Lab Assignment 1b Report: Adaptive Bitrate Algorithm Performance on DASH

Kiana Greek 40135171, Ryan Haniff 27069421

I. Abstract

This report discusses an experiment conducted to compare the efficiency of three adaptive bitrate algorithms (throughput-based, buffer-based, and dynamic). DASH is an increasingly popular approach to streaming videos on demand. For the lab, we used a media application based on dash.js that plays the video with adaptive bitrate streaming. The task was to experiment with the adaptive bitrate algorithms and network conditions to compare the efficiency of the algorithms. Every eight seconds we collected various metrics for comparison using Javascript.

II. Introduction

Video Streaming is one of the biggest uses of the Internet today. Stored videos are placed on a server and the user makes requests to the servers to view the video. With HTTP streaming, videos are saved on an HTTP server as a web object with a specific URL. When a user wants to watch a video, the application creates a TCP connection with the server and makes a request for the video. One approach to delivering the video is Dynamic Adaptive Streaming over HTTP (DASH). Dynamic adaptive streaming stores multiple different encodings in the server. Each encoding has its own URL. A manifest file stores the bitrate and its location. Instead of requesting the whole video at once, the user will request a few segments at a time. Depending on the user's network conditions, the quality of the video will automatically adapt. If network conditions are good, then the user will request the higher quality. If network conditions become poor, it will select lower-quality segments. A change in network connections while the video is playing will cause the quality of the video to change.

The dash player used (dash.js) has three adaptive bitrate algorithms: throughput-based, buffer-based (BOLA), and dynamic [1]. The throughput-based algorithm estimates the available bandwidth and picks the best bitrate that is less than or equal to that value [2]. The buffer-based algorithm determines which quality to download based on the available space in the buffer. If the buffer is occupied, then the higher bitrate version is downloaded. The lower bitrate version is downloaded if the buffer is empty or close to empty. The dynamic algorithm uses a combination of throughput-based and buffer-based Algorithms.

For our experiment, we tested the three adaptive bitrate algorithms while throttling network conditions. We created a script to collect bitrate in Mbps, buffer level in seconds, throughput in Mbps, segment download time in seconds, and segment size in bytes. We collected the metrics with JavaScript and dash.js built-in functions [3]. To be able to access some of the built-in functions, we needed to use the player object for the video. We used the functions getDashMetrics(), getDashAdapter() and getStreamInfo() to obtain average throughput, bitrate, and buffer level. The segment download time and segment size required getCurrentHttpRequest() and getHttpRequests(). Getting the current HTTP request has sub-functions called _tfinish and tresponse. We are able to get the time using getTime() function. Then subtracting tresponse from tfinish we get the segment download time. For the segment size, we had to first look for "MediaSegment" requests. If there is a segment request being made then we use reduce() and pass in "acc" and "cur". The callback function "acc" is summed with the first element in "cur" to calculate the segment size. We saved all metrics to a txt file once the playback of the video has ended. When that event gets called, we created a function called writeLogsToFile() that is executed and saves all of our logs.

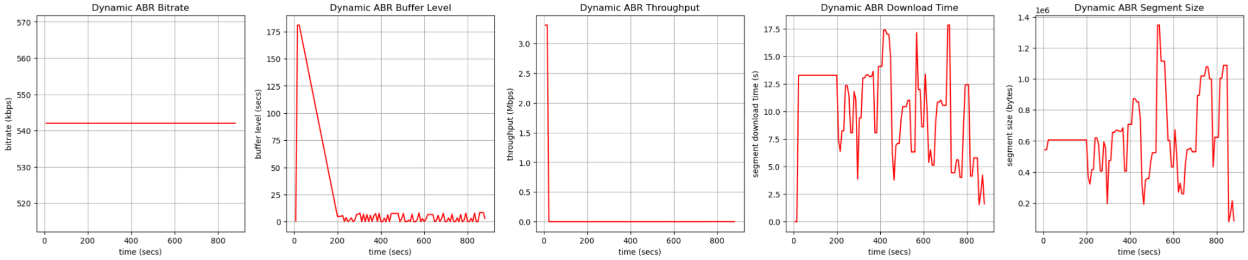

Fig.1. Dynamic ABR

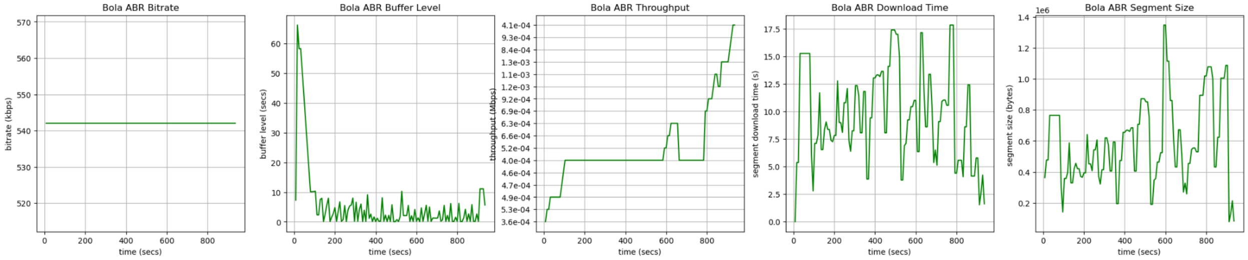

Fig. 2. BOLA ABR

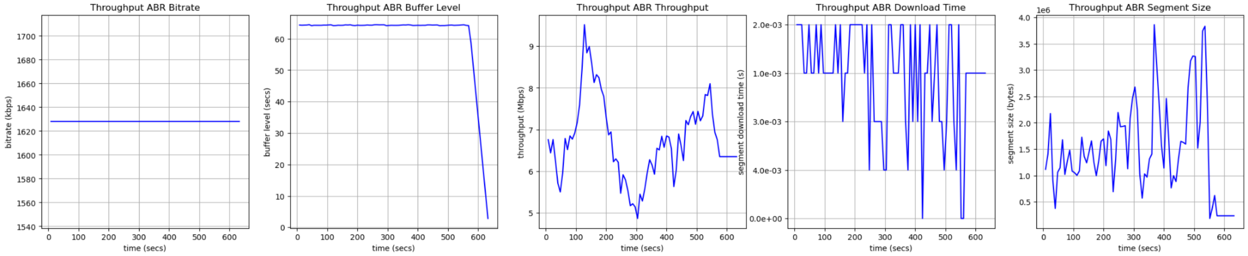

Fig. 3. Dynamic ABR

III.Results

The throughput-based algorithm had a consistent bitrate and relatively consistent buffer level while throughput, segment download time, and segment size varied. Bitrate was 1628 Kbps for the whole video. Buffer level was around 64 Mbps up until the last 8 time units. As the video ended, the buffer level would quickly approach 0 seconds. Throughput was 5-8 Mbps for the duration of the video. Download time was between 0 seconds - 0.004 seconds with the most common being .001 seconds. Segment size was between 235679 - 3860075 bytes. The lowest values for segment size were in the last few measurements. All metrics for the buffer-based algorithm are graphed in Fig. 1.

The buffer-based algorithm had a consistent bitrate but the rest of the metrics varied. Bitrate was consistently 542 Kbps. Buffer level varied greatly ranging from 0.115-66.136 Mbps. Most values were in the 0-10 Mbps range. Throughput varied from 0.0004 - 0 .0013 Mbps. The most common value was 0004 Mbps. Segment download size was between 0 - 17.85 seconds. Segment size ranged from 83374 bytes to 1347055 bytes. Metrics recorded for the buffer-based algorithm are in Fig. 2.

The dynamic algorithm had a consistent bitrate but the rest of the metrics were inconsistent. Bitrate was consistently 542 Kbps. The values for buffer level were between 0.026 - 180.82 seconds. The highest buffer level values were in the first 15 collection intervals. After that, there was a pattern of the buffer levels going to around 0.1 seconds and then up to 1,2, or 6 seconds. Throughput ranged from 0.0004 to 3.3079 Mbps. The highest measurements for throughput were in the first and second measurements. Segment download time was between 0.001 seconds and 17.85 seconds. The most common time is 13.83 seconds. Segment size ranged from 78982 to 1347055 bytes. It was not uncommon for the segment size to be around 1 million bytes. Metrics for the dynamic algorithm are in Fig. 3.

IV.Comparison

The buffer-based and dynamic algorithms had identical results for bitrate and throughput based had different results. As shown in Fig. 4, the bitrate for the buffer-based algorithm and the dynamic algorithm was 542 Kbps. The throughput-based algorithm had a significantly higher bitrate with 1628 Mbps.

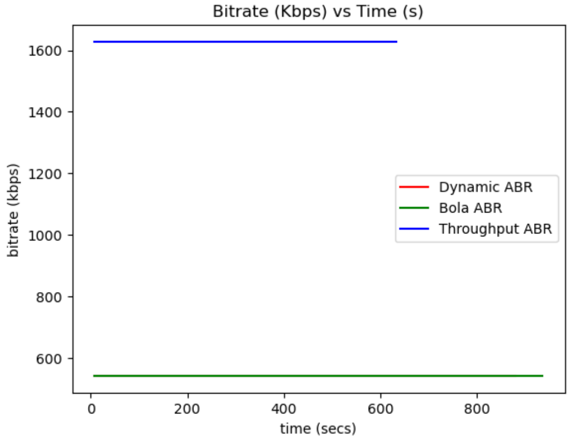

Fig. 4. Bitrate comparisons for Dynamic, Bola and Throughput ABRs.

For all three algorithms, the value of the bitrate did not change during the duration of the video. Dynamic and buffer-based algorithms had a similar pattern in the buffer level. The values were not identical but both had a high peak and then a large drop (see Fig. 5). Following this, there were many smaller peaks and drops.

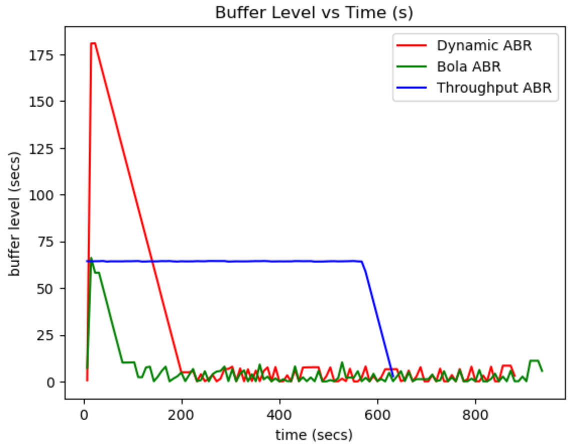

Fig. 5. Buffer level comparisons for Dynamic, Bola and Throughput ABRs.

Each algorithm had a very different pattern for throughput. Dynamic started with the highest value and had a significant drop. After this drop, the throughput value was consistent and always the lowest compared to the other two algorithms (See Fig. 6). Buffer-based started with a low value and increased as the video played. Midway through the playback the throughput stayed consistent until the end when it dropped sharply. The throughput-based algorithm had a number of peaks and troughs. The highest value was seen after 100 seconds of video playback. The lowest value in this algorithm was seen mid-way through playback, around 300 seconds. The dynamic ABR ended at its lowest throughput. Buffered-based ABR ended on its highest throughput. Throughput ABR ended in the middle of its peak and trough levels. The values were significantly smaller for dynamic ABR compared to the other two algorithms.

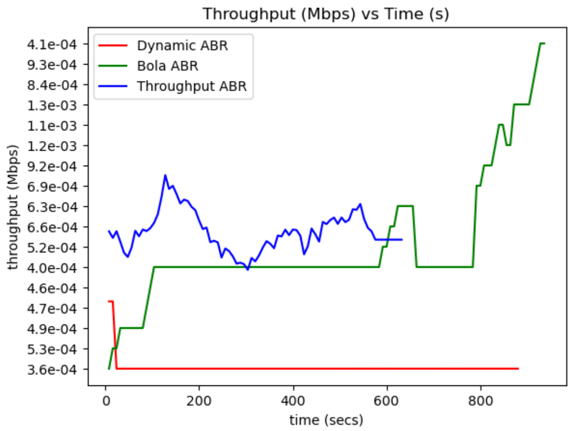

Fig. 6. Throughput comparisons for Dynamic, Bola and Throughput ABRs.

Throughput-based had the longest download times while buffer based and dynamic had lower values as seen in Fig. 7. All three algorithms had many peaks and troughs in download time. Throughput based took the shortest amount of time to play the video at its expected playback time compared to the other two algorithms which will be discussed further in the discussion.

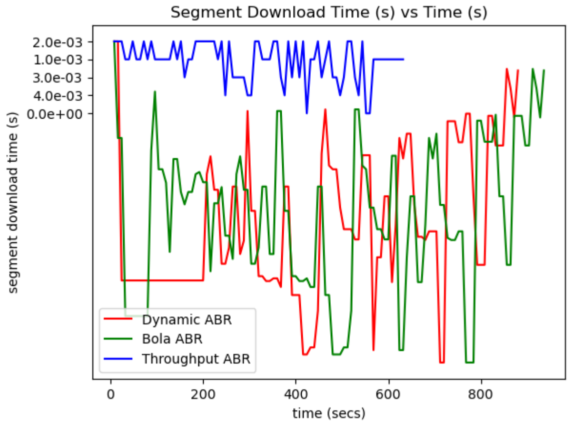

Fig. 7. Segment download time for Dynamic, Bola and Throughput ABRs.

Throughput-based had the largest segment sizes as shown in Fig. 8. This can be correlated back to the segment download time. The largest segment size was in the throughput-based algorithm. Throughput had the most variation in segment size. Buffer-based and dynamic had the least amount of changes in segment size. Their graphs were very similar, but the peaks were offset by 200 seconds between them.

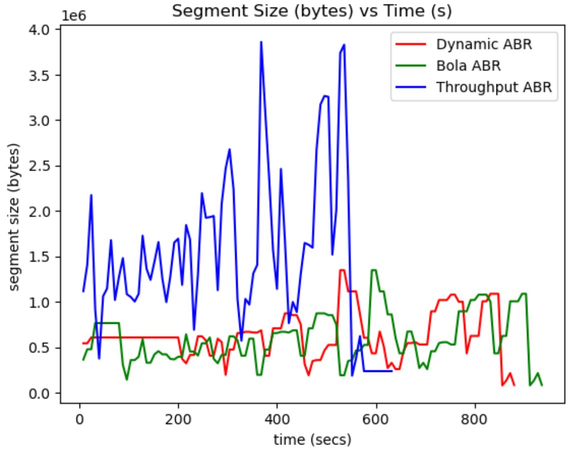

Fig. 8. Segment size for Dynamic, Bola and Throughput ABRs.

V.Discussion

The experiment aimed to compare the efficiency of the adaptive bitrate algorithms of HTTP Adaptive Streaming applications based on dash.js, namely throughput-based, buffer-based, and dynamic algorithms. The metrics collected were bitrate, buffer level, throughput, segment download time, and segment size. The results showed that each algorithm had its strengths and weaknesses, and there was no one-size-fits-all approach to delivering video content.

The throughput-based algorithm had a consistent bitrate, but throughput, segment download time, and segment size varied. This algorithm estimated the available bandwidth and picked the best bitrate that was less than or equal to that value. The buffer-based algorithm had a consistent bitrate, but the rest of the metrics varied. This algorithm determined which quality to download based on the available space in the buffer. The dynamic algorithm had a consistent bitrate, but the other metrics were inconsistent. This algorithm used a combination of throughput-based and buffer-based algorithms.

Overall, the experiment highlighted the importance of adaptive bitrate algorithms in delivering video content over the internet. The selection of the algorithm depended on various factors such as network conditions, available bandwidth, buffer size, and user preferences. The experiment provided valuable insights into the strengths and weaknesses of each algorithm and demonstrated the need for further research to improve the performance of adaptive bitrate algorithms.

VI.Conclusion

In conclusion, the experiment demonstrated that there is no single adaptive bitrate algorithm that performs optimally in all network conditions. The selection of the algorithm depended on various factors, and it was essential to evaluate each algorithm's strengths and weaknesses before selecting the appropriate one. The experiment provided valuable insights into the performance of different adaptive bitrate algorithms, and further research in this area could lead to the development of better algorithms to deliver high-quality video content over the internet.

References

[1] “Dash Reference Client 4.5.0,” Dash javascript player. [Online]. Available: https://reference.dashif.org/dash.js/v4.5.0/samples/dash-if-reference-player/index.html. [Accessed: Jan-2023].

[2] V. S. Vinnakota, “A tl;dr on ABR algorithms in MPEG DASH,” LinkedIn. [Online]. Available: https://www.linkedin.com/pulse/tldr-abr-algorithms-mpeg-dash-vijaya-sagar-vinnakota/. [Accessed: 24-Feb-2023].

[3] “Module: Dashmetrics,” dash.js Module: DashMetrics. [Online]. Available: https://cdn.dashjs.org/latest/jsdoc/module-DashMetrics.html. [Accessed: Feb-2023].