Robust Live Streaming over LEO Satellite Constellations: Measurement, Analysis, and Handover-Aware Adaptation

Hao Fang
Simon Fraser University
Burnaby, Canada
fanghaof@sfu.ca

Haoyuan Zhao
Simon Fraser University
Burnaby, Canada
hza127@sfu.ca

Jianxin Shi
Nankai University
Tianjin, China
Simon Fraser University
Burnaby, Canada
shijx@mail.nankai.edu.cn

Miao Zhang
Simon Fraser University
Burnaby, Canada
mza94@sfu.ca

Guanzhen Wu
Simon Fraser University
Burnaby, Canada
gwa52@sfu.ca

Yi Ching Chou
Simon Fraser University
Burnaby, Canada
ycchou@sfu.ca

Feng Wang
University of Mississippi
Oxford, United States
fwang@cs.olemiss.edu

Jiangchuan Liu*
Simon Fraser University
Burnaby, Canada
jcliu@sfu.ca

Abstract

Live streaming has experienced significant growth recently. Yet this rise in popularity contrasts with the reality that a substantial segment of the global population still lacks Internet access. The emergence of Low Earth orbit Satellite Networks (LSNs), such as SpaceX's Starlink and Amazon's Project Kuiper, presents a promising solution to fill this gap. Nevertheless, our measurement study reveals that existing live streaming platforms may not be able to deliver a smooth viewing experience on LSNs due to frequent satellite handovers, which lead to frequent video rebuffering events. Current state-of-the-art learning-based Adaptive Bitrate (ABR) algorithms, even when trained on LSNs' network traces, fail to manage the abrupt network variations associated with satellite handovers effectively. To address these challenges, for the first time, we introduce Satellite-Aware Rate Adaptation (SARA), a versatile and lightweight middleware that can seamlessly integrate with various ABR algorithms to enhance the performance of live streaming over LSNs. SARA intelligently modulates video playback speed and furnishes ABR algorithms with insights derived from the distinctive network characteristics of LSNs, thereby aiding ABR algorithms in making informed bitrate selections and effectively minimizing rebuffering events that occur during satellite handovers. Our extensive evaluation shows that SARA can effectively reduce the rebuffering time by an average of 39.41% and slightly improve latency by 0.65% while only introducing an overall loss in bitrate by 0.13%.

CCS Concepts

• **Information systems → Multimedia streaming**; • **Networks → Wireless access networks**.

*Corresponding Author

Keywords

Multimedia Services, Video Streaming, Low Earth Orbit Satellite Networks, Network Performance Measurement & Optimization

1 Introduction

Live streaming witnessed a significant 99% growth over the past three years,[1] currently engaging almost 30% of Internet users on a weekly basis.[2] While it has become integral to urban life, approximately 34% of the global population remains without Internet access.[3] Fortunately, the commercial success of Low Earth orbit Satellite Networks (LSNs) operators, notably SpaceX's Starlink and Amazon's Project Kuiper [30], presents a promising solution to bridge this gap.

Low Earth Orbit (LEO) satellites, orbiting significantly closer to the Earth than their Geosynchronous Orbit counterparts, offer reduced communication delays but with a much smaller coverage area.[4] Additionally, due to their faster orbital periods of 128 minutes or less, they only have a visible duration of 2 to 10 minutes from a fixed location on the Earth [2, 25]. Consequently, a constellation of LEO satellites, forming an LSN, is necessary to achieve comprehensive global coverage and provide seamless, high-quality Internet services [22, 44]. With a dense satellite constellation, LSNs can offer significant advantages, especially in areas where establishing terrestrial infrastructure is either challenging or cost-prohibitive. By resorting to LSNs, live streaming can fully unleash its potential in currently underserved areas, reaching a broader audience and offering more resilient services.
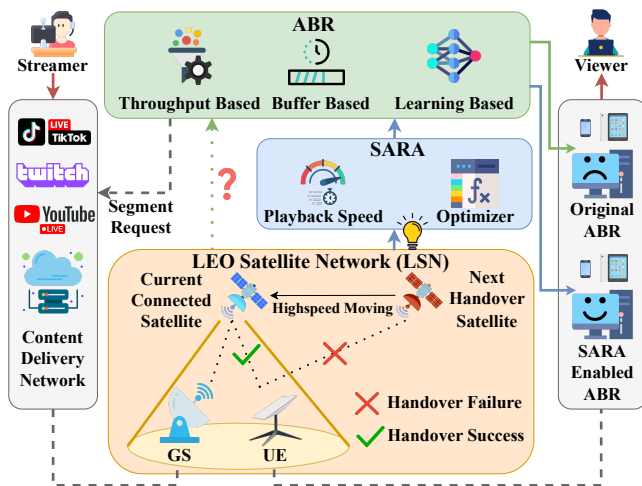
Figure 1: Overview of live streaming services over LSNs where SARA can be easily integrated with various ABR algorithms to enhance viewers' live streaming experience.

Yet, user equipment (UE) connected to LSNs, even stationary ones, inevitably experience frequent satellite handovers due to satellite motion. Recent measurement studies have shown that Starlink updates the UE-satellite link at a granularity of 15 seconds [3, 39]. Such frequent handovers can lead to network outages (loss of network connection) [17, 28, 45], adversely affecting live streaming services which rely on continuous connectivity. Live streaming services predominantly use HTTP adaptive streaming and HTTP Live Streaming protocol to deliver video content. At the heart of this technology are Adaptive Bitrate (ABR) algorithms. By selecting from various pre-encoded video quality levels, ABR algorithms can combat variations in underlying network conditions to ensure viewers receive high-quality and low latency video streams.

Nevertheless, these ABR algorithms are optimized for terrestrial networks and lack awareness of the sudden network disruptions caused by frequent satellite handovers. According to our measurements, *Twitch's* ABR algorithm struggles with the variable network conditions of LSNs, leading to video rebuffering events at the frequency of several minutes during Internet peak hours. Even using state-of-the-art learning based ABR algorithms trained directly with satellite network data, the streaming quality still struggles as these algorithms do not distinguish LSNs from such wireless terrestrial networks as WiFi or cellular networks, where the handovers are mostly attributable to user movements, leading to sporadic and infrequent occurrences and a more gradual degradation in network conditions. In contrast, LEO satellites usually move at the speed of 28,080 kilometres per hour,[5] which is two magnitudes higher than the typical moving speed of pedestrian or vehicular users in wireless terrestrial networks. Making those handovers in LSNs usually occurs at a much more frequent pace, and the consequent shifts in network conditions are not only much more abrupt but also typically accompanied by disruptions. Such fast-paced handovers in LSNs, coupled with the abruptness of network condition variations and disruptions, severely challenge the adaptability of current ABR algorithms, outpacing their capacity to adjust and

resulting in sub-optimal performance and diminished streaming experiences in the LSN context.

We strive to tackle this challenge and, for the first time, propose a versatile and lightweight middleware solution named Satellite-Aware Rate Adaptation (SARA), which can seamlessly work with various existing ABR algorithms, furnishing them with insightful information interpreted from specific network characteristics unique to LSNs, enhancing their adaptability, and enabling them to more effectively navigate and optimize streaming performance within this brand new type of network environment. The design of SARA is motivated by our preliminary measurements of the Starlink network, which focus specifically on its performance in the context of live streaming services and identifying key characteristics that can significantly impact users' viewing experience. As illustrated in Figure 1, SARA is explicitly crafted to enhance the performance of a variety of different types of ABR algorithms such as Robust Model Predictive Control (RobustMPC), Buffer Based Approach (BBA), and Pensieve [12, 26, 42] for live streaming services over LSNs where SARA can intelligently control the playback speed by both accelerating and decelerating the video and assist the ABR algorithm in selecting the suitable bitrate to avoid rebuffering events during satellite handovers. Our extensive evaluation shows that SARA can effectively reduce rebuffering time by an average of 39.41% while simultaneously maintaining both high bitrate and low latency for live streaming viewers.

Our primary contributions are as follows:

- We conduct a preliminary measurement study focusing on LSNs, using Starlink as a case study. The results, particularly on *Twitch*, reveal that existing ABR algorithms struggle to provide smooth viewing experiences under LSN conditions, often resulting in frequent video pauses ranging from 5.93 seconds to 23.57 seconds.
- By further in-depth analysis of Starlink's outage, we discover that the occurrence of outages surges by 150% during Internet peak hours compared to non-peak hours.
- Motivated by our measurement study, we propose for the first time a versatile middleware named SARA to dynamically adjust video playback speed and utilize throughput and buffer scalars to guide existing ABR algorithms with bitrate selection and combat network outages under LSNs.
- Our extensive evaluation shows that SARA can significantly reduce average rebuffering time by 39.41% at a negligible cost of an average loss in bitrate of 0.65%.

2 Measurements and Analysis

We first conduct a measurement on Starlink to help us understand the network characteristics and challenges associated with LSNs in the context of live streaming services. In this paper, we focus on the client side and evaluate the downlink performance. Measurements were conducted using two Starlink UEs located in distinct settings: one in an urban area and the other in a rural area of the Pacific West Coast, situated in a forest. To ensure unobstructed signals, we mount the rural area's UE at the top of a tree, approximately 50 meters high. The obstruction ratio for our UEs is 0.734% and 0.753%, respectively, as reported by the Starlink mobile app.[6] We collect our

---

[5]https://www.esa.int/ESA_Multimedia/Images/2020/03/Low_Earth_orbit

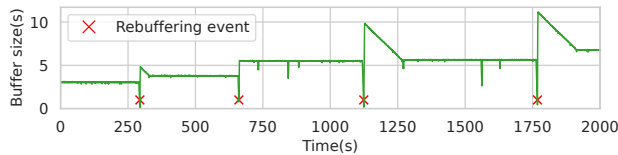[6]https://play.google.com/store/apps/details?id=com.starlink.mobile

Figure 2: A sequence of rebuffering events observed from *Twitch* on the Starlink network.
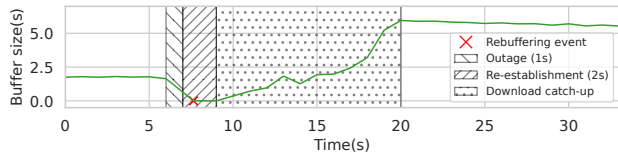

Figure 3: A typical rebuffering event, illustrating that short duration outages can still cause video rebuffering due to connection re-establishment in the application layer.

measurement data on clear days to avoid any performance degradation due to extreme weather conditions [24, 25]. The data collection commenced in late March 2023 and spanned approximately four months. We measured basic network data, including network delay and throughput. Additionally, we have collected Starlink outage history using the Starlink mobile app, which provides valuable insights into the frequency and duration of network disruptions. Beyond these general network measurements, we also evaluate the performance of *Twtich*, *TikTok LIVE* and *YouTube Live* in order to understand the performance of ABR algorithms under LSNs in the context of live streaming. Our preliminary measurements indicate that all three platforms yield comparable results. Hence, in this paper, we focus on the performance of *Twitch*, the predominant live streaming platform in North America [37].

2.1 Live Streaming Challenges in the LSNs

To evaluate *Twitch* ABR's performance, we choose a channel that streams content at a typical resolution of 1920×1080, frame rate of 35 FPS and bitrates around 6 Mbps. During our measurement, we observed a stark contrast in the number of rebuffering events and duration between LSN and terrestrial network. In terms of duration, rebuffering lasts an average of 5.93 seconds with a peak of 23.57 seconds for LSN, while for terrestrial networks, the maximum rebuffering time is only 4.64 seconds with an average of 2.97 seconds. In terms of frequency, we observed an average of 1.04 rebuffering events per hour with LSN and an average of 3.49 rebuffering events during Internet rush hours, with instances of up to 8 rebuffering events occurring within one hour. In comparison, terrestrial networks exhibit an average of 0.27 rebuffering events per hour, with no sequential rebuffering events observed within one hour.

Figure 2 provides a 30-minute snapshot depicting buffer size fluctuations on *Twitch*, which highlights the inadequacy of the existing ABR algorithms in LSNs. Notably, the average rebuffering duration, at 5.93 seconds, consistently exceeds the current Latency to Broadcaster (LtB), which is 3 seconds in the figure. LtB represents the time taken for content to travel from the streamer's side to the viewer's side and also serves as the maximum limit for the amount of video content that can be preloaded on the viewer's side. As a result, the buffer is consistently emptied after each network outage. Even when the ABR algorithm always selects the lowest bitrate
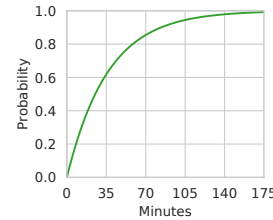

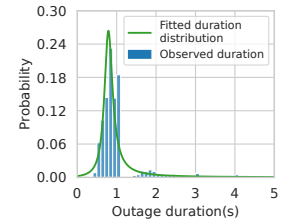**Figure 4: Probability of the first outage over time.**

Figure 5: Outage duration distribution.

to minimize rebuffering, completely avoiding such events remains difficult. We also noticed that the outage reported by the Starlink mobile app could lead to an extended rebuffering duration at the application level. Figure 3 zooms in one particular rebuffering event. As shown, there is a 1-second outage followed by a 2-seconds buffer re-establishment, which ultimately leads to a rebuffering event lasting for 1.4 seconds. Therefore, even if the buffer size marginally exceeds the outage duration at the onset of the outage, users may still encounter rebuffering events.

2.2 In-Depth Analysis of Network Outage

Numerous studies have confirmed that network outages in LSNs, such as Starlink, are primarily due to satellite handovers [17, 25, 31, 45]. It also has been demonstrated through both end-to-end measurements and signal analysis that Starlink schedules UE-satellite link reallocation every 15 seconds, specifically occurring at the $12th$, $27th$, $42nd$, and $57th$ seconds of each minute [3, 39]. However, the exact impact of satellite handovers is still unknown, as the success or failure of a handover depends on complex factors such as satellite load and the number of candidate satellites. To the best of our knowledge, the precise impact of these handovers is still under exploration, and no unified pattern has been identified. Building on this, we delve deeper into the distribution of outage events and utilize statistical models to formalize the frequency and duration of Starlink network outages.

Over three months, we recorded a total of 3,755 outage events. We consider each outage event as two separate statistical models: one for the occurrence of outages and another for their duration. We analyze the outage occurrence using a binomial distribution. The Cumulative Density Function (CDF) is illustrated in Figure 4. Our analysis indicates that there is an approximately 80% chance of experiencing an outage within a 60-minute interval. For the outage duration distribution, we evaluate various distributions commonly used to model event duration and compare their fitting accuracy based on the sum of squared errors (SSE). The Normal Inverse Gaussian (NIG) distribution demonstrates the lowest error, with an SSE of 2.84 and a standard deviation of 0.13. Figure 5 illustrates the probability distribution function of different outage durations, comparing the actual outage data (represented by bars) with the NIG distribution model (depicted as a line). The heavy-tail property of the NIG distribution also corresponds well with our observations: While 87.33% of outages last less than 2 seconds, 2.73% of outages can extend beyond 5 seconds, with durations reaching up to a maximum of 31 seconds. Although the majority of durations are less than 2 seconds, given the connection re-establishment delays discussed in Section 2.1, these short-duration outages can still trigger rebuffering events in live streaming services.
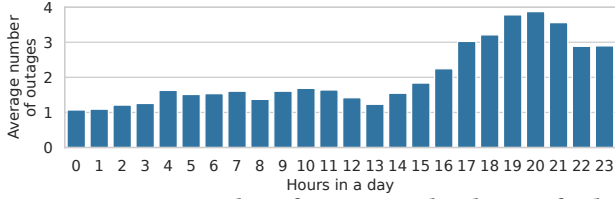
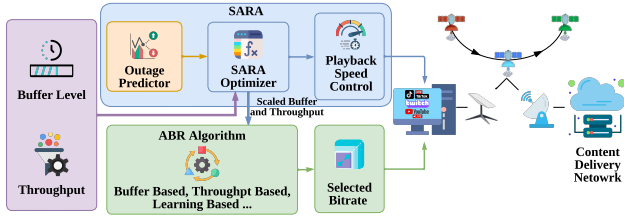Figure 6: Average number of outages within hours of a day.

Figure 7: Overview of SARA.

Additionally, we notice a clear correlation between the occurrence and duration of outages and specific times of day. As illustrated in Figure 6, the frequency of outages experiences a dramatic increase from 15:00 to 18:00, peaking at 20:00, which coincides with the Internet rush hour.[7] The duration of these outages varies significantly throughout the day, with the standard deviation reaching its highest at 8.03 around 00:00, then collapsing to 0.77 around 01:00. Given that each Starlink satellite has a finite number of antennas and shares communication channels among four users [31], the likelihood of handover failure escalates due to the increased frequency of handover operations required per satellite during these peak times. Intriguingly, our findings show that the evening Internet rush hour is more pronounced compared to the afternoon, suggesting that the majority of Starlink users are more likely to engage in entertainment activities, such as live streaming.

3   System Mechanism

3.1   SARA Overview

From our measurement and analysis, we develop SARA, a versatile middleware that can be easily integrated with a variety of existing ABR algorithms, whether they are buffer based, throughput based, or learning based. SARA's principal aim is to optimize user QoE by effectively minimizing rebuffering events caused by satellite handover and maintaining reasonably good video quality and smoothness. SARA achieves this by dynamically adjusting video playback speed and assisting ABR algorithms in selecting the ideal bitrate. The overall structure of an ABR system incorporating SARA is depicted in Figure 7. Upon the current playback status and outage event detected by the outage predictor, SARA optimizer returns three values: buffer scalar, throughput scalar, and playback speed. The scalars are forwarded to the ABR algorithm and act as factors influencing the final bitrate decision of the target ABR algorithm, leaving the core bitrate selection mechanism unchanged. This design enables target ABR to be aware of the influence of LSN outage events and requires minimal modifications to existing algorithms.
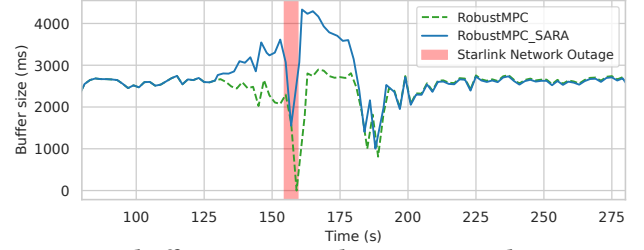
Figure 8: buffer occupation during a network outage.

Figure 8 illustrates an example of how SARA operates alongside other ABR algorithms. When there is an outage event which happens at time 160 with a duration of 2 seconds. In the case of RobustMPC, its throughput predictor failed to anticipate the upcoming network disruption. Consequently, the buffer is entirely depleted after the network outage. In comparison, when SARA is integrated with RobustMPC, it detects the impending outage event in advance and reacts by banking more video content both by slowing down the video playback speed and informing RobustMPC to lower the bitrate. Before the network outage that lasted for 2 seconds, it had already stored more than 3 seconds of video data, which allowed the system to withstand the network outage. Subsequently, SARA carefully increases the playback speed after the outage to reduce the LtB back to the desired level, thus maintaining a low-latency viewing experience.

3.2   Problem Formulation

In a typical adaptive video streaming system, video content is divided into K chunks with a fixed duration of α seconds. Each chunk is further encoded into multiple bitrate levels of different bitrates, denoted by $\mathcal{B} = \{b_1, b_2, \ldots, b_M\}$ where M represents the total number of bitrate levels. Let $b_{k,m}$ represent the bitrate selected for the k-th chunk, and $q(b_{k,m})$ represent the quality received by the user for the selected bitrate using any video quality metric (e.g., PSNR and VMAF [32]). Let ξ_k represent the average throughput, C_k represent the current buffer occupation in seconds, and LtB_k signify the current LtB in seconds. We denote the predicted outage event at chunk k as a 2-tuple $o_k = (o_k^t, o_k^d)$, where o_k^t is the time remaining until the outage occurs, and o_k^d is the duration of the outage.

For each decision round, our control variables consist of the buffer scalar s_k^b, the throughput scalar s_k^r, and the playback speed β_k. The buffer scalar and the throughput scalar range between 0 and 1 aimed to bias the ABR algorithms to select the appropriate bitrate. In addition, the range of β_k is constrained between 0.95 and 1.03 to ensure that changes in playback speed remain imperceptible to the viewers, as corroborated by prior research [7]. We denote $\psi_x(\cdot)$ as the generalized optimization target where the inputs are buffer occupancy C_k and/or throughput ξ_k, and the output is bitrate $b_{k,m}$. Thus, the selected bitrate can be represented as:

$$b_{k,m} = \begin{cases} \psi_b(s_k^b C_k), & \text{if } \psi_x(\cdot) \text{ is buffer based} \\ \psi_r(s_k^r \xi_k), & \text{if } \psi_x(\cdot) \text{ is throughput based} \\ \psi_h(s_k^b C_k, s_k^r \xi_k), & \text{if } \psi_x(\cdot) \text{ is hybrid or learning based} \end{cases}$$

Prior to the onset of outage o_k, we have a duration of o_k^t video content on the fly that can be downloaded. The maximum number of downloadable chunks prior to this outage is denoted as θ_k.

The actual θ_k is affected by our current throughput ξ_k and the selected bitrate $b_{k,m}$. Additionally, θ_k is constrained by the number of chunks produced by the broadcaster within the time period o_k^t. Thus, we can formulate the number of chunks that can be downloaded as:

$$\theta_k = \min \left\{ \left\lfloor \frac{\xi_k o_k^t}{b_k \alpha} \right\rfloor , \left\lfloor \frac{o_k^t}{\alpha} \right\rfloor \right\}, \forall k \in \mathcal{K}, \tag{1}$$

Given the operational constraint that a video chunk can only be decoded once the entire chunk is fully downloaded, it is necessary to round down the θ_k value to account for incomplete chunks. Taking into account the predicted outage events during each interval, we apply the following constraint to ensure that the buffer remains healthy after an outage:

$$\frac{1}{\beta_k}(C_k + \theta_k \alpha) - o_k^t - o_k^d \geq \gamma , \forall k \in \mathcal{K}, \tag{2}$$

The first term encapsulates the total available video content for playback before an outage o_k occurs. This is calculated by adding the already buffered video content to the video content that can be downloaded, adjusted by the playback speed. After subtracting o_k^t and o_k^d, we get the size of the remaining available buffer. This value must be equal to or exceed a predefined safety buffer level threshold γ to avoid playback interruptions. The necessity for a larger buffer becomes apparent in practical scenarios where outages can result in prolonged re-establishment times at the application level, as highlighted in Section 2.1. Drawing insight from our measurements, we configure γ to be 2 seconds.

If Eq. (2) cannot be satisfied for any possible values of β_k and $b_{k,m}$ and the buffer runs dry. This means SARA has insufficient time to prepare for the outage. In this case, the rebuffering duration can be defined as:

$$T_k = \max\{o_k^t + o_k^d + \gamma - \frac{1}{\beta_k}(C_k + \theta_k \alpha), 0\} \tag{3}$$

In light of previous work [1, 10, 11, 36, 42], we adopt the following linear-based QoE metric:

$$Q_k = q(b_{k,m}) - \omega T_k - \rho|b_k - b_{k-1}| - \eta|\beta_k - \beta_{k-1}| - \iota L_k, \forall k \in \mathcal{K}, \tag{4}$$

where $|b_k - b_{k-1}|$ represents the video bitrate smoothess and $|\beta_k - \beta_{k-1}|$ represents the playback speed smoothness. The L_k is live latency, which will be applied when LtB_k is greater than the target latency LtB_0. In our experiment, the LtB_0 is configured to 3 seconds, as per the Low-Latency live guideline [9]. The expression for L_k is as follows:

$$L_k = \max\{LtB_k - LtB_0, 0\}, \tag{5}$$

ρ, η, ω and ι are adjustment factors to trade-off the quality benefits and penalties. The objective of the system is to find the appropriate buffer scalar s_b^k, the throughput scalar s_r^k, and the playback speed β_k for each chunk k based on the above constraints, with the aim of maximizing total user QoE, represented as:

$$\max_{s_b^k, s_r^k, \beta_k} \sum_{k=1}^{K} Q_k \tag{6}$$
$$s.t. \quad (2) - (5)$$

Algorithm 1 Video Adaption Process using SARA

1: **for** k = 1 to $\mathcal{K}$ **do**
2:      $C_k \leftarrow$ current buffer state
3:      $\xi_k \leftarrow$ throughput measured when downloading chunk $k-1$
4:      $(o_k^t, o_k^d) \leftarrow Outage\_Predictor()$
5:      $(s_k^b, s_b^r, \beta) \leftarrow SARA\_Optimizer(o_k^t, o_k^d)$
6:      **if** $s_k^b < 1$ or $s_b^r < 1$ **then**
7:          $C_k \leftarrow s_k^b \cdot C_k$
8:          $\xi_k \leftarrow s_k^r \cdot \xi_k$
9:      **end if**
10:      set playback speed to β
11:      $b_{k,m} \leftarrow ABR\_Algorithm(C_k, \xi_k)$
12:      download chunk k with bitrate m, wait until finished
13: **end for**

Algorithm 2 SARA Optimizer

1: $\mathcal{N} \leftarrow$ number of iterations
2: $\mathcal{R} \leftarrow$ number of particles
3: $\mathcal{A} \leftarrow$ aggressiveness
4: $w_1, w_2, w_3 \leftarrow$ initialize weights
5: **if** $C_k < o_k^t$ **then**
6:      $O \leftarrow max\{(C_k - o_k^t)/C_k, -0.2\}$
7: **end if**
8: **for** i = 1 to $\mathcal{R}$ **do**
9:      $(s_{k,i}^b, s_{k,i}^r, \beta_{k,i}) \leftarrow$ randomly generate within bounds
10:      $v_i^{s_k^b}, v_i^{s_k^r}, v_i^{\beta_k} \leftarrow$ randomly generate between 0 and $\mathcal{A}$
11:      $(p_i^{s_k^b}, p_i^{s_k^r}, p_i^{\beta_k}) \leftarrow (s_{k,i}^b, s_{k,i}^r, \beta_{k,i})$
12:      $QoE_i = QoE(s_{k,i}^b, s_{k,i}^r, \beta_{k,i}) \leftarrow$ calculate using Equation 6
13: **end for**
14: $G^{s_k^b}, G^{s_k^r}, G^{\beta_k} \leftarrow$ arg max$\{QoE_i, \ldots, QoE_n\}$
15: **for** n = 1 to $\mathcal{N}$ **do**
16:      **for** i = 1 to $\mathcal{R}$ **do**
17:          $r_1, r_2 \leftarrow$ randomly generate between 0 and 1
18:          $v_i^{s_k^b} \leftarrow w_1 \cdot v_i^{s_k^b} + w_2 \cdot r_1 \cdot (p_i^{s_k^b} - s_k^{b_i}) + w_3 \cdot r_2 \cdot (G^{s_k^b} - s_k^{b_i})$
19:          $s_{k,i}^b \leftarrow clip\{s_{k,i}^b + v_i^{s_k^b} + O, 0, 1\}$
20:          repeat steps 17-19 for $s_{k,i}^r$ and $\beta_{k,i}$ where the clip range is (0.95, 1.03) for $\beta_{k,i}$
21:          **if** $QoE(s_{k,i}^b, s_{k,i}^r, \beta_{k,i}) > QoE(p_i^{s_k^b}, p_i^{s_k^r}, p_i^{\beta_k})$ **then**
22:              $(p_i^{s_k^b}, p_i^{s_k^r}, p_i^{\beta_k}) \leftarrow (s_{k,i}^b, s_{k,i}^r, \beta_{k,i})$
23:          **end if**
24:      **end for**
25:      $G^{s_k^b}, G^{s_k^r}, G^{\beta_k} \leftarrow$ arg max$\{QoE_i, \ldots, QoE_n\}$
26: **end for**
27: **return** $G^{s_k^b}, G^{s_k^r}, G^{\beta_k}$

Algorithm 1 delineates the operational methodology of SARA when it is integrated with any ABR algorithm. The process begins with SARA conducting a preliminary assessment via the Outage_Predictor module to predict upcoming network outages. Following this prediction, SARA utilizes the outage forecasts to adjust the values of several key parameters: the buffer scalar, throughput scalar, and playback speed. These scalars are strategically applied to

the current state of the buffer and the estimated throughput, thereby modulating the inputs to the ABR algorithm. This modulation is designed to decisively influence the ABR's bitrate selection. In scenarios where no outage is anticipated and the LtB metric is within optimal ranges, these scalars default to a value of 1. This means that the ABR algorithm operates without external adjustments, maintaining full autonomy over bitrate decisions.

Since both buffer and throughput scalars, as well as playback speed, are continuous variables, identifying the optimal solution can be computationally intensive, rendering it impractical for live streaming contexts. To address this, we have employed a heuristics-based strategy using Particle Swarm Optimization (PSO) [4], as detailed in Algorithm 2. In this algorithm, we spawn $\mathcal{R}$ particles, each representing a potential state vector $(s_{k,i}^b, s_{k,i}^r, \beta_{k,i})$ with random initial values. Each particle is assigned a velocity vector $(v_i^{s_k^b}, v_i^{s_k^r}, v_i^{\beta_k})$ between 0 and $\mathcal{A}$, which dictates the magnitude of exploration within the solution space. The exploration aggressive, denoted by $\mathcal{A}$, determines how aggressively the particles explore the solution space. Furthermore, every particle maintains a record of its optimal state encountered thus far, represented as $(p_i^{s_k^b}, p_i^{s_k^r}, p_i^{\beta_k})$. Additionally, a global vector denoted as $G^{s_k^b}, G^{s_k^r}, G^{\beta_k}$ is kept to track the state vector with the highest QoE observed across all particles so far. During the optimization phase, we update the velocity of each particle based on its relative displacement from both its local best state and the global best state. This update is moderated by weight factors w_1, w_w and w_3 to balance exploration and exploitation dynamics and is randomized through coefficients r_1 and r_2 (uniformly distributed between 0 and 1) to introduce variability in the search process. To further improve the algorithm's performance, we introduce a variable O, which assesses the buffer health relative to upcoming predicted network outages. A higher O value encourages exploration towards lower state values, facilitating faster convergence to more advantageous states. This exploratory process is iterated $\mathcal{N}$ times to ascertain the final configurations for the buffer scalar, throughput scalar, and playback speed.

4 Evaluation

4.1 Experimental Setup

Considering the dynamics of LSNs and the fact that the exact time and duration of network outages caused by satellite handover are not accessible ahead of time, we conduct evaluations in a simulated environment as widely used in [10, 11, 15, 26, 35] to ensure a fair assessment of SARA in a live streaming environment under LSNs. The simulated network conditions, such as ping and bandwidth, were based on the real-world data we gathered from our measurements on Starlink. The network outages sampled from Section 2.2 will be embedded into the network trace to emulate the realistic Starlink network outages. The video trace utilized in our experiment featured a ten-minute movie from Big Buck Bunny.[8] The movie was processed using the same standard definition encoding as [36], with four bitrates $\mathcal{B} = \{1000, 2500, 5000, 8000\}$ Kbps and a segment duration of 500ms. For the QoE metric in Eq. (4), we adopt two widely used settings as in [14, 26, 36, 42]. The linear quality metric QoE_{lin}
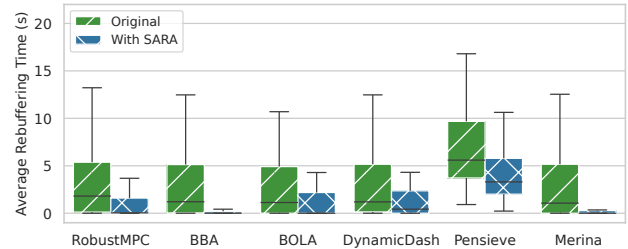

Figure 9: Average rebuffering time of different ABR algorithms and their integration with SARA.

with $q(b_k) = b_k/1000$, $\omega = 4.33$, $\rho = 1$, and the log-form quality metric QoE_{log} with $q(b_k) = log(b_k/min(\mathcal{B}))$, $\omega = 2.66$, $\rho = 1$. η and ι are set to $min(\mathcal{B})$ and 1 respectively inspired by [1].

4.2 Baseline Algorithms

To evaluate the versatility of SARA, we selected a variety of well-known ABR algorithms. These algorithms represent a broad cross-section of different approaches to adaptive bitrate control, and their diverse strategies and performance characteristics make them an ideal testing ground for SARA's performance and compatibility. We have further modified these ABR algorithms to better align with live streaming scenarios. The chosen ABRs are as follows:

- RobustMPC [42]: Utilizes the harmonic mean of past throughput to predict future network throughput.
- Pensieve [26]: Employs reinforcement learning, allowing for adaptation to various environments and QoE metrics.
- Buffer Occupancy based Lyapunov Algorithm (BOLA) [36]: Utilizes Lyapunov optimization to minimize rebuffering and maximize video quality.
- BBA [12]: Decides bitrates based on current buffer capacity and reservoir estimation.
- DynamicDash [33]: The default ABR algorithm used by the current standard DASH reference player (4.7.3) which switches between throughput based and buffer based ABR.[9]
- Merina [14]: Utilizes meta reinforcement learning to rapidly adjust its control policy in response to the dynamic changes in network throughput.

4.3 SARA Performance Evaluation

We integrate SARA into six previously mentioned ABR algorithms and evaluate their performance. The results, illustrated in Figure 9, highlight SARA's impact in terms of rebuffering times. On average, rebuffering time is reduced by 39.41%, with the BBA achieving an average reduction of 52.26%. SARA also reduced overall rebuffering events by an average of 21.41%. It is worth noting that a significant standard deviation is observed in average rebuffering times. This variability is attributed to the broad range of outage durations in the Starlink network, as discussed in Section 2.2. With network outages lasting anywhere from 0.2 seconds to 23.57 seconds, such a wide spectrum of durations inevitably results in significant fluctuations in average rebuffering times. Moreover, our network traces include a few instances without any network outages, further adding to the observed variance. Figure 10 further shows the impact of SARA on three other metrics across the selected ABR algorithms and their
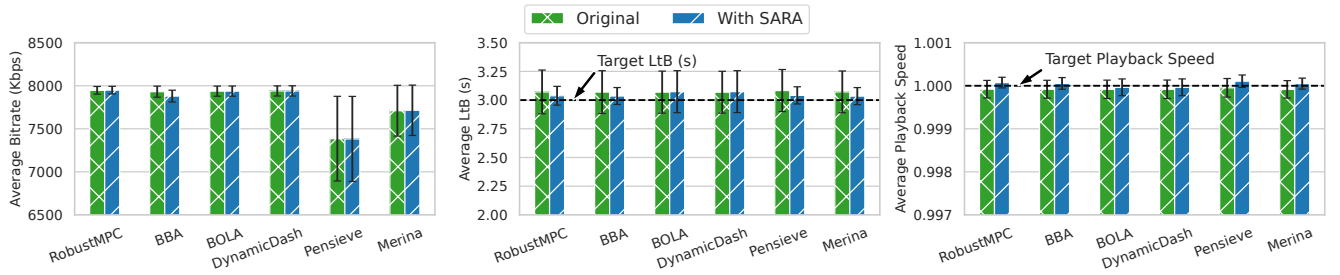
---

[8]https://peach.blender.org/

[9]https://reference.dashif.org/dash.js/

Figure 10: QoE metrics of different ABR algorithms and their integration with SARA.
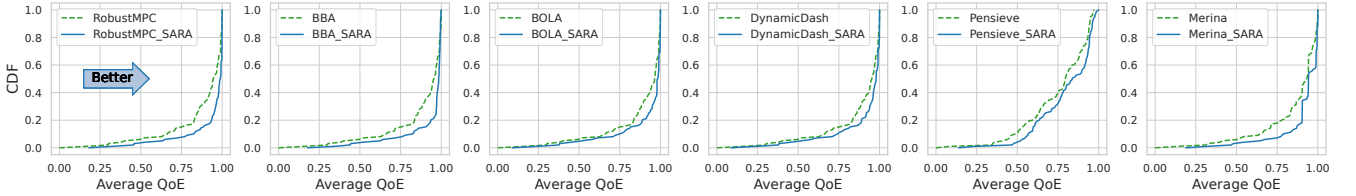

Figure 11: CDF of QoE_{lin} metrics of different ABR algorithms and their integration with SARA.
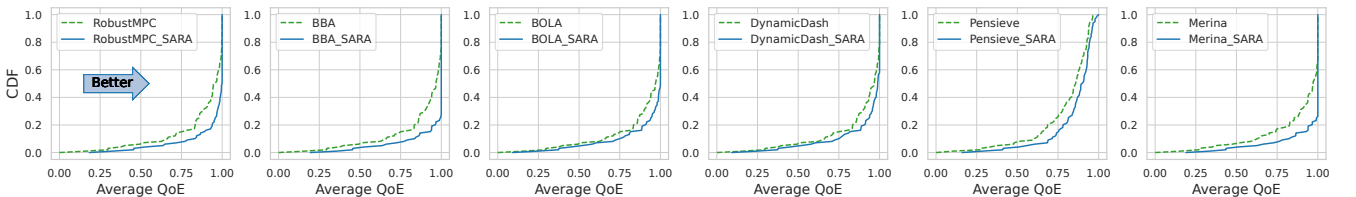

Figure 12: CDF of QoE_{log} metrics of different ABR algorithms and their integration with SARA.

SARA-enhanced versions. SARA results in a slight improvement in LtB of 0.65% and only incurs a negligible reduction in bitrate by 0.13%. Although it leads to a 23.36% increase in the amount of time where the playback speed differs from 1x, compared to the original ABR algorithms, SARA only alters the playback speed within a narrow range from 0.95x to 1.03x, which is typically imperceptible to viewers in practice [1].

We also notice that Pensieve performs worse than other ABR algorithms. Further analysis reveals that Pensieve tends to respond slowly to outage events. It often reduces the bitrates with a delay, by which time the bandwidth may have already recovered, resulting in reduced bandwidth efficiency and an increase in rebuffering time. Although Pensieve does not perform optimally under LSN conditions, the integration of SARA still improves its performance, resulting in an average of 33% less rebuffering time.

Figure 11 and Figure 12 present the performance evaluation in the form of CDFs for each ABR algorithm and their integration with SARA among two QoE metrics. The results show that SARA either matches or surpasses the performance of the base ABR algorithm that it is incorporated into. For example, for Merina, 75% of QoE_{log} values are above 0.8, while when SARA is integrated with it, 80% of QoE values exceed 0.97. In essence, SARA's ability to predict outages, dynamically adjust playback speed, and assist ABRs in selecting the ideal bitrate forms an efficient strategy that significantly enhances these ABR algorithms' performance, thereby improving the live streaming experiences in LSN environments.

5 Further Discussion: Outage Predictor

A key component of SARA is the "Outage Predictor " which can forecast upcoming network outages from satellite handovers. However,

predicting network outages presents a significant challenge due to the black box design of satellite operators' system architecture, such as Starlink [20, 25]. Essential details such as the strategies for satellite handovers, network routing, and satellite system load remain proprietary and are therefore not disclosed to the general public [31, 39, 45]. Fortunately, the orbital positions of Starlink satellites are accessible publicly in the Two-Line Element (TLE) format from various sources. This accessibility enables the calculation of the satellites' positions relative to the UE at any moment. Moreover, the orbital paths of the satellites exhibit consistent, timed characteristics, recurring at fixed intervals. This regularity positions the problem well for analysis using time-sequenced modelling techniques. Furthermore, recent advancements in transformers offer promising solutions for analyzing and predicting time-sequenced data [40]. Therefore, one possible further enhancement on SARA is to utilize *Informer*, a state-of-the-art transformer-based approach designed for efficient long-sequence time-series forecasting in the design of the outage predictor [46]. In this section, we further discuss our initial efforts in this direction.

In our informer based design, the training data incorporates the measurement data outlined in Section 2 and TLEs from CelesTrak.[10] Since we do not know which satellite Starlink will switch to after each handover period, all satellites that have an angle of elevation higher than $25°$ [34], along with their distance to the UE, are included in the training data. To further improve the accuracy of the outage predictor, we also integrate weather data sourced from the *Weather API* provided by OpenWeather [29], acknowledging the impact of weather conditions on the performance of the Starlink network, as highlighted in previous studies [25, 28, 45].

---

[10]https://celestrak.org/NORAD/Elements/table.php?GROUP=starlink

Table 1: Performance of the outage predictor with various configurations

Input Length	Prediction Length	MSE	MAE
	30	0.519	0.107
300	60	0.521	0.113
	120	0.542	0.121
	30	0.517	0.107
600	60	0.523	0.116
	120	0.563	0.119
	30	0.524	0.109
900	60	0.567	0.120
	120	0.619	0.129

Figure 13: Average rebuffering time of different ABR algorithms and their integration with SARA using a transformer based outage predictor.

We train the Informer network under a range of configurations, varying both the input length and the prediction length. For instance, an input length of 300 and a prediction length of 30 means the model will analyze the latest 5 minutes of data to forecast network outages in the upcoming 30 seconds. An extended prediction length equips SARA with a longer preparation window ahead of network outages, potentially enhancing its resilience to prolonged disruptions. On the other hand, augmenting the input length tends to improve performance due to the richer contextual data provided to the model, albeit at the cost of increased inference times. The impact of these variations on the model's accuracy is summarized in Table 1, which presents the results in terms of Mean Squared Error (MSE) and Maximum Absolute Error (MAE).

After fine-tuning, the optimal configuration for the model was determined to be an input length of 600 seconds and a prediction length of 120 seconds. The outage predictor achieved an overall accuracy and recall rate of 79.43% and 38.23%, respectively. In our configuration, the model operates every 5 seconds and has an average inference time of approximately 128.96 ms using an A5000 GPU. It is important to note that this model was developed with a constrained dataset. In practice, satellite operators such as SpaceX and Amazon have access to more comprehensive data, enabling the development of predictors with even superior performance metrics. Nevertheless, our model still offers a versatile solution that is simultaneously applicable across various satellite operators, including SpaceX Starlink, Amazon Kuiper and future LSN operators. Despite the limited data, our model still effectively reduces rebuffering events, as illustrated in Figure 13.

6 Related Works

The domain of LSNs has been extensively explored in recent research, with a significant focus on the Starlink network, emphasizing comprehensive end-to-end network performance evaluations. These studies have delved into environmental influences on network performance, such as terrain and weather conditions [17, 25, 28, 45], affirming Starlink's applicability across a spectrum of applications while underscoring the challenges inherent to LSNs, notably the frequency of satellite handovers. Further investigations into satellite handovers within LSNs have unveiled the deployment of a global scheduler by Starlink, which executes handover decisions at 15-second intervals [31, 39]. Concurrently, advancements in simulation tools have deepened our understanding of LSN operational conditions [18, 19], enhancing the fidelity of network
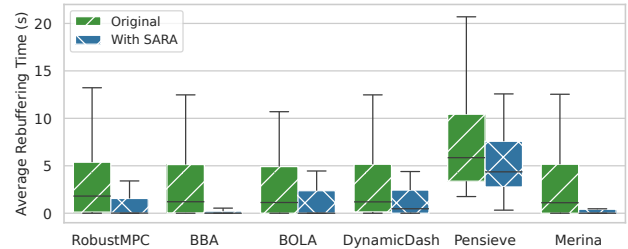
models. Efforts to improve lower-level TCP protocols have also been pursued, aiming to enhance overall LSN performance [5].

In the context of video streaming services, significant advancements have been made in the development of ABR algorithms that adeptly respond to fluctuating network conditions. These include buffer based approaches [12, 36], throughput based approaches [13, 38] and a hybrid of both [33, 42], which are simple yet effective. Additionally, innovative learning-based ABR algorithms, notably Pensieve and Comyco, are capable of directly optimizing QoE without the necessity for iterative computation [8, 14, 15, 20, 26, 41, 43]. ABR algorithms such as LoL and LoL+ have been tailored specifically for low latency in live streaming platforms [1, 6, 16, 21, 27], integrating bitrate adaptation with heuristic based playback speed control. In parallel, various studies have focused on minimizing hardware load and enhancing power efficiency, marking significant strides towards more sustainable streaming technology [23, 47].

7 Conclusion and Future Works

In this paper, we examined the challenges of live streaming over LSNs, notably the impact of frequent satellite handovers on live streaming services. Our investigation revealed that existing ABR algorithms, which perform well in terrestrial network environments, struggle to maintain consistent streaming quality in the dynamic conditions characteristic of LSNs. We introduced SARA, a versatile and lightweight middleware solution designed to enhance the adaptability of existing ABR algorithms in LSN contexts by feeding ABR algorithms with specific network characteristics unique to LSNs. Our evaluation shows that SARA significantly reduced rebuffering times by approximately 39.41% while only incurring a neglectable loss in an average bitrate of 0.65%, demonstrating its efficacy in optimizing live streaming services in satellite networks.

LSN services are rapidly evolving and continue to present many fascinating and challenging areas in need of exploration. An area of particular interest is the realm of realtime communication and asynchronous interaction, which has not yet been extensively investigated. Our future work will delve into these uncharted territories, seeking to optimize the performance of LSNs in facilitating seamless, realtime interactions.

References

[1] Abdelhak Bentaleb, Mehmet N Akcay, May Lim, Ali C Begen, and Roger Zimmermann. 2021. Catching the moment With LoL+ in Twitch-Like low-latency live streaming platforms. *IEEE Transactions on Multimedia* 24 (2021), 2300–2314.

[2] Debopam Bhattacherjee and Ankit Singla. 2019. Network topology design at 27,000 km/hour. In *Proc. ACM CoNEXT*.

[3] Rodrigo Blázquez-García, Diego Cristallini, Martin Ummenhofer, Viktor Seidel, Jörg Heckenbach, and Daniel O'Hagan. 2023. Experimental comparison of Starlink and OneWeb signals for passive radar. In *Proc. IEEE RadarConf*.

[4] Mohammad Reza Bonyadi and Zbigniew Michalewicz. 2017. Particle swarm optimization for single objective continuous space problems: a review. *Evolutionary computation* 25, 1 (2017), 1–54.

[5] Xuyang Cao and Xinyu Zhang. 2023. SaTCP: Link-layer informed TCP adaptation for highly dynamic LEO satellite networks. In *Proc. IEEE INFOCOM*.

[6] Ying Chen, Qing Li, Aoyang Zhang, Longhao Zou, Yong Jiang, Zhimin Xu, Junlin Li, and Zhenhui Yuan. 2021. Higher quality live streaming under lower uplink bandwidth: an approach of super-resolution based video coding. In *Proc. ACM NOSSDAV*.

[7] Claudio de'Sperati and Gabriel Baud Bovy. 2017. Low perceptual sensitivity to altered video speed in viewing a soccer match. *Nature Publishing Group Scientific reports* 7, 1 (2017), 15379.

[8] Jianchao He, Miao Hu, Yipeng Zhou, and Di Wu. 2020. LiveClip: towards intelligent mobile short-form video streaming with deep reinforcement learning. In *Proc. ACM NOSSDAV*.

[9] Jake Holland, Ali C. Begen, and Spencer Dawkins. 2022. Operational Considerations for Streaming Media. RFC 9317. https://doi.org/10.17487/RFC9317

[10] Tianchi Huang, Rui-Xiao Zhang, Chenglei Wu, and Lifeng Sun. 2023. Optimizing Adaptive Video Streaming with Human Feedback. In *Proc. ACM MM*. 1707–1718.

[11] Tianchi Huang, Chao Zhou, Rui-Xiao Zhang, Chenglei Wu, Xin Yao, and Lifeng Sun. 2019. Comyco: Quality-aware adaptive video streaming via imitation learning. In *Proc. ACM MM*. 429–437.

[12] 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. In *Proc. ACM SIGCOMM*.

[13] Junchen Jiang, Vyas Sekar, and Hui Zhang. 2012. Improving fairness, efficiency, and stability in http-based adaptive video streaming with festive. In *Proc. ACM SIGCOMM*.

[14] Nuowen Kan, Yuankun Jiang, Chenglin Li, Wenrui Dai, Junni Zou, and Hongkai Xiong. 2022. Improving Generalization for Neural Adaptive Video Streaming via Meta Reinforcement Learning. In *Proc. ACM MM*.

[15] Nuowen Kan, Chenglin Li, Caiyi Yang, Wenrui Dai, Junni Zou, and Hongkai Xiong. 2021. Uncertainty-aware robust adaptive video streaming with bayesian neural network and model predictive control. In *Proc. ACM NOSSDAV*.

[16] Theo Karagkioules, Rufael Mekuria, Dirk Griffioen, and Arjen Wagenaar. 2020. Online learning for low-latency adaptive streaming. In *Proc. ACM MMSys*.

[17] Mohamed M Kassem, Aravindh Raman, Diego Perino, and Nishanth Sastry. 2022. A browser-side view of Starlink connectivity. In *Proc. ACM IMC*.

[18] Simon Kassing, Debopam Bhattacherjee, André Baptista Águas, Jens Eirik Saethre, and Ankit Singla. 2020. Exploring the" Internet from space" with Hypatia. In *Proc. ACM IMC*.

[19] Zeqi Lai, Hewu Li, and Jihao Li. 2020. Starperf: Characterizing network performance for emerging mega-constellations. In *IEEE ICNP*.

[20] Yueheng Li, Qianyuan Zheng, Zicheng Zhang, Hao Chen, and Zhan Ma. 2023. Improving ABR performance for short video streaming using multi-agent reinforcement learning with expert guidance. In *Proc. ACM NOSSDAV*.

[21] May Lim, Mehmet N Akcay, Abdelhak Bentaleb, Ali C Begen, and Roger Zimmermann. 2020. When they go high, we go low: low-latency live streaming in dash.js with LoL. In *Proc. ACM MMSys*.

[22] Xingqin Lin, Stefano Cioni, Gilles Charbit, Nicolas Chuberre, Sven Hellsten, and Jean-Francois Boutillon. 2021. On the path to 6G: Embracing the next wave of low Earth orbit satellite access. *IEEE Communications Magazine* 59, 12 (2021), 36–42.

[23] Tomasz Lyko, Matthew Broadbent, Nicholas Race, Mike Nilsson, Paul Farrow, and Steve Appleby. 2020. Evaluation of CMAF in live streaming scenarios. In *Proc. ACM NOSSDAV*.

[24] Sami Ma, Yi Ching Chou, Miao Zhang, Hao Fang, Haoyuan Zhao, Jiangchuan Liu, and William I Atlas. 2024. LEO Satellite Network Access in the Wild: Potentials, Experiences, and Challenges. *IEEE Network* (2024).

[25] Sami Ma, Yi Ching Chou, Haoyuan Zhao, Long Chen, Xiaoqiang Ma, and Jiangchuan Liu. 2023. Network characteristics of LEO satellite constellations: A Starlink-based measurement from end users. In *Proc. IEEE INFOCOM*.

[26] Hongzi Mao, Ravi Netravali, and Mohammad Alizadeh. 2017. Neural adaptive video streaming with pensieve. In *Proc. ACM SIGCOMM*.

[27] Ying Mao, Jiayin Wang, and Bo Sheng. 2014. DAB: Dynamic and agile buffer-control for streaming videos on mobile devices. *Elsevier Procedia Computer Science* 34 (2014), 384–391.

[28] François Michel, Martino Trevisan, Danilo Giordano, and Olivier Bonaventure. 2022. A first look at Starlink performance. In *Proc. ACM IMC*.

[29] OpenWeather. 2024. Current weather data. Retrieved Aug 7th, 2024 from https://openweathermap.org/current.

[30] Nils Pachler, Inigo del Portillo, Edward F Crawley, and Bruce G Cameron. 2021. An updated comparison of four low earth orbit satellite constellation systems to provide global broadband. In *Proc. IEEE ICC*.

[31] Jianping Pan, Jinwei Zhao, and Lin Cai. 2023. Measuring a low-earth-orbit satellite network. In *Proc. IEEE PIMRC*.

[32] Reza Rassool. 2017. VMAF reproducibility: Validating a perceptual practical video quality metric. In *Proc. IEEE BMSB*. IEEE, 1–2.

[33] Iraj Sodagar. 2011. The mpeg-dash standard for multimedia streaming over the internet. *IEEE Multimedia* 18, 4 (2011), 62–67.

[34] SpaceX. 2024. Request for Orbital Deployment and Operating Authority for the SpaceX Gen2 NGSO Satellite System. Retrieved Aug 7th, 2024 from https://docs.fcc.gov/public/attachments/DA-24-222A1.pdf.

[35] Kevin Spiteri, Ramesh Sitaraman, and Daniel Sparacio. 2019. From theory to practice: Improving bitrate adaptation in the DASH reference player. *ACM Transactions on Multimedia Computing, Communications, and Applications* 15, 2s (2019), 1–29.

[36] Kevin Spiteri, Rahul Urgaonkar, and Ramesh K Sitaraman. 2020. BOLA: Near-optimal bitrate adaptation for online videos. *IEEE/ACM Transactions on Networking* 28, 4 (2020), 1698–1711.

[37] statista. 2023. Leading gaming and non-gaming live streaming platforms worldwide in September 2023, by hours watched. Retrieved Aug 7th, 2024 from https://www.statista.com/statistics/1409393/top-live-streaming-platforms-hours-watched/.

[38] Yi Sun, Xiaoqi Yin, Junchen Jiang, Vyas Sekar, Fuyuan Lin, Nanshu Wang, Tao Liu, and Bruno Sinopoli. 2016. CS2P: Improving video bitrate selection and adaptation with data-driven throughput prediction. In *Proc. ACM SIGCOMM*. 272–285.

[39] Hammas Bin Tanveer, Mike Puchol, Rachee Singh, Antonio Bianchi, and Rishab Nithyanand. 2023. Making Sense of Constellations: Methodologies for Understanding Starlink's Scheduling Algorithms. In *Proc. ACM CoNEXT*.

[40] Qingsong Wen, Tian Zhou, Chaoli Zhang, Weiqi Chen, Ziqing Ma, Junchi Yan, and Liang Sun. 2022. Transformers in time series: A survey. In *Proc. IJCAI*. 6778–6786.

[41] Francis Y Yan, Hudson Ayers, Chenzhi Zhu, Sadjad Fouladi, James Hong, Keyi Zhang, Philip Levis, and Keith Winstein. 2020. Learning in situ: a randomized experiment in video streaming. In *USENIX NSDI*. 495–511.

[42] Xiaoqi Yin, Abhishek Jindal, Vyas Sekar, and Bruno Sinopoli. 2015. A control-theoretic approach for dynamic adaptive video streaming over HTTP. In *Proc. ACM SIGCOMM*.

[43] Danfu Yuan, Yuanhong Zhang, Weizhan Zhang, Xuncheng Liu, Haipeng Du, and Qinghua Zheng. 2022. PRIOR: deep reinforced adaptive video streaming with attention-based throughput prediction. In *Proc. ACM NOSSDAV*.

[44] Miao Zhang, Linfeng Shen, Xiaoqiang Ma, and Jiangchuan Liu. 2023. Toward 6G-enabled mobile vision analytics for immersive extended reality. *IEEE Wireless Communications* 30, 3 (2023), 132–138.

[45] Haoyuan Zhao, Hao Fang, Feng Wang, and Jiangchuan Liu. 2023. Realtime multimedia services over Starlink: A reality check. In *Proc. ACM NOSSDAV*.

[46] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. 2021. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proc. AAAI*. 11106–11115.

[47] Ziehen Zhu, Xianglong Feng, Zhongze Tang, Nan Jiang, Tian Guo, Lisong Xu, and Sheng Wei. 2022. Power-efficient live virtual reality streaming using edge offloading. In *Proc. ACM NOSSDAV*.