The Impact of GEO Satellite Latency on Twitch Live Streams

Ziv Weissman
Computer Science Department
University of California, Santa Barbara
ziv@ucsb.edu

Jiamo Liu*
Viasat, Inc.
Carlsbad, CA
jiamo.liu@viasat.com

Elizabeth Belding
Computer Science Department
University of California, Santa Barbara
ebelding@ucsb.edu

***Abstract*—Live video streaming is widespread and a heavy consumer of Internet bandwidth. As such, it is important to evaluate the quality of that streaming, particularly over links that may pose challenges to near-real-time content delivery. In this paper, we study the performance of Twitch, one of the leading live streaming platforms, over Geosynchronous Earth Orbit (GEO) satellite networks; GEO networks are a key technology for connecting users in challenging environments, yet they suffer from high latency. To do so, we conduct controlled experiments that compare Twitch live stream performance on a high-latency GEO network to that on a low-latency campus network. We analyze core quality of experience metrics – resolution, frames per second, rebuffering, and playback delay – to pinpoint how satellite-induced delays disrupt streaming quality. Our findings reveal a critical flaw in Twitch's client scheduling: chunk request intervals are not calibrated to accommodate GEO network latency. As a result, playback buffers deplete before the next video chunk arrives, triggering frequent rebuffering, increased latency, and a notable deterioration in quality of experience. By highlighting this gap, our work underscores the urgency of latency-aware streaming strategies and adaptive scheduling algorithms. These insights offer actionable guidance for platform developers, satellite ISPs, and researchers, ultimately paving the way for more robust, inclusive live stream experiences as the medium's popularity and influence continue to climb.**

***Index Terms*—Live Video Streaming, Geosynchronous Satellite Networks, High-Latency Environments, Adaptive Bitrate Algorithm Evaluation, QoE Measurement**

I. Introduction

Geosynchronous (GEO) satellite networks are a crucial component of the Internet ecosystem due to their ability to extend Internet access to challenging connectivity environments where traditional infrastructure is impractical or costly. This includes remote and underserved areas around the world, as well as aircraft and sea vessels. Simultaneously, the live streaming industry has grown rapidly, driven by platforms such as Twitch, YouTube Live, and even traditional streaming services such as Netflix and Prime Video, which have recently launched live event streaming features [1]. This expansion reflects a broader global trend, with the live streaming market projected to reach a value of more than $247 billion by 2027, in large part due to increasing demand for interactive, real-time experiences in gaming, entertainment, education, and business [2]. In the U.S. alone, Twitch has become a central part of this ecosystem, boasting over 33 million active users, many of whom engage with live streams multiple times each week [3].

Given the important role that GEO networks play in internet access and the rise of live streaming, it is essential to evaluate how live streaming performs over these satellite connections. Although GEO satellites provide extensive coverage, they introduce significant latency due to their geosynchronous earth orbit, typically resulting in a 600 ms round-trip delay [4]. Unlike the low delay in terrestrial connections, this latency has the potential to present challenges for real-time applications such as live streaming, where the algorithms are typically inherently intolerant of high latency connections.

Quality of Experience (QoE), as a measure of user satisfaction, is influenced by various Key Performance Indicators (KPIs). On live streaming platforms such as Twitch, these KPIs include video quality, smooth playback, interactivity, and responsiveness, all of which are critical to maintaining high viewer engagement. To this end, Twitch minimizes delays between the broadcaster and the viewer to enable real-time, high quality interactions that promote user retention. However, when streaming algorithms are designed for low-latency delivery, there can be unintended consequences for high-latency connections because this latency is not taken into account in the algorithm design.

In this study, we systematically measure and analyze the QoE of Twitch live streaming over GEO satellite (high-latency) networks and compare the performance with that of campus (low-latency) networks, looking for key differences to understand how GEO network latency affects the behavior of Twitch's streaming algorithm. Specifically, this study addresses the following research questions:

- How does the high latency inherent in GEO satellite networks affect the QoE of Twitch live streams?
- What behaviors of Twitch's streaming algorithm contribute to QoE degradation in high-latency environments?
- What strategies can be implemented to enhance live streaming performance over high-latency networks such as GEO satellites?

By answering these questions, our goal is to provide insight that can help improve live streaming services in high-latency network environments, such as through informing the development of latency-aware algorithms and optimizations.

*Jiamo Liu was a Ph.D. student at the University of California, Santa Barbara when the work was performed.

II. Related Work

Previous video QoE research for terrestrial networks has emphasized on-demand platforms such as YouTube and Netflix, measuring KPIs such as resolution and rebuffering in varying network conditions [5]–[10]. These studies assume low latency and focus on buffering mechanisms and, hence, do not yield insight into the performance of latency-sensitive live streaming over long latency links.

Other recent studies have explored live streaming over satellite networks, particularly integrated terrestrial-satellite systems, which demonstrate QoE improvements through architecture-level optimizations [11]–[13]. However, these studies often overlook the client-side algorithmic behavior critical to platforms such as Twitch. LEO satellite studies highlight low-latency live streaming feasibility using technologies like DASH [14], but the findings are inapplicable to GEO networks due to significantly higher delays. Other work on GEO satellite QoE, such as traffic shaping and dataset-based prediction models, primarily focuses on on-demand platforms or general QoE impacts, rather than the unique latency sensitivity of live streaming [15], [16]. Although strategies like bandwidth aggregation and edge computing improve streaming in low-latency contexts, they do not address the challenges of high-latency GEO connections [17].

Our work fills the gap in prior work by investigating Twitch's live streaming algorithms when subjected to the latencies inherent in GEO satellite networks, identifying flaws in client-side adaptation and proposing latency-aware improvements to improve QoE in such high-latency environments.

III. Methodology and Data Collection

To investigate how GEO network latency impacts Twitch streaming QoE, we conducted controlled experiments comparing a high-latency GEO satellite network with a low-latency campus network, both shaped to identical bandwidth rates to isolate the effect of the high latency.

A. Experimental Setup

We used the same Lenovo ThinkPad P14s laptop (running Ubuntu 22.04) for both experiments to ensure consistency. The campus network provided high throughput (>100 Mbps) with 3.5 ms round trip time (RTT) to Twitch servers, while the satellite connection introduced 627.5 ms RTT via a GEO satellite link. Figure 1 illustrates our testbed configuration.

In sequential experiments, both networks were shaped to both the 3 Mbps and 5 Mbps rates—chosen based on FCC recommendations; 3 Mbps supports standard-definition and 5 Mbps supports HD video streaming [18]. This standardization isolated the impact of latency from bandwidth effects. For the satellite connection, bandwidth shaping was performed at the connection level by the ISP. For the campus network, we used Mahimahi [19] for interface-level shaping.

We collected HTTP-level data using browser proxies that captured HTTP Archive (HAR) files during Chrome sessions. HAR files contain detailed records of all HTTP transactions between the browser and servers, providing insight into chunk-level network behavior and timing for each video stream.

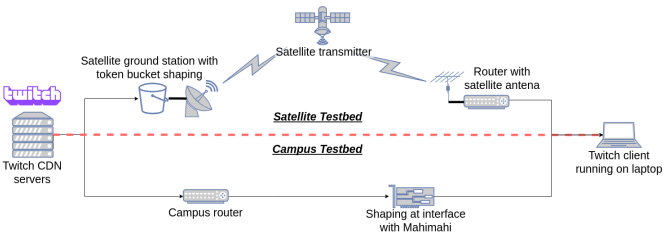


Fig. 1: Campus and satellite network testbed configuration.

B. Data Collection Process

We automated data collection using Python scripts with Selenium and ChromeDriver to stream four-minute videos while scraping Twitch's "Advanced Video Stats" feature every 100 ms. This feature is a utility that displays detailed playback statistics on the front end—similar to YouTube's "Stats for Nerds"—allowing for real-time monitoring of stream quality. We collected 500 streams total: 250 per network type, with 125 streams each at the 3 Mbps and 5 Mbps rates. To ensure content diversity, we sampled the top five streams from 25 different Twitch categories across genres including gaming, simulation, and lifestyle content. Error handling ensured successful data collection by automatically selecting alternative streams when issues such as age restrictions or video player errors occurred.

C. Metrics and Analysis

The metrics collected from the web-player's "Advanced Video Stats" include:

- **Vertical Pixel Resolution**: Video vertical resolution.
- **Frames Per Second (FPS)**: Video stream frame rate.
- **Rebuffering Status**: Flag indicating rebuffering events (FPS of zero or rebuffering wheel present).
- **Latency to Broadcaster**: Total delay from broadcaster to viewer, including rebuffering delays.

From HAR files, we extracted chunk-level metrics, as illustrated in Figure 2:

- **Chunk Download Time**: Duration from request start to completion.
- **Chunk Size**: Size of each downloaded video chunk.
- **Chunk Latency**: Time from request to first byte received.
- **Inter-Chunk Request Interval**: Time between consecutive chunk requests.
- **Pre-Request Completion Fraction**: Percentage of chunks fully loaded before next request.
- **Effective Chunk Throughput (*TNetwork*)**: Chunk size divided by download time.

For each metric, we calculate key statistical measures for each stream, including mean, median, variance, minimum, and maximum values. We primarily use the mean values in our analysis as those were found to be most informative. In

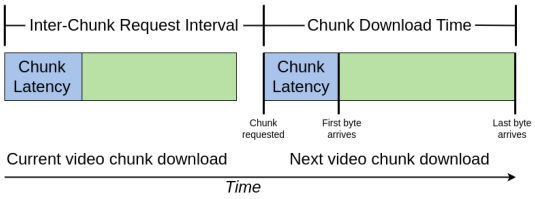


Fig. 2: Diagram of chunk level metrics

distributions and graphs, when we plot the mean of a statistic and there are multiple points, we use "mean" to refer to the mean of that statistic for each stream in the subset of data shown in the plot.

D. Limitations of Bandwidth Shaping Methods

Our network shaping methods differed: Mahimahi performed interface-level shaping on the campus network, thereby shaping all traffic passing through the network interface, including background processes and system activities. As a result, some of the allocated bandwidth intended for the Twitch streams was consumed by these background tasks, leading to slightly lower effective bandwidth—quantified in section V-A—compared to the intended shaping rates; this could, in turn, theoretically degrade the QoE of the campus network. In contrast, the satellite connection used token bucket shaping, allowing brief bandwidth bursts up to 10% above the nominal rate, which can benefit bursty applications like video streaming.[1] However, despite the campus network's shaping disadvantages, it still outperformed the satellite network, indicating that high latency—not throughput limitations—primarily drives the QoE degradation we observe.

IV. Live Stream Performance

We compare QoE metrics between the satellite and campus networks to illustrate performance differences and highlight challenges GEO satellite networks face in delivering consistent live streaming quality.

Resolution and FPS Comparisons. Figure 3 shows video resolution distributions. While the satellite connection tended to achieve slightly higher video resolution than the campus network (due to connection-level vs. interface-level bandwidth shaping), both connections achieved similar FPS distributions at each bandwidth rate. However, these occasional satellite advantages in resolution did not translate to better QoE. As shown in the following analysis, the GEO satellite connection suffers from significantly higher rebuffering and latency.

Rebuffering and Latency. Figure 4 shows that satellite streams experience substantially more rebuffering. Campus networks had median rebuffering times of 14.24 and 7.75 seconds for 3 and 5 Mbps, respectively, while satellite networks had 22.69 and 27.62 seconds—nearly triple at 5 Mbps.

[1]The satellite ISP, Viasat, does not implement bandwidth shaping in production due to performance issues arising from the interplay between high latency and the client-side playback algorithm.

Figure 5 shows similar trends for latency to broadcaster. Campus networks averaged 8.54 and 8.38 seconds for 3 and 5 Mbps, while satellite networks averaged 17.14 and 22.54 seconds, respectively.

We observe a strong linear correlation between rebuffering time and reported latency to broadcaster (Figure 6). When streams rebuffer, they freeze and resume from where they left off, increasing reported latency by the rebuffering duration. This trend continues until approximately 45 seconds of rebuffering, after which chunks are deleted from CDN servers, causing indefinite rebuffering unless manually reloaded.

Takeaways. Despite achieving higher resolution, Twitch delivers significantly worse QoE in GEO networks. The satellite connection experiences substantially higher rebuffering times and latency to broadcaster. This performance degradation, despite adequate throughput, suggests that factors other than bandwidth—specifically network latency—primarily impact streaming performance. In the next section, we explore the underlying causes of these performance issues.

V. Explaining Poor QoE

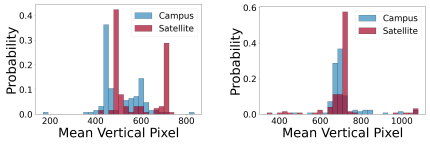
While the Twitch streaming algorithm is mostly hidden from the user, we were still able to find and identify key faults in the chunk scheduling by examining the HAR files. These faults largely explain the poor QoE. In this section, we delve deeper into the QoS of each network type to understand the QoE discrepancies. We find that Twitch's chunk scheduling algorithm has a critical flaw that leads to inefficient streaming over high-latency networks.

A. Throughput is Not Limiting

An initial assumption might be that the poor QoE observed on the satellite network is due to throughput limitations caused by the high latency, which in turn impact TCP performance. To assess this, we analyzed the effective throughput achieved during video chunk downloads using the *TNetwork* metric. This metric, as described in section III-C, represents the average throughput achieved for a single video chunk, calculated over the duration of the entire download.

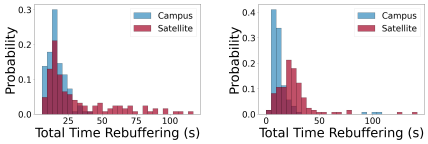
Figure 7 presents a histogram of the *TNetwork* values for the satellite and campus networks at the studied shaping rates. The distributions reveal that the GEO network consistently achieves higher effective throughput than the campus network. As described in section III-D, shaping at the interface as implemented in our campus network leads to a lower effective throughput than we explicitly set. On the other hand, the token-bucket shaping used by the satellite ISP enables higher achieved throughput for the video chunks in the video streams.

These observations confirm that the poor QoE experienced on the satellite connection is not due to throughput limitations. Despite the increased latency, the TCP protocol effectively utilizes the available bandwidth during chunk downloads (likely thanks to the TCP proxies implemented by the ISP). The high latency does not hinder the network's capacity to achieve throughput close to the shaping rate. Therefore, the degradation in QoE on the satellite network must be attributed to other



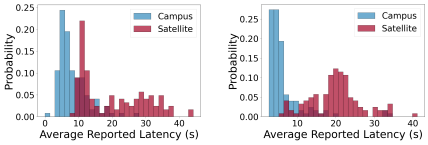
(a) 3 Mbps (b) 5 Mbps

Fig. 3: Distribution of video resolution.



(a) 3 Mbps (b) 5 Mbps

Fig. 4: Distribution of total time rebuffering.



(a) 3 Mbps (b) 5 Mbps

Fig. 5: Distribution of average reported latency.

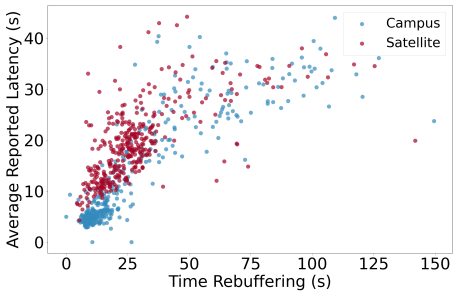


Fig. 6: Relationship between time rebuffering and average reported latency to broadcaster.

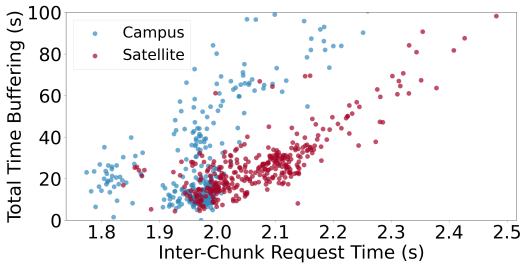
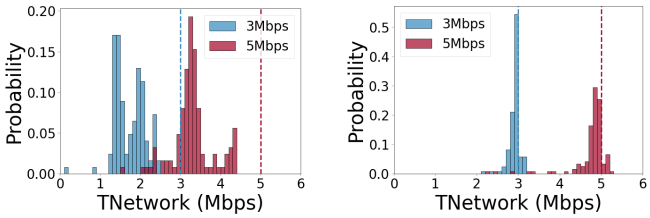


Fig. 8: Relationship between the average inter-chunk request time per stream and the total time spent rebuffering.



(a) Campus network (b) Satellite network

Fig. 7: Distribution of mean TNetwork for chunks in a stream.

factors. In the subsequent sections, we explore how increased latency affects the chunk scheduling algorithm used by the Twitch client, leading to inefficient streaming performance in high-latency environments. Specifically, we examine how the timing of chunk requests relative to their playback rate impacts the streaming buffer and overall viewing experience.

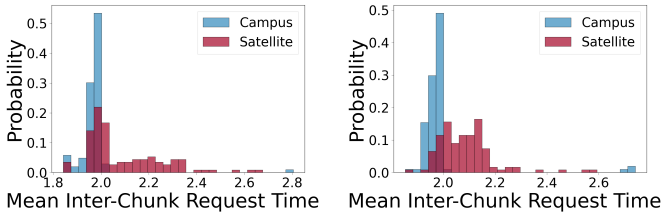
B. Chunk Request Interval is Greater than Playback Rate

The primary observation explaining the poor Twitch QoE over the satellite connection is that chunks are requested less frequently than they are played back. Since Twitch's video chunks are not encrypted, we were able to intercept them during streaming using Chrome's developer tools. We observed that the live stream is delivered to the end host in two-second video clips.

For smooth and uninterrupted playback, the client needs to request new chunks at an average rate that matches the playback rate—on average, every two seconds. If the average time between chunk requests exceeds two seconds, the client consumes video data faster than it receives new data. This mismatch causes the playback buffer to gradually deplete because new chunks do not arrive quickly enough to replace those being played. As a result, the buffer eventually empties, leading to rebuffering stalls. The more the average inter-chunk request time exceeds two seconds, the faster the buffer depletes and the more frequent these interruptions become. For example, if chunks are requested every 2.1 seconds on average, the client falls behind by 0.1 seconds with each chunk. Over time, this small delay accumulates, increasing the likelihood of rebuffering. If the average request time stretches even higher, the deficit grows more quickly, leading to more frequent and longer pauses in playback.

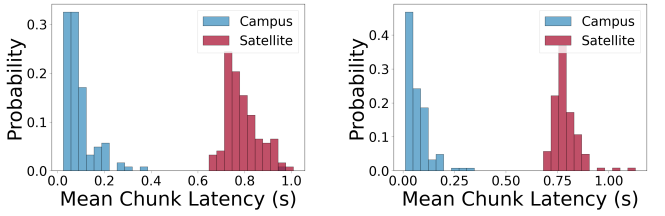
To verify this relationship using our experimental data, we calculated the mean inter-chunk request time and the total time spent rebuffering for each stream we collected. We plot these values in Figure 8. The plot shows a sharp increase in the total rebuffering time when the average inter-chunk request time exceeds two seconds. This indicates that surpassing the chunk's playback duration in request intervals directly contributes to buffer depletion and increased rebuffering events. Maintaining an average inter-chunk request interval that matches or is less than the chunk playback rate is essential to prevent these issues and to ensure a smooth live-streaming experience.

Figure 9 shows the inter-chunk request times for the two networks. Due to factors that are explained in section V-C, the satellite connection tends to have significantly greater inter-chunk request times compared to the campus network connection. This disparity suggests that Twitch's streaming algorithm does not properly compensate for increased latency, resulting in increased time between chunk requests with higher latency. The delayed chunk requests accelerate buffer depletion and increase rebuffering events.



(a) 3 Mbps (b) 5 Mbps

Fig. 9: Distributions of the middle 99% of mean inter-chunk request times.



(a) 3 Mbps (b) 5 Mbps

Fig. 10: Mean chunk latency for satellite and campus connections.

C. Effect of Latency on Inter-Chunk Request Time

The achieved throughput for video chunk downloads over the GEO network can be comparable to—or even exceed—that of the campus network. However, despite this adequate throughput, the inter-chunk request times (as explained in section III-C) are significantly longer on the GEO satellite connection compared to the campus connection; this was illustrated in Figure 9.

To investigate this discrepancy, we analyze the chunk latency (calculated in section III-C). Figure 10 presents a comparison of the mean chunk latency for streams on the two network types. The distributions reveal that the GEO satellite network experiences significantly higher latencies, with a median mean chunk latency of 0.77 seconds, compared to 0.06 seconds for the campus network.

Upon analyzing the chunk scheduling algorithm used by Twitch, we discovered that the Twitch client waits for a chunk to be fully downloaded before requesting the next one. This behavior implies that any increase in chunk latency directly extends the inter-chunk request time; the client remains idle until the current chunk has completely loaded.

To quantify this effect, we calculated the pre-request completion fraction of chunks in the stream; this calculation is detailed in section III-C. On the satellite network, an average of 98.51% of the chunks were fully loaded before the next chunk request was made. Similarly, for the campus network, the average was 99.17%. However, due to the higher chunk latency on the GEO satellite network, the waiting period significantly extends inter-chunk request times, reducing effective network utilization and unnecessarily causing the average inter-chunk

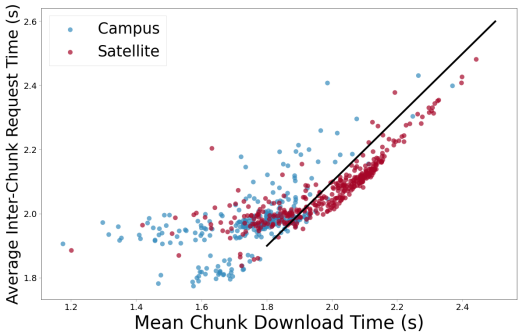


Fig. 11: Mean chunk download time vs. average inter-chunk request time for the satellite and campus networks. The diagonal line shows a linear one-to-one relationship.

request interval to exceed the two-second critical point. In contrast, on the low latency campus network, the waiting period after each chunk download is minimal, and the inter-chunk request times remain below the two-second playback duration. Therefore, the client can maintain a stable buffer, ensuring smoother playback.

The relationship between chunk download times and inter-chunk request times is further illustrated in Figure 11. The scatterplot shows a clear correlation between the average chunk download time and the inter-chunk request time for both connection types. While the campus network maintains inter-chunk request times around or below the critical two-second threshold, even with varying chunk download times, the satellite network inter-chunk request times consistently exceed two seconds as chunk download times increase.

As Figure 11 shows, the GEO satellite network tends to experience much higher inter-chunk request times compared to the campus network, which correlates with the higher chunk download times due to the increased latency. For GEO satellite connections, inter-chunk request times are often above two seconds, the critical threshold for uninterrupted playback. This behavior is a direct result of waiting for each chunk to be fully loaded before requesting the next one, a process that is especially inefficient in high-latency environments.

Takeaways. Our analysis shows that the primary reason for lower QoE on the GEO satellite connection is the client's chunk scheduling algorithm failing to account for high latency. Specifically, the Twitch client aims to optimize QoE by maximizing playback bitrate and frame rate, selecting high-quality chunks that saturate the available bandwidth for a two second download period. However, in high-latency environments, this strategy leads to chunk download times approaching or exceeding the chunks' playback duration. Crucially, because the client waits until the current chunk is fully downloaded before initiating the next request and does not request the next chunk early enough to account for the increased latency, the inter-chunk request intervals exceed the chunks' playback duration. Consequently, the client consumes video data faster than it receives new chunks, causing the playback buffer to

deplete. Viewers experience this as a repetitive cycle: watching two seconds of video followed by a rebuffering pause. We confirmed this behavior by manually observing streams on the satellite connection, where the playback pattern matched our empirical findings.

VI. Towards Latency-Resilient Live Streaming

To address buffer depletion and frequent rebuffering events in high latency networks, a latency-aware scheduling mechanism that enforces a strict upper bound on inter-chunk request intervals offers a promising approach. Specifically, if a chunk downloads in less than two seconds, the client immediately requests the next chunk. However, if two seconds have elapsed since the previous request, the next request is issued at exactly the two-second mark, regardless of the current chunk's download status. This approach ensures that request intervals never exceed the chunk playback duration while maintaining responsiveness when network conditions permit. This design aims to stabilize the buffer in high-latency conditions without fully disassociating request timing from download progress.

However, challenges could arise when throughput fluctuates significantly. In mobile or congested networks, short-term bandwidth drops may push download times beyond two seconds, potentially prompting premature quality downgrades or momentary overlaps in chunk downloads. Addressing these challenges requires further evaluation in real-world conditions, where additional optimizations—such as integrating latency-aware ABR logic or leveraging network-side enhancements—may improve performance.

VII. Conclusion

In this paper, we investigated the impact of a high-latency GEO satellite connection on Twitch QoE, revealing inefficiencies in the chunk scheduling algorithm used by the Twitch client at the time of our experiments. Specifically, the client tries to saturate the available bandwidth but fails to account for latency, leading to inter-chunk request intervals that exceed the playback duration of the chunks. This in turn causes buffer depletion, rebuffering, and playback interruptions. In extreme cases, streams stall indefinitely when chunks are no longer retrievable from the CDN.

To enhance QoE for users on high-latency networks, Twitch and other live streaming services should ensure their chunk scheduling algorithms account for network latency. A refined solution is to employ a dynamic scheduling mechanism that enforces a strict two-second upper bound on the inter-chunk request interval—aligned with the chunk playback duration—while still allowing immediate requests when network conditions permit rapid downloads. As streaming platforms reduce chunk durations to achieve lower end-to-end latencies, even minor network delays can significantly impact performance, underscoring the necessity for streaming services to consider network latency in their algorithm designs, regardless of the absolute latency values.

Acknowledgments

The authors sincerely thank Viasat for supporting this work and providing access to the GEO satellite network connection used in our experiments.

References

[1] Demand Sage, "Live Streaming Statistics," [Online]. Available: https://www.demandsage.com/live-streaming-statistics/, 2024, [Accessed: Nov. 13, 2024].

[2] Market Research Future, "Live Streaming Market Research Report," [Online]. Available: https://www.marketresearchfuture.com/reports/live-streaming-market-10134, 2023, [Accessed: Nov. 13, 2024].

[3] Statista, "Twitch usage statistics," [Online]. Available: https://www.statista.com/topics/8906/live-streaming/, 2024, [Accessed: Nov. 13, 2024].

[4] Viasat Blog, "Unique Qualities of High-Capacity GEO Satellites," Available: https://news.viasat.com, 2024, [Accessed: Nov. 13, 2024].

[5] F. Loh, K. Hildebrand, F. Wamser, S. Geißler, and T. Hoßfeld, "Machine Learning Based Study of QoE Metrics in Twitch.tv Live Streaming," in *NOMS 2023-2023 IEEE/IFIP Network Operations and Management Symposium*. IEEE, 2023, pp. 1–7.

[6] T. Hoßfeld, M. Seufert, M. Hirth, T. Zinner, P. Tran-Gia, and R. Schatz, "Quantification of YouTube QoE via Crowdsourcing," in *Proceedings of the IEEE International Symposium on Multimedia*, 2011, pp. 494–499.

[7] F. Wamser, M. Seufert, P. Casas, R. Irmer, P. Tran-Gia, and R. Schatz, "YoMoApp: A Tool for Analyzing QoE of YouTube HTTP Adaptive Streaming in Mobile Networks," in *Proceedings of the European Conference on Networks and Communications (EuCNC)*, 2015, pp. 239–243.

[8] C. Gutterman, K. Guo, S. Arora, X. Wang, L. Wu, E. Katz-Bassett, and G. Zussman, "Requet: Real-Time QoE Detection for Encrypted YouTube Traffic," in *ACM MMSys '19*, 2019, p. 48–59.

[9] F. Bronzino, P. Schmitt, S. Ayoubi, G. Martins, R. Teixeira, and N. Feamster, "Inferring Streaming Video Quality from Encrypted Traffic: Practical Models and Deployment Experience," *ACM Transactions on Internet Technology*, vol. 3, no. 3, December 2019.

[10] F. Loh, F. Wamser, F. Poignée, S. Geißler, and T. Hoßfeld, "YouTube Dataset on Mobile Streaming for Internet Traffic Modeling and Streaming Analysis," *Scientific Data*, vol. 9, no. 1, 2022.

[11] C. Ge, N. Wang, I. Selinis, J. Cahill, M. Kavanagh, K. Liolis, C. Politis, J. Nunes, B. Evans, Y. Rahulan, N. Nouvel, M. Boutin, J. Desmauts, F. Arnal, S. Watts, and G. Poziopoulou, "QoE-Assured Live Streaming via Satellite Backhaul in 5G Networks," *IEEE Transactions on Broadcasting*, vol. 65, no. 2, pp. 381–391, 2019.

[12] S. Kumar, N. Wang, Y. Rahulan, and B. Evans, "QoE-Aware Video Streaming over Integrated Space and Terrestrial 5G Networks," *IEEE Network*, vol. 35, no. 4, pp. 95–101, 2021.

[13] ——, "Edge Computing-Based Layered Video Streaming Over Integrated Satellite and Terrestrial 5G Networks," *IEEE Access*, vol. 10, pp. 19 971–19 985, 2022.

[14] J. Zhao and J. Pan, "Low-Latency Live Video Streaming over a Low-Earth-Orbit Satellite Network with DASH," in *Proceedings of the 15th ACM Multimedia Systems Conference*, 2024, pp. 109–120.

[15] J. Liu, D. Lerner, J. Chung, U. Paul, A. Gupta, and E. Belding, "Watching Stars in Pixels: The Interplay of Traffic Shaping and YouTube Streaming QoE over GEO Satellite Networks," in *Proceedings of PAM*, 2024, pp. 153–169.

[16] B. Chen, Z. Shang, J. W. Chung, D. Lerner, W. Robitza, R. R. R. Rao, A. Raake, and A. C. Bovik, "Satellite Streaming Video QoE Prediction: A Real-World Subjective Database and Network-Level Prediction Models," Available: https://arxiv.org/abs/2410.13952, 2024, [Accessed: Nov. 13, 2024].

[17] C. Wu, Z. Wang, J. Liu, and S. Yang, "Improving Crowdsourced Live Streaming with Aggregated Edge Networks," [Online]. Available: https://arxiv.org/abs/1605.08969, 2016, [Accessed: Nov. 13, 2024].

[18] Federal Communications Commission, "Broadband Speed Guide," Available: https://www.fcc.gov/consumers/guides/broadband-speed-guide, 2022, [Accessed: Dec. 7, 2024].

[19] R. Netravali, A. Sivaraman, S. Das, A. Goyal, K. Winstein, J. Mickens, and H. Balakrishnan, "Mahimahi: accurate record-and-replay for HTTP," in *Proceedings of USENIX*, 2015, p. 417–429.