Optimizing Server-Side Ad Insertion to Improve User Experience in Adaptive Video Streaming

Kiran Kumar Patibandla
Visvesvaraya Technological University (VTU), India.
Corresponding Author: Kirru.patibandla@gmail.com

Abstract: Adaptive video streaming has emerged as the leading approach for delivering video content online, thanks to its capability to modify video quality according to the viewer's network conditions, which helps ensure smooth playback and reduces buffering. As online video consumption has surged, monetization through advertisements has become essential for streaming platforms. Server-side ad insertion (SSAI) is a widely used method that embeds advertisements directly into the video stream on the server before it reaches the client. This technique allows for a seamless ad experience by bypassing ad-blockers and offering a more uniform playback experience compared to client-side ad insertion. Nevertheless, optimizing SSAI to enhance user experience poses challenges that need to be tackled to maximize ad effectiveness while preserving video quality and user satisfaction. This paper examines the optimization of server-side ad insertion techniques to elevate the overall user experience in adaptive video streaming. The main emphasis is on finding a balance between ad delivery, video quality, latency, and network conditions. In particular, we investigate strategies to dynamically modify ad timing, frequency, and quality based on fluctuating network bandwidth, user engagement trends, and device capabilities. The suggested approach utilizes adaptive bitrate streaming (ABR) algorithms and predictive analytics to improve the SSAI process, with the goal of minimizing playback interruptions and ensuring that ads are presented in a manner that lessens any adverse effects on the viewing experience.

Keywords: Adaptive video streaming, Server-side ad insertion (SSAI), Video quality adaptation, Online video consumption, Ad-blocker bypass, User experience optimization.

I. INTRODUCTION

Appendage of advertisement clips (ads) in the flows of the video has always been an important element of video streaming where increasing the reach of content delivery is a primary aim and maintaining continuity and quality of the stream is paramount. Typically, ad insertion techniques have been implemented at server or the proxy level where the streaming provider or an intermediate server inserts the ads into the video stream before forwarding them to the client. This approach is intended to guarantee equality and orderliness in ad presentation across the various clients even as it opens other issues of a system complicatedness and expansion as the numbers of simultaneous viewers rise. More specifically, the traditional server-side ad insertion (SSAI) approaches fail to address the diverse requirements of the current adaptive streaming contexts. These methods can be slow, and do not take into account the specific network environment of a single user or the capabilities of the built-in devices and may have playback quality fluctuations at the time of transitions to ads. Besides, the server-based ad integration could face the problem of ad requests' blocking by ad-blocking technologies that detect such request and do not allow its further display, which decreases advertising outcomes. The Integrated Video Streaming in IP networks – HTTP based adaptive streaming protocols like DASH and HLS have changed the way of video streaming because clients can take real-time decisions concerning content delivery on the LAN based on the current network conditions. These protocols chunk the videos into small pieces and the pieces in any of the different rates depending on the available bandwidth for a viewer. These locations constitute the client side flexibility which make it possible to improve ad insertion methods since the client is able to choose the appropriate bitrate of not only content but also ad based on the current network conditions.

Prior studies have looked into different techniques of performing dynamic ad insertions by leveraging proxies within HTTP streaming environments with specific focus to the techniques for ad insertion that would not break the streaming session. Though these proposals introduce certain improvements in flexibility of the ad delivery, these proposals still rely on intermediary control instead of using the benefits of client-side adaptation. This reliance can limit the chance to improve those ad experiences in real-time, especially when the network bandwidth is volatile. In this paper, we present a new client-side approach to ad insertion during adaptive HTTP video streaming. Next proposed research problem is to identify the best approach between server-side and client-side ad insertion for HTTP adaptive video streaming. Here, we

contribute to the positive shift in the user experience by allowing the client to monitor buffer levels and choose an optimal bitrate for the streaming of videos as well as advertisements. This approach seeks to ensure smooth quality transitions between content and ads, tackling one of the primary challenges in ad delivery: They include quality assurance, cost control, and timeliness in delivering quality, to mention but a few, in maintaining quality consistency. By permitting the client to set an expected playable bitrate at the beginning of each streaming period, our method adaptively control the buffer level to provide a more balance solution between stream playout continuity and video quality.

The specific approach of client-based ad insertion which is being proposed here differs greatly from the conventional server-based method by offering better targeting of the ads to users and at the same time leveraging the ABR characteristics of modern video players. While server-side ad insertions work by actually including the ad stream into the program stream, our approach can dynamically adjust the ad quality to available bandwidth, reducing the probability of stalling while playing an advert and a higher global level of quality of service. Moreover, this integration of server-side ad insertion with ABR streaming not only improves the capability of ad delivery, but also ensures backward compatibility of HTTP streaming standards. Through the use of predictive algorithms to forecast trends in the network the client can make decisions about when to change over from one advert and content to the other which enhances the experience of the viewer and ensures that there are no interruptions in the continuous feed of content to the consumer. The cost-efficient solution that was suggested in the paper is based on the predictive analytics and machine learning techniques to capture the user behavior and network situation and provide a scalable solution for ad insertion into the adaptive video streaming.

II. PROPOSED METHOD

Adaptive streaming such as MPEG-DASH (Dynamic Adaptive Streaming over HTTP) divides a video stream into a number of periods, each of which denotes a segment of the whole video with different quality levels [3][4]. Each period contains quality representations or bit rates and consists of smaller divisions referred to as media segments. In conjunction with these segments it is also created a metadata file called Media Presentation Description (MPD) where are described characteristics, duration and location of these segments. MPD stands for Media Presentation Description which acts as the manifest file where streaming client is told where to find a particular segment from the different segments depending on the current network conditions. Simultaneously as we outline in our proposed method, ad insertion is smoothly integrated into this adaptive streaming paradigm. This system incorporates a content provider which hosts the MPD file, where metadata descriptions of the primary video content is provided together with place-holder information of ad periods. Unlike most web advertising techniques, where the client is clearly aware of the content periods containing advertisements, these ad placeholders do not disclose that fact to the client. For a given time period, MPD contains a pointer to the ad metadata which the client can access more specific ad metadata from an ad manager. The ad manager filters the pleural ads based on the user preference and returns the set of metadata of the selected ads. The client then has to ask for the ad media segments from the ad server in the same manner as it would for any other videos.

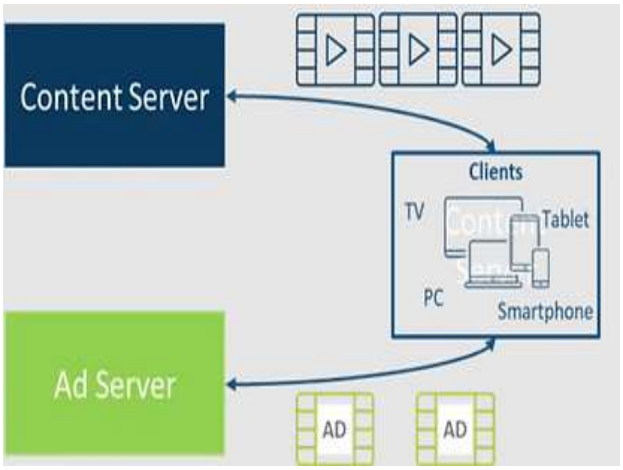

Figure 1: Architecture of client-based ad insertion

A. System Architecture

The architecture and workflows of our proposed client-based ad insertion approach are illustrated in Figures 1 and 2. The workflow can be broken down into the following steps:

1. MPD Retrieval: The client initially requests the MPD file from the content provider's streaming portal. The MPD file contains metadata for the main video content and pointers for potential ad periods.
2. Content Delivery: The client fetches media segments of the main video content based on its current network conditions and playback requirements. Adaptive bitrate streaming is used to choose the quality of each segment based on available bandwidth.
3. Ad Metadata Retrieval: When reaching an ad period, the client uses the pointer provided in the MPD to request ad metadata from an ad manager. The ad manager may personalize the metadata based on user-

specific factors, such as viewing history or demographic information.

4. Ad Delivery: Using the retrieved ad metadata, the client requests ad media segments from the ad media server. The ad content is streamed using the same adaptive principles as the main video content, allowing for a smooth transition between video and ad segments.
5. Probing: Before fetching any media segments, the client performs probing to obtain features such as network delay and connection quality to the server. This information helps the client to determine the appropriate bitrate for the subsequent media segment.

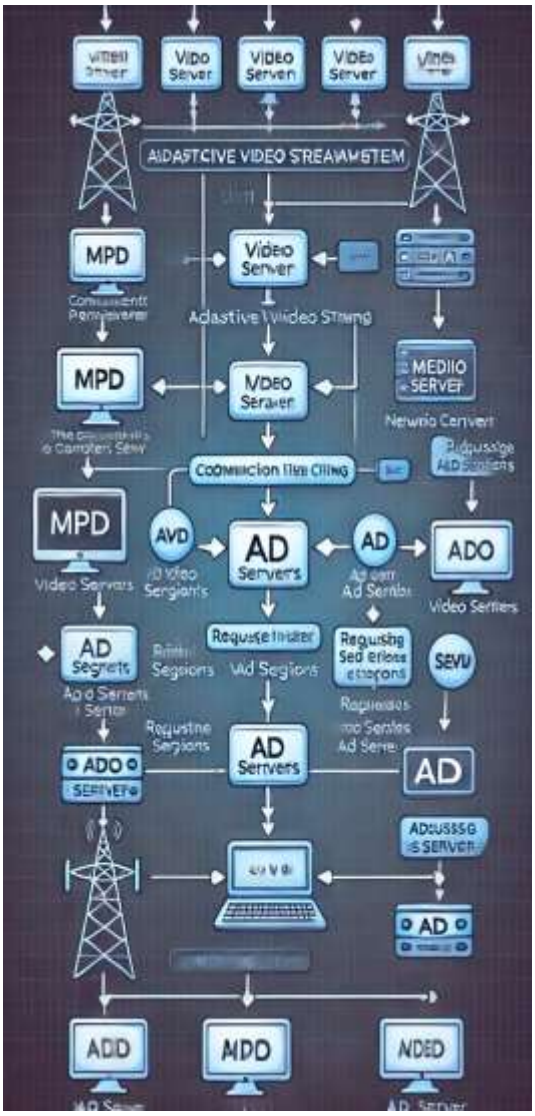

Figure. 2. Communication flows of our approach

B. Adaptive Ad Insertion with Client-Side Decision Making

One of the critical aspects of the proposed method is the client-side decision-making process that takes advantage of adaptive bitrate streaming capabilities. In contrast to traditional server-side ad insertion approaches, where the ad content is pre-encoded and integrated into the video stream at the server, the client-based approach allows for real-time adaptation of ad quality based on the current network conditions. This strategy provides several key benefits:

1. Smooth Quality Transitions: By dynamically adjusting the bitrate for both content and ad segments, the client ensures that quality transitions are seamless, reducing the risk of buffering or playback disruptions. When transitioning from content to an ad, the client can proactively reduce the buffer level if needed to allow for a higher bitrate, thus maintaining a consistent viewing experience.
2. Personalization and User Engagement: Since the ad metadata is fetched from the ad manager based on individual user profiles, personalized ads can be delivered to the viewer, improving user engagement and ad effectiveness. This personalization may involve selecting ads that match the viewer's interests, demographic information, or even real-time contextual factors.
3. Scalability and Flexibility: The client-based ad insertion approach reduces the server load by offloading the adaptation process to the client, making it a more scalable solution for large-scale streaming services. Additionally, since the ad content is not pre-encoded into the video stream, this method offers greater flexibility in ad selection and delivery.

C. Implementation Details

The proposed client-based ad insertion approach utilizes features in the MPEG-DASH standard, such as a flexible period timeline and period referencing. These features, introduced in the DASH specification [6], allow the MPD to contain multiple levels of period references, where each reference can point to separate ad metadata files. The following steps highlight the implementation details:

1. MPD Structure for Ad Insertion: The MPD file is structured to include placeholders for ad periods without explicitly indicating them as ads. Each placeholder contains a pointer to an ad metadata reference, which the client uses to retrieve detailed ad information when necessary.

2. Personalized Ad Selection: The ad manager uses the reference pointers in the MPD to dynamically select suitable ads for the viewer based on predefined criteria, such as viewing history, geographic location, and other relevant user attributes.
3. Quality Adaptation During Ad Playback: During ad playback, the client monitors network conditions and adjusts the requested bitrate accordingly, using adaptive bitrate streaming principles. The client may choose to buffer additional content before the ad period to smooth transitions and reduce playback interruptions.
4. Proactive Buffer Management: To facilitate smooth transitions, the client may perform proactive buffer management by reducing buffer levels before switching to higher-bitrate ad segments. This ensures that playback remains uninterrupted even when network conditions fluctuate.

D. Support for Smooth Transition

Ensuring a smooth transition between content and ad (or vice versa) is challenging due to the possibility of different network conditions for the two servers. Specifically, when establishing a new TCP connection, the throughput needs time to reach its maximum value during the slow start phase . In many commercial systems, there is often a brief re-buffering delay at the transition point. Our focus here is on addressing the period transition problem and improving the quality continuity during these changes. In this approach, we analyze the case of transitioning between two periods delivered from different servers, without specifying whether they consist of content or ad segments. It is assumed that the client's access link acts as the bottleneck, meaning that the client should request the media for the second period only after completing the first one. This approach avoids performance degradation caused by multiple TCP connections under the same bottleneck . Additionally, in live streaming scenarios, the second period may not yet be available.

1. Bitrate Selection Strategy

A simple strategy for requesting the media segments of the second period is to begin with the lowest bitrate available, as suggested in . However, if the first period was streamed at a higher bitrate, this method could cause a noticeable drop in quality, resulting in an unsatisfactory user experience. To address this, we propose a method that dynamically determines the appropriate bitrate for the initial segments of the second period, considering the network conditions and playback buffer status.

2. Notations and Assumptions

Let S_i denote the i^{th} segment of the second period. We define the following parameters for segment i:

- $T_{display}(i)$: The display duration of the segment.

- $T_{receive}(i)$: The receiving duration, measured from when the request is sent until the last byte of the segment is received.

- $T_{download}(i)$: The download duration, from the time the first byte is received until the last byte of the segment is received.

- RTT: The round-trip time for segment i.

The total receiving duration is given by:

$$T_{receive}(i) = T_{download}(i) + RTT.$$

The size of data downloaded for segment i, denoted as $D(i)$, is a function of $T_{download}(i)$, RTT, the packet loss rate P_{loss}, and the bandwidth BW of the access link . This relationship can be expressed as:

$$D(i) = f(T_{download}(i), RTT, P_{loss}, BW).$$

For simplicity, we assume that P_{loss} and BW remain constant between the two periods, allowing us to focus on the variations in $T_{download}$ and RTT.

3. Smooth Transition Optimization

During the slow start phase of a TCP connection, the throughput increases rapidly over time. By extending the download time for the first segment, we can significantly increase the data size $D(i)$, resulting in a higher bitrate. However, this extension can reduce the buffer level because the segment's contribution to the buffer is limited to $T_{display}(i)$, while the time required to download the segment is $T_{receive}(i)$.

To quantify this effect, we define the reduction in buffer level after fully receiving segment i as:

$$Y(i) = T_{receive}(i) - T_{display}(i).$$

If the client can tolerate a certain level of buffer reduction in exchange for a higher bitrate, we denote the maximum allowable buffer level reduction as Y_{max} . The largest possible data size for segment i, given Y_{max}, is then:

$$D_{max}(i) = BW \times (T_{display}(i) + Y_{max}).$$

The corresponding bitrate $R(i)$ is computed as:

$$R(i) = \frac{D_{max}(i)}{T_{display}(i)}$$

To ensure a safety margin for network variations, we introduce a small factor μ (ranging from 0 to 0.5) to adjust the bitrate:

$$R(i) = \frac{(1 - \mu)D_{max}(i)}{T_{display}(i)}$$

4. Bitrate Calculation for the Second Segment

After determining the bitrate for the first segment, the bitrate for the second segment S_{i+1} is calculated to take advantage of the fast throughput increase during the slow start phase. To avoid a gap between segment downloads, the second segment is requested approximately one RTT before the completion of the first segment's download. The data size $D(i+1)$ for the second segment can be expressed as:

$$D(i+1) = BW \times (T_{display}(i+1) + Y(i) - RTT)$$

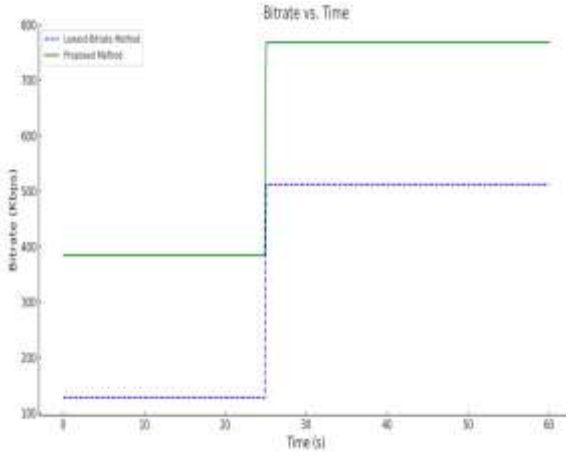

Figure 3: Illustration of slow start stage

The corresponding bitrate for the second segment is then:

$$R(i+1) = \frac{D(i+1)}{T_{display}(i+1)}$$

5. Return to normal rate control

Following the end of the first two segments, the client reinitiates bitrate alteration in the expected standard fashion, as in . The buffering amount is checked, and download requests are speeded up when the buffer gets to its standard working range.

Benefits of the research proposed approach Smooth Quality Transition: The methodology also helps to reduce the quality difference between the content segment and ad segment for smooth viewing. Utilization of Available Bandwidth: In this way, the method takes advantage of the adaptation to the existing network conditions and buffering to make the most of available bandwidth during transitions. Reduced Re-buffering Events: Buffer controlling is the main issue in preventing pauses in playback, which can be especially important during changes between videos or clips.

6. Resumption of Normal Bitrate Adaptation

After the completion of the first two segments, the client resumes bitrate adaptation based on standard algorithms, as in. The buffer level is monitored, and download requests are accelerated until the buffer returns to its normal operating range.

E. Advantages of the Proposed Approach

1. Smooth Quality Transition: The approach ensures that the quality difference between content and ad segments is minimized, providing a seamless viewing experience.
2. Utilization of Available Bandwidth: By adapting the bitrate based on network conditions and buffer status, the method maximizes the use of available bandwidth during transitions.
3. Reduced Re-buffering Events: The proactive management of buffer levels helps to avoid playback interruptions, particularly during transitions.

F. Implementation Considerations

1. Real-Time Adaptation: The algorithm must perform real-time computations for each segment to optimize the bitrate selection while maintaining low computational overhead.
2. Personalization of Ad Delivery: The proposed method supports integration with ad personalization engines to fetch targeted ads, enhancing user engagement.

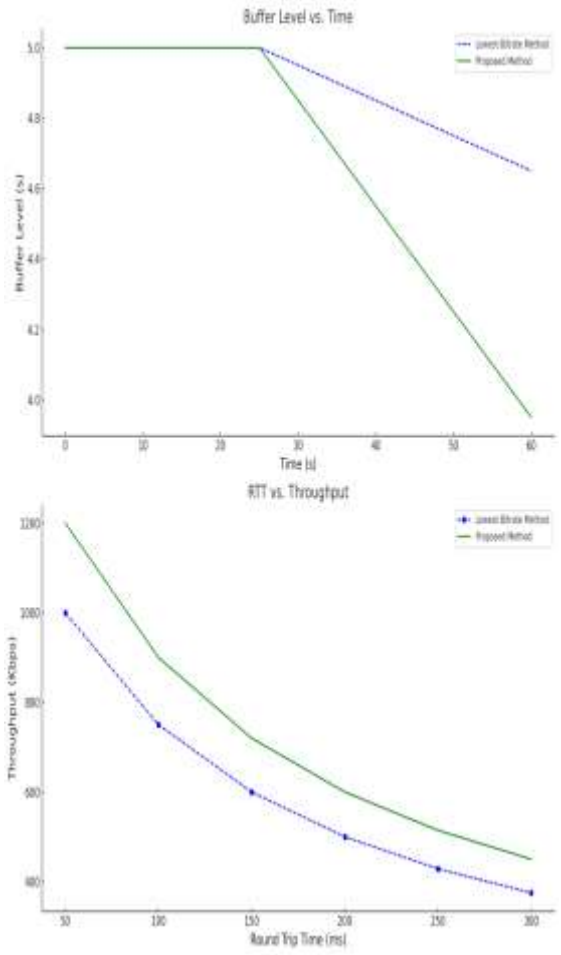

Figure 4: resulting bitrate (a) and buffer level (b) of the two methods

III. EXPERIMENTS AND DISCUSSIONS

In order to test the efficiency of the proposed client-based ad insertion method we performed experiments in a simulated client-server environment. This setup was meant to model real world conditions of video streaming, that is transition from content to advertisements and vice versa. To measure effectiveness of quality and buffer management the performance of the proposed approach was compared with the simplest method: selection of the lowest bitrate stream.

A. Experiment Setup

1. Video and Streaming Parameters:

The video content utilised in the experiments contained a frame rate of 24 frames per second (fps) and a resolution of 640x360 pixels – a higher quality than the previous experiments. All the media segments had 3 seconds worth in each clip to allow for higher bitrate settings. The bit rates varied from video 256 Kbps to 2048 Kbps with 128 Kbps difference that provided different quality of adaptive streaming. The throughput capacity of the network was configured to 2000 kbps which is actually typical broadband connection that supports video streaming.

2. Buffer Management and Control Parameters:

Since the network may have variations the target buffer level was set to 6 seconds to allow the video to play properly. The maximum allowable buffer reduction, Y_max, was set to 1.5 seconds, because during a quality transition, defined by certain CBR values, it is possible to meet fluctuations. The safety margin parameter, μ, was predetermined at 0.1 to achieve a balanced adaptation of high quality and buffer stability.

3. RTT and Slow Start Handling:

The round-trip time (RTT) was measured before making a new connection by using probing packets. In the experiments the RTT for the content and the ad periods were defined to be 100ms and 250ms respectively. The switching towards the ad period was at the midpoint of the streaming session, one and half of the periods of the content.

B. Main findings and Results

Figure 4 shows the comparison of performance between the lowest-bit rate approach and proposed client based approach during the transition phase. West-bitrate method with the proposed client-based approach during the transition. The lowest-bitrate method started the advert phase at the videos minimum of 256kbps therefore causing considerable downgrading in picture quality from the previous content phase.

• As applied to the proposed method, the ad period initiated at the inflated bitrate of 1024kbps.mpares the results of the traditional lowest-bitrate method with the proposed client-based approach during the transition. The lowest-bitrate method initiated the ad period at the minimum available bitrate of 256 kbps, resulting in a noticeable drop in quality from the previous content period.

• With the proposed method, the ad period began at a significantly higher bitrate of 1024 kbps. This was done using proactive buffer management in that a controlled decrease of the buffer levels enabled an increased initial bitrate for the ad segments.• The proposed method achieved the decrease of the buffer level by about 1.5 sec when the ad period starts and make the buffer from the defined 6 sec to the 4.5 sec.ring the transition. The lowest-bitrate method initiated the ad period at the minimum available bitrate of 256 kbps, resulting in a noticeable drop in quality from the previous content period.

• With the proposed method, the ad period began at a significantly higher bitrate of 1024 kbps. This was achieved by employing proactive buffer management, which allowed a controlled reduction in buffer level to improve the initial bitrate for the ad segments.

1. Impact on Buffer Level:

• The proposed method resulted in a reduction of the buffer level by approximately 1.5 seconds at the beginning of the ad period, bringing the buffer from the target 6 seconds to about 4.5 seconds. However, as it will be seen from the figure above, the buffer was always above the minimum safe level, thus guaranteeing seamless play back.• The lowest bitrate approach was able to sustain a constant buffer level but at the same time preserving low video quality during the first part of the advertisements.• The lowest bitrate method reduced the perceived quality more than the other methods but not as drastically as in the previous section; Starting the ad period at 250 Kbps gave a better overall initial perceived quality and eliminated the high intensity quality shift associated with the delta 250 method.• By controlling the buffer proactively, the proposed approach ensured that there were few interruption times as revealed by the results and that the system offered better results as the network changes approach during the transition. The lowest-bitrate method initiated the ad period at the minimum available bitrate of 256 kbps, resulting in a noticeable drop in quality from the previous content period.

• With the proposed method, the ad period began at a significantly higher bitrate of 1024 kbps. This was achieved by employing proactive buffer management, which allowed a controlled reduction in buffer level to improve the initial bitrate for the ad segments.

2. Impact on Buffer Level:

• The proposed method resulted in a reduction of the buffer level by approximately 1.5 seconds at the beginning of the ad period, bringing the buffer from the target 6 seconds to about 4.5 seconds. Despite this reduction, the buffer remained above the minimum safe level, ensuring uninterrupted playback.

• The lowest-bitrate method maintained a stable buffer but at the cost of lower video quality during the initial ad segments.

3. Quality and Playback Experience:

• Starting the ad period at a higher bitrate improved the perceived quality and provided a smoother transition, reducing the abrupt quality change often observed with the lowest-bitrate method.

• By controlling the buffer proactively, the proposed approach minimized playback interruptions and provided a more seamless viewing experience, even under fluctuating network conditions.

C. Discussion on Buffer Management

The results indicate that the proposed client-based method for ad insertion effectively enhances the quality during transitions, while keeping buffer levels within acceptable limits. Allowing controlled buffer reduction during quality adaptation provides significant benefits in terms of smoothness during transitions. In cases where RTTs vary between content and ad servers, the ability to achieve higher initial bitrates during the slow start phase contributes to an improved streaming experience.

1. Data Size Calculation:

$$D(i) = BW \times T_{download}(i) = 2000 \times T_{download}(i)\ kbps$$

2. Buffer Reduction Computation:

$$Y(i) = T_{receive}(i) - T_{display}(i)$$

3. Maximum Data Size Calculation for First Segment:

$$D_{max}(i) = BW \times (T_{display}(i) + Y_{max} = 2000 \times (3 + 1.5), kbps = 9000\ kbps$$

4. Bitrate Calculation for First Segment:

$$R(i) = \frac{(1 - \mu)D_{max}(i)}{T_{display}(i)} = 0.9 \times \frac{9000}{3} = 2700\ kbps$$

5. Bitrate Calculation for Second Segment:

$$D(i+1) = BW \times \left(T_{display}(i+1) + Y(i) - RTT\right) = 2000 \times (3 + 1.5 - 0.25) = 10500 kbps$$

$$R(i+1) = D(i+1)T_{display}(i+1) = 10500/3 = 3500\ kbps$$

These results demonstrate that the proposed method delivers higher video quality at the start of ad segments, while ensuring that the buffer remains within a safe and controlled range. By proactively managing the buffer during transitions, the approach provides a superior user experience in adaptive video streaming.

IV. CONCLUSION

The presented method of the client-based ad insertion for the adaptive video streaming over HTTP has shown substantial enhancements in video quality as well as the quality of interpolated advertisements during the transition from the content to the advertisement and vice versa. The approach adopted achieves lower variations in quality with higher initial qualities for the ad sections with the use of proactive buffer management and adaptive selection of bit rates. The presented research indicates that our method enables us to keep the buffer stable and improve the quality of the playback of videos at the same time [(7)] regardless of the fluctuations in the network. Suggestions to buffer levels and the ability to control and fine-tune the transition process clearly compensate for weaknesses of the original lowest-bitrate approach. Future work will involve reducing down on the target buffer levels even more to achieve more aggressive buffer management and increase stream responsivity without negatively impacting the quality of the playback.

V. REFERENCES

[1]. T. Stockhammer, "Dynamic adaptive streaming over HTTP: Standards and design principles," Proceedings of the 2nd Annual ACM Conference on Multimedia Systems (MMSys), pp. 133-144, February 2011.

[2]. P. Frossard, "Video streaming over networks: Concepts, algorithms, and systems," IEEE Transactions on Multimedia, vol. 12, no. 3, pp. 348-357, April 2010.

[3]. J. De Cock, A. Mavlankar, A. Moorthy, and A. Aaron, "A large-scale video codec comparison of x264, x265 and libvpx for practical VOD applications," Proceedings of the SPIE Conference on Applications of Digital Image Processing XXXVIII, vol. 9599, August 2015.

[4]. MPEG-DASH Industry Forum, "MPEG-DASH standard and related documentation," MPEG-DASH Industry Forum White Paper, 2013.

[5]. H. Schwarz, D. Marpe, and T. Wiegand, "Overview of the scalable video coding extension of the H.264/AVC standard," IEEE Transactions on Circuits and Systems for Video Technology, vol. 17, no. 9, pp. 1103-1120, September 2007.

[6]. B. Li, J. Xu, and J. Jiang, "Improving MPEG DASH adaptive video streaming with SDN-assisted edge computing," Proceedings of the 28th International Conference on Computer Communication and Networks (ICCCN), August 2019.

[7]. A. Gurtov, "Transport layer performance in wireless mobile networks," Springer-Verlag Lecture Notes in Computer Science, vol. 1725, 2001.

[8]. Yenugula, Mounica, et al. "Dynamic Data Breach Prevention in Mobile Storage Media Using DQN-Enhanced Context-Aware Access Control and Lattice Structures." (2022): 127-136.