{curr} falls within the dangerous range, similar to the downtrend case, video quality version will be switched to the lowest quality version. However, if throughput increases strongly back again (througput rapid rise), the version of the previous segment will be applied for this segment.

In another case, the highest possible version will be chosen to obtain the best QoE while causing no decrease in buffer level. The goal of ABRA is to assure the high buffer level over time, that improves the adaptability of ABRA in bad scenarios, especially in the case of sharp bandwidth drops. The summary of our proposed algorithm is presented in Algorithm 2.

IV. EXPERIMENTAL RESULTS

To evaluate the performance of the ABRA solution, at first, we will compare the MBA solution previously proposed in work [21] with the 4 cutting-edge solutions *Aggressive* [2], *SARA* [3], *Tran's* [4] and *SATE* [5]. Then, we will show the performance of MBA in comparison with ABRA as the enhanced version of MBA.

A. EXPERIMENTAL SET-UP

In our experiment, we set up a testbed that comprises of:

- A server and a client.
- The IP network in between the server and the client is emulated by the DummyNet tool in which throughput can be varied.

The buffer thresholds are set as follows:

- $B_{min} = 10s$, $B_{low} = 20s$, $B_{high} = 30s$, $B_{max} = 40s$.
- the $margin$ parameter is set to 0.2

Bandwidth fluctuation is emulated by using 2 trace files in [20]. At the server side, we use a 180-second long video extracted from the Big Buck Bunny film [18]. The video is partitioned into 2-seconds segments (i.e. ϕ = 2 seconds), each of which then is encoded into 9 different versions corresponding to 9 quantization parameters (QP) as illustrated in Table 1. The encoding process is done by using Variable Bitrate (VBR). These 9 versions of each segment are stored on the server, being ready for the adaptive streaming process. At the client's side, the adaptive streaming algorithms ABRA calculates and makes decision which suitable versions should be downloaded for each single segments, based on the buffer and network conditions as explained in the previous section.

TABLE 1: Definition of video quality versions

Version	QP	Average bitrate (Kbps)
9	24	6663.121
8	26	5214.973
7	28	4088.887
6	32	2546.112
5	36	1595.753
4	40	1001.490
3	44	646.894
2	48	432.716
1	52	327.070

As aforementioned, we apply the QoE model proposed by work [17] to evaluate the effectiveness of our ABRA algorithm versus the other existing solutions.

B. PERFORMANCE EVALUATION

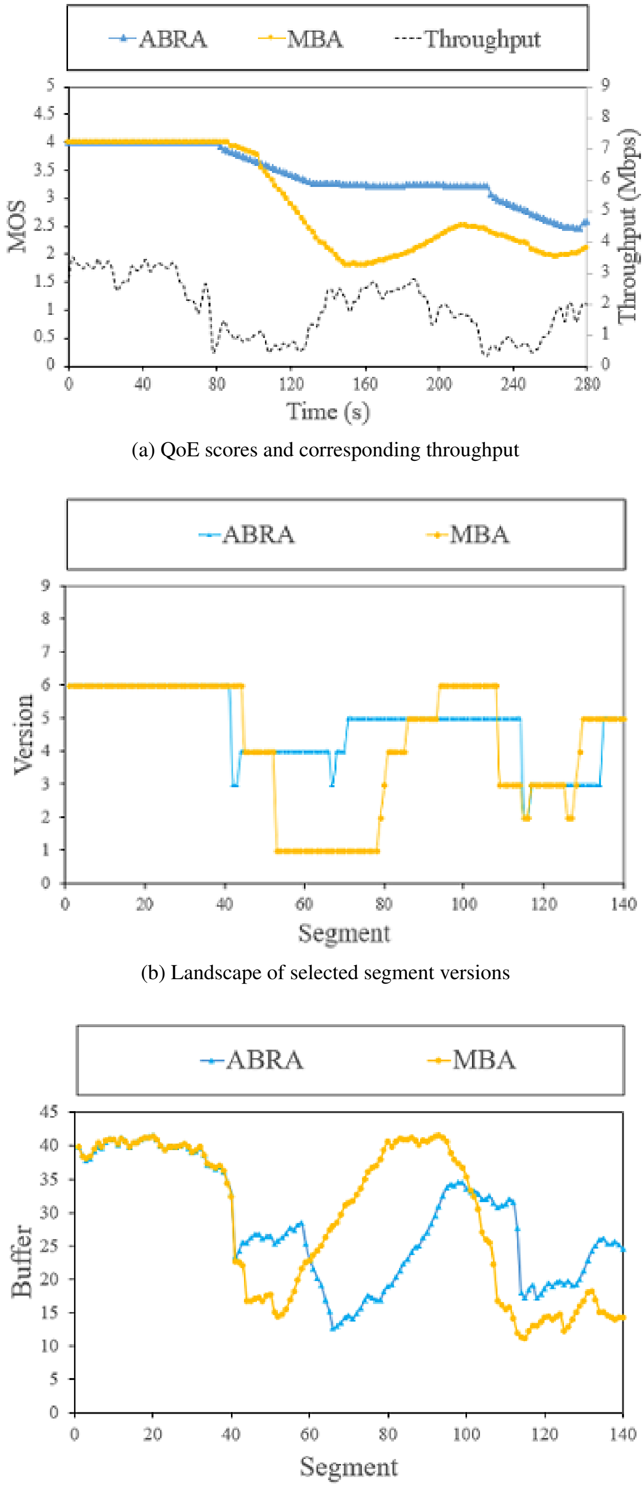
In this section, at first, we will compare the performance of the so-called MBA method (i.e., Medium-Buffer Adaptation algorithm) proposed by a recent work [21]. The MBA method was actually proven to out perform some state of the art researches such as *Aggressive* [2], *SARA* [3], *Tran's* [4] and *SATE* [5]. MBA can solve some problems such as low QoE score achievement during throughput fluctuation in *Aggressive* [2]; or buffer drop-down if bandwidth is not sufficient enough in *SARA* [3]; or significant degradation in QoE scores sometimes due to attempt to keep highest version in a long period of *SATE* [5] and *Tran's* [4].

In summary, MBA is able to provide better performance compared with the other 4 reference solutions in terms of QoE stability throughout the streaming session and highest achievement of the overall QoE score. MBA earns those benefits due to the fact that it selects the number of segments flexibly while maintaining a minimum secured level of buffer for the worst case. Moreover, since determination on the number of predicted segments should be for the sake of a good QoE, versions are finally selected evenly at close intervals, that in turn creates a smooth video with high QoE score (i.e. high perception by users).

As the upgraded version of MBA, ABRA inherits all the advantages of MBA while improving the performance in all ranges of buffer level. In this section, the performance of ABRA is evaluated by directly comparing with MBA in the following aspects:

- (1) QoE perceived by users,
- (2) client's buffer while streaming,
- and (3) the selected version in full session.

Note that our play-out session is assumed to start after the buffer is full. We also use other metrics to test the performance of our live streaming algorithms including: average received quality rate (rav) in Kbps, the number of freeze-free sessions (nff), the number of stalls (nf), the total stall duration (tf) in seconds, the number of switches (nsw), and

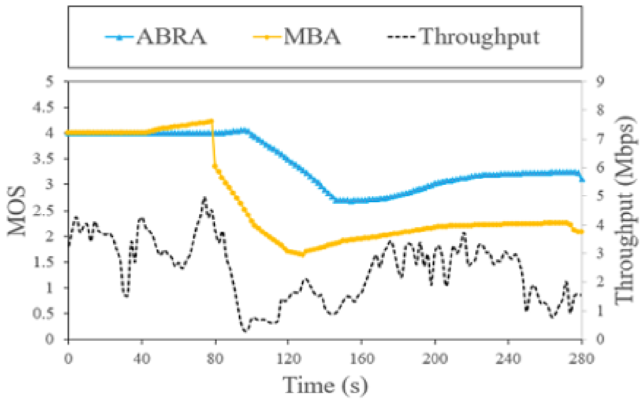


(a) QoE scores and corresponding throughput

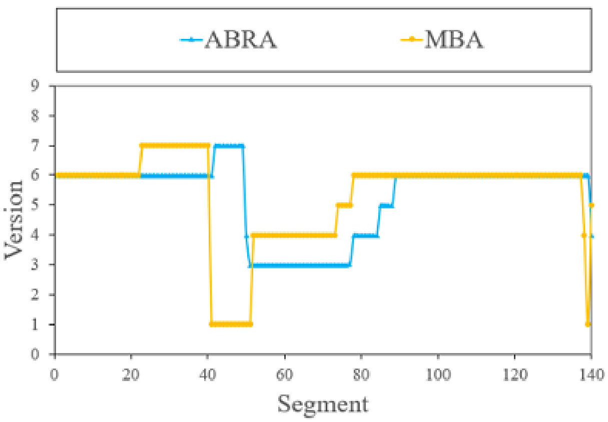
(b) Landscape of selected segment versions

(c) Buffer level

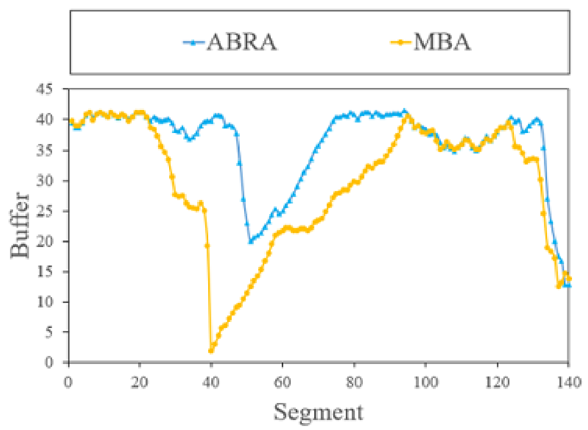
FIGURE 3: Adaptation performance of ABRA vs MBA with bandwidth trace #1



(a) QoE scores and corresponding throughput



(b) Landscape of selected segment versions



(c) Buffer level

FIGURE 4: Adaptation performance of ABRA vs MBA with bandwidth trace #2

the switching level. Fig. 3 and Fig. 4 shows the comparison between ABRA and MBA in terms of QoE, version and buffer in two different bandwidth traces.

The two direct comparisons between the two methods MBA and ABRA in Fig. 3 and Fig. 4 show that the difference in QoE values perceived by users reaches the MOS score of 1.29 in Fig. 3) and 1.77 in Fig. 4. This QoE disparity occurs when throughput drops dramatically, making the difference between the two algorithms obvious.

As we can see, the QoE score of MBA is slightly higher than the that of ABRA at the time before throughput drops down (i.e. the "thrp attenuation" event). However, the good performance of MBA is only temporary for a very short period. We can see that ABRA can optimize the quality for the entire streaming session.

As Fig.4 illustrates, with both of the 2 algorithms, sometimes the version drops down and immediately coming back right afterwards. This fact makes the version increase and decrease continuously in a short period of time, leading to QoE degradation. This phenomenon with ABRA happens more frequently than with MBA. For example at segment 43, 67, 116 and 116, 117 respectively. However, the overall version of ABRA is more stable than MBA. Thereby, the overall quality perceived by clients (i.e QoE score) of ABRA is better than of MBA.

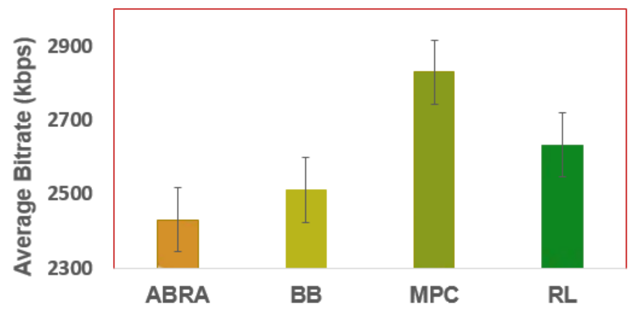


FIGURE 5: Average Bitrate

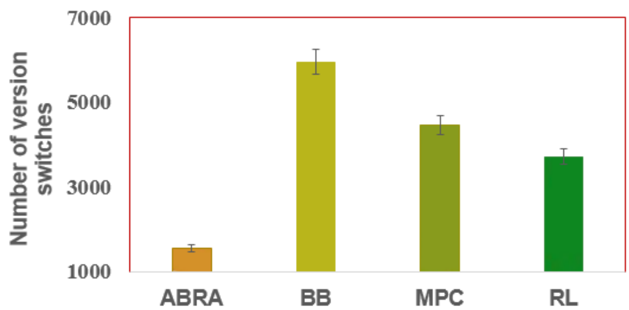


FIGURE 6: Number of version switches

C. ABRA VERSUS OTHER EXISTING METHODS

As already described in section IV-B, ABRA shows a slight improvement over its predecessor- MBA. In this section we will compare ABRA with the state of the art solutions including: MPC [29], Pensive [31], and Buffer-based [37] under real network conditions. The average results of average bitrate, number of switches, time stalling, and the total QoE score are obtained as shown in Fig. 5, 6, 7, and 8, respectively.

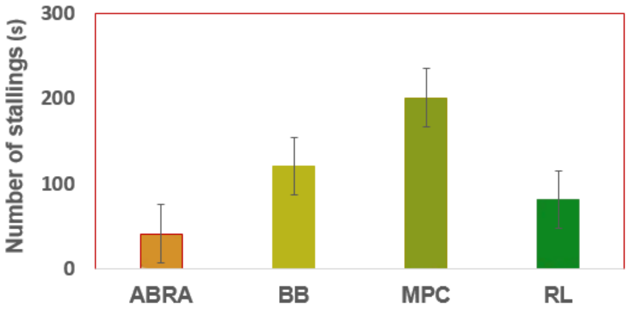


FIGURE 7: Number of Time Stallings

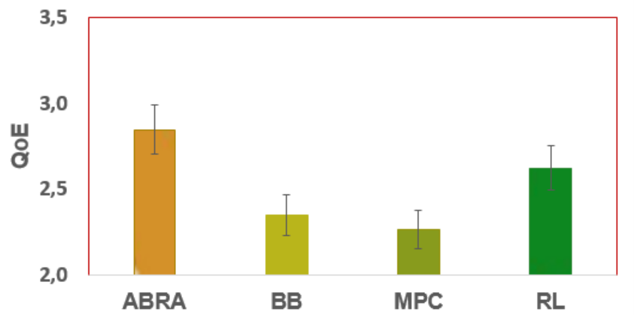


FIGURE 8: Total QoE

In general, as Figure 5 shows, ABRA provides average bitrate 10-20% lower than the existing solutions. However, as we observe the behavior of version switching in Fig. 6, we see that ABRA is the solution that achieves the most stability in deciding versions for next segments. The ABRA algorithm has about 2 times, 3 times, and 5 times less the number of switch versions than the BB, MPC, and RL algorithms. This fact helps to keep the user experience stable, avoiding user annoyance due to frequent video quality changes like other methods do. On the other hand, ABRA also has a low rebuffering time comparable to a compatible solution that relies on buffers, which has approximately 2.5 times, 4 times, and 1.5 times less when compared to the BB, MPC and RL algorithms, respectively. Thanks to these two factors, Fig. 8 shows that ABRA is the solution that achieves the highest QoE score of about 17.55%, 20.41%, and 7.86% than the BB, MPC, and RL algorithms, respectively.

Additionally, ABRA consistently maintains a buffer level more significant than the 20s, a relatively safe buffer level that helps prevent stalling, resulting in video freezing and negatively affecting the user experience. Unlike the Buffer-based methods, in which the prior size maintains the buffer size at a constant level, ABRA dynamically the buffer level to avoid depletion when the throughput reduces. That will prevent the most significant disadvantage of buffer-based systems: too much version change between segments due to buffering concerns. In our opinion, consumers will value keeping steady video quality better than maintaining a stable buffer size because users will only perceive a difference when the buffer is empty, i.e., stalling. Thanks to the two characteristics mentioned earlier, ABRA can be considered an algorithm that obtains the highest QoE level, as shown in Fig. 8. Although MPC always decides to achieve the best level of video quality but causes buffer levels fail to keep safe levels and causes a lot of re-buffering. That will prolong user wait times and make its QoE the worst. Besides, Pensive, a solution based on reinforcement learning, strikes a good balance between improving video quality and maintaining stable buffer levels. However, the compatibility according to the network data does not provide a good user experience when there are too many version changes between segments.

Thereby, we conclude that ABRA has optimized the trade-off between image quality and safe buffer level so that the user experience can be achieved the best among the existing solutions.

V. CONCLUSIONS AND FUTURE WORK

In this research, we have proposed a QoE-driven video adaptation method over HTTP - ABRA. ABRA can flexibly select versions by adapting to bandwidth fluctuation based on throughput variations and the client's status. The advantage of ABRA is that it can work stably in all different ranges of buffer level statuses. It can keep a high QoE score while keeping those scores stable for an extended period. That fact makes ABRA stand out from the existing adaptive streaming schemes in the state of the art. In our future work, we plan to conduct additional experiments in more different bandwidth scenarios to gain a deeper insight into ABRA performance, thereby improving the solution through rough aspects.

ACKNOWLEDGEMENT

This work was funded by Vingroup and supported by Vingroup Innovation Foundation (VINIF) under project code VINIF.2020.DA03.

REFERENCES

[1] Cisco, *Global - 2020 Forecast Highlights*
[2] T. C. Thang, H. T. Le, H. X. Nguyen, A. T. Pham, J. W. Kang and Y. M. Ro, "Adaptive video streaming over HTTP with dynamic resource estimation," in *Journal of Communications and Networks*, vol. 15, no. 6, pp. 635-644, Dec. 2013, doi: 10.1109/JCN.2013.000112.
[3] P. Juluri, V. Tamarapalli and D. Medhi, "SARA: Segment aware rate adaptation algorithm for dynamic adaptive streaming over HTTP," *2015 IEEE International Conference on Communication Workshop (ICCW)*, London, 2015, pp. 1765-1770, doi: 10.1109/ICCW.2015.7247436.
[4] Huyen T.T. Tran, Hung T. Le, Nam Pham Ngoc, Anh T. Pham, and Truong Cong Thang, "Quality Improvement for Video On-Demand Streaming over HTTP", *IEICE Transactions on Information and Systems*, 2017, Volume E100.D, Issue 1, Pages 61-64, Released January 01, 2017, Online ISSN 1745-1361, Print ISSN 0916-8532, https://doi.org/10.1587/transinf.2016MUL0005
[5] W. Choi and J. Yoon, "SATE: Providing Stable and Agile Adaptation in HTTP-Based Video Streaming," in *IEEE Access*, vol. 7, pp. 26830-26841, 2019, doi: 10.1109/ACCESS.2019.2901279.
[6] A. Bentaleb, B. Taani, A. C. Begen, C. Timmerer and R. Zimmermann, "A Survey on Bitrate Adaptation Schemes for Streaming Media Over HTTP," in *IEEE Communications Surveys Tutorials*, vol. 21, no. 1, pp. 562-585, Firstquarter 2019, doi: 10.1109/COMST.2018.2862938.
[7] Chenghao Liu, Imed Bouazizi, and Moncef Gabbouj. 2011. Rate adaptation for adaptive HTTP streaming. In *Proceedings of the second*

annual ACM conference on Multimedia systems (MMSys '11). Association for Computing Machinery, New York, NY, USA, 169–174. DOI:https://doi.org/10.1145/1943552.1943575

[8] D. V. Nguyen, H. T. T. Tran, Pham Ngoc Nam and T. C. Thang, "A QoS-adaptive framework for screen sharing over Internet," *2016 Eighth International Conference on Ubiquitous and Future Networks (ICUFN)*, Vienna, 2016, pp. 972-974, doi: 10.1109/ICUFN.2016.7536942.

[9] Z. Li et al., "Probe and Adapt: Rate Adaptation for HTTP Video Streaming At Scale," in *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 4, pp. 719-733, April 2014, doi: 10.1109/JSAC.2014.140405.

[10] Te-Yuan Huang, Ramesh Johari, Nick McKeown, Matthew Trunnell, and Mark Watson. 2014. A buffer-based approach to rate adaptation: evidence from a large video streaming service.*SIGCOMM Comput. Commun. Rev*. 44, 4 (October 2014), 187–198. DOI:https://doi.org/10.1145/2740070.2626296

[11] C. Mueller, S. Lederer, R. Grandl and C. Timmerer, "Oscillation compensating Dynamic Adaptive Streaming over HTTP," *2015 IEEE International Conference on Multimedia and Expo (ICME)*, Turin, 2015, pp. 1-6, doi: 10.1109/ICME.2015.7177435.

[12] K. Spiteri, R. Urgaonkar and R. K. Sitaraman, "BOLA: Near-optimal bitrate adaptation for online videos," *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*, San Francisco, CA, 2016, pp. 1-9, doi: 10.1109/INFOCOM.2016.7524428.

[13] Abdelhak Bentaleb, Ali C. Begen, Saad Harous, and Roger Zimmermann. 2018. Want to play DASH? a game theoretic approach for adaptive streaming over HTTP. In *Proceedings of the 9th ACM Multimedia Systems Conference (MMSys '18)*. Association for Computing Machinery, New York, NY, USA, 13–26. DOI:https://doi.org/10.1145/3204949.3204961

[14] K. Miller, E. Quacchio, G. Gennari and A. Wolisz, "Adaptation algorithm for adaptive streaming over HTTP," *2012 19th International Packet Video Workshop (PV)*, Munich, 2012, pp. 173-178, doi: 10.1109/PV.2012.6229732.

[15] Cong Wang, Amr Rizk, and Michael Zink. 2016. SQUAD: a spectrum-based quality adaptation for dynamic adaptive streaming over HTTP. In *Proceedings of the 7th International Conference on Multimedia Systems (MMSys '16)*. Association for Computing Machinery, New York, NY, USA, Article 1, 1–12. DOI:https://doi.org/10.1145/2910017.2910593

[16] A. Beben, P. Wiśniewski, J. Mongay Batalla, and P. Krawiec. 2016. ABMA+: lightweight and efficient algorithm for HTTP adaptive streaming. In *Proceedings of the 7th International Conference on Multimedia Systems (MMSys '16)*. Association for Computing Machinery, New York, NY, USA, Article 2, 1–11. DOI:https://doi.org/10.1145/2910017.2910596

[17] H. T. T. Tran, N. P. Ngoc, A. T. Pham and T. C. Thang, "A Multi-Factor QoE Model for Adaptive Streaming over Mobile Networks," 2016 IEEE Globecom Workshops (GC Wkshps), 2016, pp. 1-6, doi: 10.1109/GLOCOMW.2016.7848818.

[18] Big Buck Bunny, Xiph.org Test Media, https://media.xiph.org/

[19] Huyen T. T. TRAN, Nam PHAM NGOC, Yong Ju JUNG, Anh T. PHAM, Truong Cong THANG, A Histogram-Based Quality Model for HTTP Adaptive Streaming, *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 2017, Volume E100.A, Issue 2, Pages 555-564, Released February 01, 2017, Online ISSN 1745-1337, Print ISSN 0916-8508, https://doi.org/10.1587/transfun.E100.A.555

[20] C. Müller, S. Lederer, and C. Timmerer, "An evaluation of dynamic adaptive streaming over HTTP in vehicular environments". In *Proceedings of the 4th Workshop on Mobile Video (MoVid '12)*. Association for Computing Machinery, New York, NY, USA, 37–42. DOI:https://doi.org/10.1145/2151677.2151686

[21] C. T. Dac et al., "QoE-aware Video Adaptive Streaming over HTTP," 2020 IEEE Eighth International Conference on Communications and Electronics (ICCE), 2021, pp. 117-122, doi: 10.1109/ICCE48956.2021.9352077.

[22] S. Kumar, R. Devaraj, A. Sarkar and A. Sur, "Client-Side QoE Management for SVC Video Streaming: An FSM Supported Design Approach," in IEEE Transactions on Network and Service Management, vol. 16, no. 3, pp. 1113-1126, Sept. 2019, doi: 10.1109/TNSM.2019.2926720.

[23] D. J. Vergados, A. Michalas, A. Sgora, D. D. Vergados and P. Chatzimisios, "FDASH: A Fuzzy-Based MPEG/DASH Adaptation Algorithm," in IEEE Systems Journal, vol. 10, no. 2, pp. 859-868, June 2016, doi: 10.1109/JSYST.2015.2478879.

[24] Rahman, W.u.; Hossain, M.D.; Huh, E.-N. Fuzzy-Based Quality Adaptation Algorithm for Improving QoE from MPEG-DASH Video. Appl. Sci. 2021, 11, 5270. https://doi.org/10.3390/app11115270

[25] W. u. Rahman and K. Chung, "Buffer-Based Adaptive Bitrate Algorithm for Streaming over HTTP," KSII Transactions on Internet and Information Systems, vol. 9, no. 11, pp. 4585-4603, 2015. DOI: 10.3837/tiis.2015.11.019.

[26] Sobhani, A.; Yassine, A.; Shirmohammadi, S. A fuzzy-based rate adaptation controller for DASH. In Proceedings of the 25th ACM Workshop on Network and Operating Systems Support for Digital Audio and Video, Portland, Oregon, 18–20 March 2015;pp. 31–36.

[27] 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, 22(1):326–340, Feb 2014

[28] C. Zhou, C. Lin, X. Zhang, and Z. Guo. Tfdash: A fairness, stability, and efficiency aware rate control approach for multiple clients over dash. IEEE Transactions on Circuits and Systems for Video Technology, 29(1):198–211,Jan 2019

[29] Y. Xiaoqi, J. Abhishek, S. Vyas, and S. Bruno. A control-theoretic approach for dynamic adaptive video streaming over http. In Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication, SIGCOMM '15, pages 325–338, New York, NY, USA, 2015. ACM.

[30] A. H. Zahran, D. Raca and C. J. Sreenan, "ARBITER+: Adaptive Rate-Based InTElligent HTTP StReaming Algorithm for Mobile Networks," in IEEE Transactions on Mobile Computing, vol. 17, no. 12, pp. 2716-2728, 1 Dec. 2018, doi: 10.1109/TMC.2018.2825384.

[31] M. Hongzi, N. Ravi, and A. Mohammad. Neural adaptive video streaming with pensieve. In Proceedings of the Conference of the ACM Special Interest Group on Data Communication, SIGCOMM '17, pages 197–210, New York, NY, USA, 2017. ACM.

[32] Xuekai Wei, Mingliang Zhou, Sam Kwong, Hui Yuan, Shiqi Wang, Guopu Zhu, Jingchao Cao,Reinforcement learning-based QoE-oriented dynamic adaptive streaming framework,Information Sciences,Volume 569,2021,Pages 786-803,ISSN 0020-0255,https://doi.org/10.1016/j.ins.2021.05.012.

[33] L. De Cicco, V. Caldaralo, V. Palmisano and S. Mascolo, "ELASTIC: A Client-Side Controller for Dynamic Adaptive Streaming over HTTP (DASH)," 2013 20th International Packet Video Workshop, 2013, pp. 1-8, doi: 10.1109/PV.2013.6691442.

[34] S. Sengupta, N. Ganguly, S. Chakraborty and P. De, "HotDASH: Hotspot Aware Adaptive Video Streaming Using Deep Reinforcement Learning," 2018 IEEE 26th International Conference on Network Protocols (ICNP), 2018, pp. 165-175, doi: 10.1109/ICNP.2018.00026.

[35] K. Spiteri, R. Sitaraman, and D. Sparacio. From theory to practice: Improving bitrate adaptation in the dash reference player. ACM TOMM, 15(2s):1–29, 2019.

[36] Z. Akhtar et al., "Oboe: Auto-tuning video ABR algorithms to network conditions," in Proc. Conf. ACM Special Interest Group Data Commun., Aug. 2018, pp. 44–58.

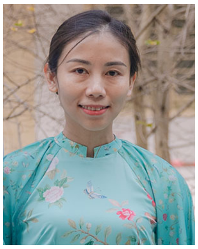
[37] T.Y. Huang et al. 2014. A Buffer-based Approach to Rate Adaptation: Evidence from a Large Video Streaming Service. In SIGCOMM. ACM.

NGUYEN VIET HUNG received the B.Sc.degree in Bachelor of Informatics pedagogy from the faculty of Engineering Technology of Ha Tinh University, Vietnam, in 2012, the M.Sc. degree Master of Information Technology from the Faculty of Information Technology of Ho Chi Minh City University of Technology, Vietnam, in 2016, and has been learning the Ph.D. degree in Telecommunications Engineering from the Hanoi University of Science and Technology, Vietnam. His research interests include multimedia communications, network security, artificial intelligence, traffic engineering in next-generation networks, QoE/QoS guarantee for network services, green networking, and applications.

TRINH DAC CHIEN is a graduate majored in Electronics and Telecommunication Engineering of Hanoi University of Science and Technology. He is currently holding a the Research Assistant position at VinUniversity. His research interests include video streaming and multimedia processing.

PHAM NGOC SON was a graduate from the talented program in Smart Electronics system and IoT of Hanoi University of Science and Technology. He is currently pursuing the master's degree at the Hanoi University of Science and Technology. He is also a Research Assistant in the Future Internet Laboratory, School of Electrical and Electronic Engineering. His research interests include video streaming, multimedia processing.

PHAM NGOC NAM is the Vice Dean of College of Engineering and Computer Science, VinUniversity, Vingroup and also a visiting scholar at Cornell University. He earned his bachelor's degree in Electronics Engineering at Hanoi University of Science and Technology (HUST) in 1997 and his M.S. in Artificial Intelligence and Ph.D. in Electrical Engineering from KU Leuven, Belgium in 1999 and 2004 respectively. His research interests include Artificial Intelligence, QoS/QoE management for multimedia applications, reconfigurable computing and low-power embedded system design. He is the author, co-author of 100 scientific articles including more than 30 ISI and Scopus publications. He has been the PI of 1 key national project, 3 ministerial-level projects and a key member in four other national projects.



TRUONG THU HUONG is the Deputy Head of Department of Communication Engineering, Hanoi University of Science and Technology. She received the B.Sc.degree in Electronics and Telecommunications from the Hanoi University of Science and Technology (HUST), Vietnam, in 2001, the M.Sc. degree in information and communication systems from the Hamburg University of Technology, Germany,in 2004, and the Ph.D. degree in telecommunications from the University of Trento, Italy, in 2007. Her research interests are oriented toward network security, artificial intelligence, traffic engineering in next generation networks, QoE/QoS guarantee for network services, green networking, and development of the Internet of Things ecosystems and applications.