ECE4016 Assignment 2 Report

Jingquan Li
Student ID: 119010148

November 10, 2024

1 Algorithm Analysis

The BOLA (Buffer Occupancy based Lyapunov Algorithm) is designed to balance video quality and rebuffering probability by dynamically selecting the video bitrate based on the buffer level. The objective of BOLA is to maximize a utility function that considers the video quality (bitrate) while minimizing interruptions (rebuffering events).

In each time slot t_k, the algorithm evaluates the potential quality of each bitrate and its impact on the buffer level. The decision-making criterion is based on maximizing a specific metric, $\rho(t_k, \mathbf{a}(t_k))$, defined as follows:

$$\rho(t_k, \mathbf{a}(t_k)) = \begin{cases} 0, & \text{if } \sum_{m=1}^{M} a_m(t_k) = 0, \\ \frac{\sum_{m=1}^{M} a_m(t_k)(Vv_m + V\gamma p - Q(t_k))}{\sum_{m=1}^{M} a_m(t_k) S_m}, & \text{otherwise.} \end{cases}$$

Here, $a_m(t_k)$ is a binary variable that indicates whether bitrate m is selected, v_m represents the utility derived from the bitrate, S_m is the segment size associated with bitrate m, $Q(t_k)$ is the buffer level at time t_k, V is a control parameter, and γp is a term that controls the rebuffering sensitivity.

The decision rule for selecting the next segment's bitrate is as follows: 1. If $Q(t_k) > V(v_m + \gamma p)$ for all $m \in \{1, 2, \dots, M\}$, the algorithm chooses not to download any segment, i.e., $a_m(t_k) = 0$ for all m, and sets $T_k = \Delta$, where Δ is the delay until the next opportunity to download. 2. Otherwise, the algorithm selects the bitrate m^* that maximizes the ratio

$$\frac{Vv_m + V\gamma p - Q(t_k)}{S_m}$$

among all m for which this ratio is positive.

In this implementation, the utility function v_m I used in my algorithm is based on a sigmoid function rather than a logarithmic function mentioned in the paper example. The sigmoid utility function can provide a smooth increase in utility with the bitrate, capped at a maximum level to avoid extreme values, which helps in stabilizing the video quality.

The BOLA algorithm, by focusing on buffer occupancy rather than explicit bandwidth prediction, provides robustness in adapting to fluctuating network conditions. The use of a control parameter V allows the algorithm to balance between higher quality and reduced rebuffering risk, making it suitable for practical streaming environments.

2 Implementation

The implementation uses a BOLA-inspired algorithm to select the optimal bitrate based on buffer occupancy, available bitrates, and control parameters. Key parameters include `v` (0.93) to balance quality and fluency, `gamma` (1) for smoothing, and `p` (5) to influence bitrate selection based on buffer levels.

The function `student_entrypoint` organizes available bitrates, sorts them in descending order, and uses `choose_optimal_bitrate` to select the best bitrate. If no optimal bitrate is found, the algorithm defaults to the lowest bitrate.

A sigmoid function calculates utility:

$$\text{utility} = \frac{1}{1 + e^{-k\cdot(\text{bitrate\_value} - S_0)}}$$

where k and S_0 control the curve's steepness and midpoint. This utility helps ensure smooth quality changes.

The algorithm checks for a no-download condition by comparing the buffer occupancy (in chunks) against a threshold:

$$\text{threshold} = v \cdot (\text{utility} + \gamma \cdot p)$$

If this condition is met, no chunk is downloaded.

For each bitrate, a score is calculated:

$$\text{score} = \frac{v \cdot \text{utility} + v \cdot \gamma \cdot p - \text{buffer\_chunks}}{\text{bitrate\_value}}$$

The bitrate with the highest score is chosen, optimizing for quality while minimizing rebuffering risk.

3 Evaluation

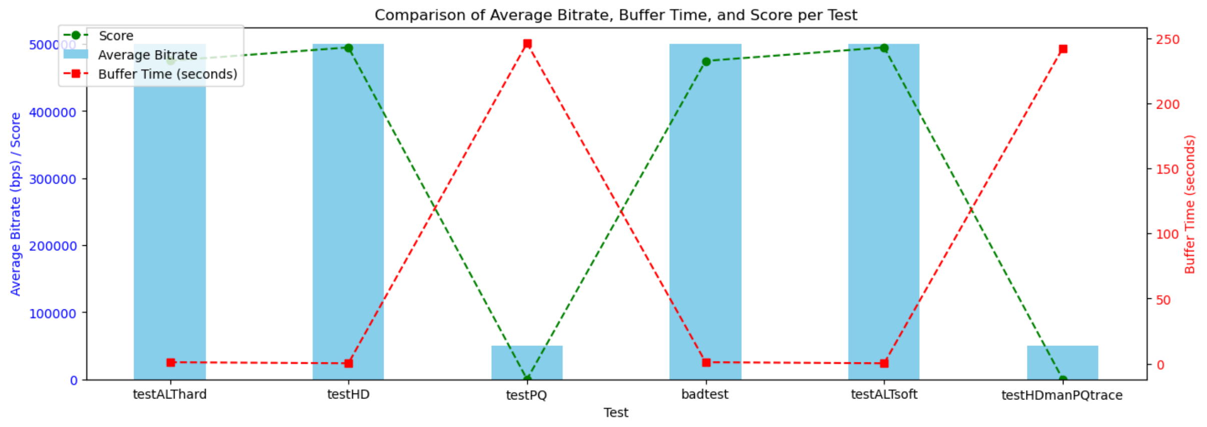

Figure 1: Comparison of Average Bitrate, Buffer Time, and Score across Different Test Cases

The performance of the adaptive bitrate selection algorithm was tested in several scenarios: `testALThard`, `testHD`, `testPQ`, `badtest`, `testALTsoft`, and `testHDmanPQtrace`.

I evaluated the algorithm based on three metrics: average bitrate, buffer time, and overall score. The results are shown in Figure 1.

The data in Figure 1 provides insights into how well the algorithm performed under different conditions:

- **Average Bitrate:** In most scenarios, the algorithm chose an average bitrate of 500,000 bps, indicating good video quality. However, in the `testPQ` and `testHDmanPQtrace` cases, the average bitrate dropped significantly to 50,000 bps. This shows that the algorithm adjusted to tougher network conditions by selecting a lower bitrate, which can help to reduce buffering.

- **Buffer Time:** Buffer time (the time spent loading video) varied widely across the tests. For example, in `testHD` and `testALTsoft`, the buffer time was low, around 0.2 seconds, indicating smooth playback with minimal interruptions. In contrast, buffer time was extremely high (over 240 seconds) in `testPQ` and `testHDmanPQtrace`, suggesting the algorithm struggled to keep the video playing smoothly in these conditions.

- **Score:** The score combines the algorithm's ability to maintain video quality and avoid buffering. Higher scores indicate better performance. In tests like `testHD` and `testALTsoft`, the algorithm achieved high scores (close to 500,000), meaning it maintained both good quality and smooth playback. However, in `testPQ` and `testHDmanPQtrace`, the scores were close to zero, reflecting the high buffer times and low bitrates seen in these tests.

Overall, the algorithm works well in stable network conditions, such as in `testHD` and `testALTsoft`, where it achieved high video quality and low buffer time. In challenging network scenarios, like `testPQ` and `testHDmanPQtrace`, the algorithm struggled, leading to long buffering times and lower video quality.

This evaluation suggests that while the algorithm can adapt to network changes by lowering the bitrate in poor conditions, it may need further tuning to handle unstable networks more effectively. Improving the algorithm's buffer management might help achieve smoother playback and more consistent quality in a wider range of situations.