How to deliver videos faster and better?

Lilac Lai, Carol Song

1 INTRODUCTION

Video delivery has gained significant prominence in today's digital era, given its widespread application across various sectors. Video delivery is vital for entertainment on platforms such as TikTok, YouTube, and Netflix, as well as for facilitating daily work operations via tools like Zoom. This survey paper aims to explore various approaches for enhancing the speed, quality, and user experience of video transport in networks. Recent video delivery research is largely done on improving video transport methods and client-side video streaming techniques such as Adaptive Bitrate Streaming (ABR).

One of the latest and most promising ABR methods is Fugu [2], developed by Francis Y. from Stanford, which has been shown to improve the quality of user experience (QoE). In addition to optimizing the QoE equation, Xu Zhang and Yiyang Ou propose a smart approach to dynamically rebuffer videos based on sensitivity [3].

Other researchers have offered a combination of transport layer and application layer optimizations. In the XLINK paper [4], the authors used QoE information extracted in the application layer to aid the multi-path transport of video segments. In the VOXEL paper [1], the authors decided the importance of each video frame based on a video-frame referencing graph to determine which frames to prioritize sending in combination with a specially designed ABR algorithm that also focuses on QoE.

2 VIDEO STREAMING

2.1 Adaptive Bitrate Streaming

Several aspects of research aim to achieve effective streaming video transmission and increase the QoE, one of them is the Adaptive Bitrate Streaming (ABR) technique. It is widely used in the application layer for video transmission. ABR can adjust the quality of video content with respect to available network bandwidth and guarantees a smooth viewing experience for users. It accomplishes this goal by encoding a small chunk of video in multiple versions with different bitrates and resolutions in the server. Then the server will then send the appropriate video stream to the user. It typically starts with a low-quality stream while start playing and then monitoring the bandwidth of the user and quickly making adjustments to the optimum value. Overall, the main goal in ABR scheme is to choose the chunk that can maximize both the utilization of network resources and QoE with the uncertainty of future throughput.

In the paper 'Learning in situ: a randomized experiment in video streaming[2]', the authors have two main contributions. The first one is they design Puffer, a live study platform of ABR. The second one is they develop a new ABR scheme, Fugu, that balanced both traditional buffer control and a situ supervised learning method. This method is used to provide a probabilistic predictor of the upcoming chunk transmission rate based on real-world networking data. The authors launched an experiment of ABR on Puffer by recording the spent time for users, SSIM and size for message chunk, etc. They used this time series data to measure the effectiveness of Fugu by comparing it to different ABR algorithms.

There is currently a lack of a universal standard to measure the effectiveness of ABR schemes. The authors thus design Puffer, a live-stream TV platform, allowing researchers to collect data of participants' behavior and network paths to measure the potential of an ABR scheme. They use libx264 to encode each video in ten different H.264 versions and use ffmpeg to calculate the SSIM of each video for measuring quality. The server for Puffer is 48-core with 10 Gbps Ethernet in a datacenter at Stanford. They have advertised their platform on Google and Reddit with keywords and there is 111,231 unique user IP with 1,904,316 streams from Jan. 26, 2019 to Feb. 2, 2020 in this system. The author has observed a heavy-tailed behavior and a large percentage of uncertainty in the user statistic they gathered.

In the algorithm designing stage, based on their preliminary findings, the authors proposed Fugu which integrates both situ machine learning method and classical MPC (model predictive control). In FuGu, the QoE of each chunk is a linear combination of video quality and stall time.

Let $Q(K)$ be the video quality of a chunk K, $T(K)$ be the uncertain transmission time of K, and B_i be the current playback buffer size. λ and μ are two weight configuration constants. Given the previously sent chunk K_{i-1}, Fugu defines the QoE obtained by sending K_i^s as

$$\begin{aligned} QoE\left(K_i^s, K_{i-1}\right) =& Q\left(K_i^s\right) - \lambda \left|Q\left(K_i^s\right) - Q\left(K_{i-1}\right)\right| \\ & - \mu \cdot \max\left\{T\left(K_i^s\right) - B_i, 0\right\}, \end{aligned}$$

where $\max\left\{T\left(K_i^s\right) - B_i, 0\right\}$ is the stall time experienced by sending K_i^s. The first two-term in this function would be easy to obtain after selecting the chunk. The third term related to stall time would be hard to evaluate straight forward but will be the outcome of a neural-network transmission-time predictor with respect to transportation time and size for previous chunks, TCP statistics(congestion window size, number of unacknowledged packets, RTT, etc.), and the size

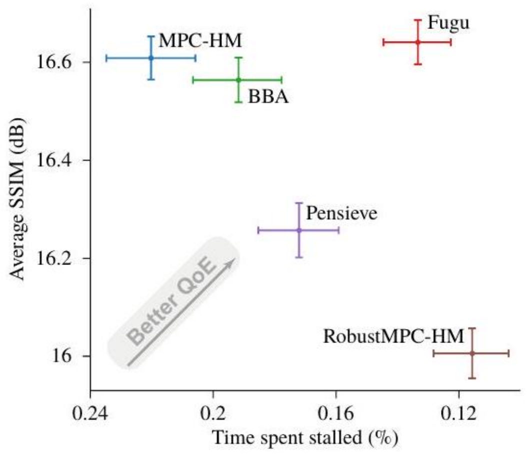


Figure 1: Primary Experiment Error bars show 95% confidence intervals.[2]

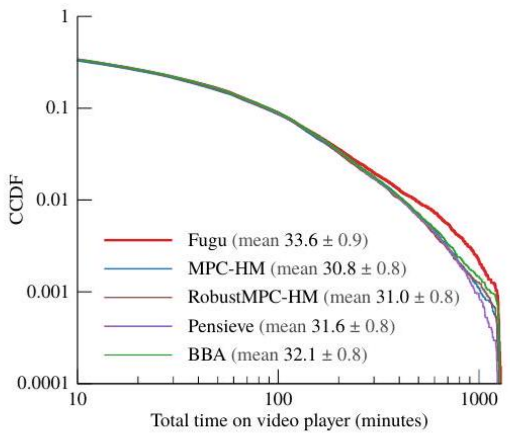


Figure 2: Users randomly assigned to Fugu chose to remain on the Puffer video player about 5%–9% longer, on average, than those assigned to other schemes.[2]

of the future proposed chunk. The result is a distribution of probability for each trunk and will be evaluated further for choosing the chunk.

Their MPC maximizes the expected cumulative QoE in the previous Equation with replanning. It queries TTP for predictions of transmission time and after sending K_{i-1}^s, they update the current function and re-plan for K_i^s and afterward. The value iteration for calculating the maximum expected sum of QoE that can be achieved denoted by $v_i^*\left(B_i, K_{i-1}\right)$, is based on the current playback buffer level B_i and the last sent chunk K_{i-1}. It can be expressed as follows:

$$v_i^*\left(B_i, K_{i-1}\right) = \max_{K_i^s}\Big\{ \sum_{t_i} \Pr\left[\hat{T}\left(K_i^s\right) = t_i\right] \cdot \left(QoE\left(K_i^s, K_{i-1}\right) + v_{i+1}^*\left(B_{i+1}, K_i^s\right)\right)\Big\},$$

As a result, they use $t = 8$ for TTP; $H = 5$, $\lambda = 1$, and $\mu = 100$ for $QoE\left(K_i^s, K_{i-1}\right)$ as the tuned parameter for next-step evaluation.

While it is commonly believed in the machine learning field that complex models outperform simpler ones, the authors of this paper contend that in the realm of computer networking, the generic buffer-based control method usually has a great performance and is hard to defeat by sophisticated algorithms. The authors conducted an experiment that compared Fugu with four other commonly used ABR algorithms. According to Figure 1, Fugu reduced the fraction of time spent stalled and was the only advanced method that consistently outperformed the baseline. Figure 2 suggests that users of Fugu spent significantly more time streaming than users of other methods.

The author has two main concerns, which are related to the user base and the lack of transparency in the study. The lack of dynamism in the Puffer userbase leads to zero benefit from daily retraining and the accuracy for TTP may stale at some point. Additionally, because the server is located only at Stanford, it is unclear whether Fugu would perform as well in other locations. Furthermore, the randomized controlled trial is considered a "black box" study of ABR algorithms for video streaming, such as there is no definitive explanation for why users leave a stream in different ABR. It's hard to emulate a controlled experiment in this experimental setting. However, the authors provide Puffer as a platform for researchers to continually explore unsolved problems.

2.2 Dynamic User Sensitivity

The previous paper discusses increasing QoE using a well-designed ABR scheme, the paper 'SENSEI: Aligning Video Streaming Quality with Dynamic User Sensitivity[3]' introduces another perspective, which is conducting a content-dependent sending priority for video chunks.

The authors believe users have different quality sensitivity to different segments of a video. For example, in a football game, the segment before shooting and goal would be much more sensitive compared to normal playtime. Therefore, a stagnant at shooting moment will decrease the overall QoE by a large amount. They further examine the existence of this problem by conducting an experiment on manually adding 1-sec rebuffering at different times, then letting crowdsourcing to label the QoE of the resulting videos. The result shows that QoE is largely different in all places so we could consider to trade off the insensitive chunks for sensitive segments.

Therefore, the authors proposed their framework, SENSEI. Figure 3 shows the workflow with video scheduling and crowd-sourcing.

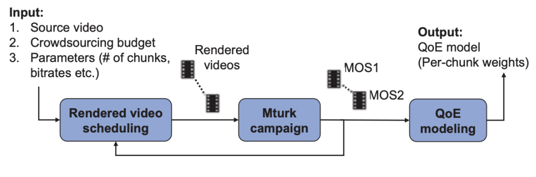


Figure 3: Workflow of SENSEI[3]

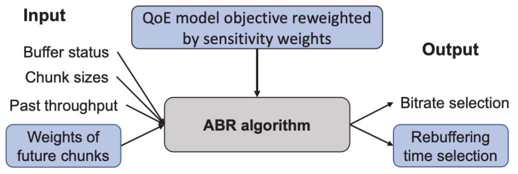
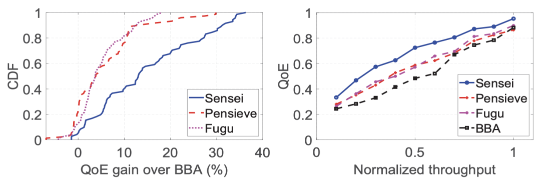


Figure 4: ABR framework of SENSEI[3]



(a) QoE gains over BBA (b) QoE vs. bandwidth usage

Figure 5: End-to-end performance of SENSEI over traditional and saliency-based ABR baselines[3]

The authors change the QoE function of SENSEI to adapt to the weight of any chunk as follows:

$$Q = \sum_{i=1}^{n} w_i q_i$$

Where w_i is the weight of the ith chunk, $q_{i,j}$ is the estimated QoE of the ith chunk of the jth rendered video. w_i will be calculated in the regression method to minimize loss.

The ABR of SENSEI is also different. Figure 4 shows the ABR framework with differences highlighted. It additionally inputs the weights and output the rebuffering time selection to unleash the sensitivity problem.

For the result part, this paper uses the Fugu in 2.1, and also other traditional ABR as the baseline. Figure 5a shows SENSEI has a high tail gaining of QoE. Figure 5b shows the bandwidth saving for SENSEI is at least 27% higher when setting QoE to 0.8 compared to other methods.

Since SENSEI highly depends on humans in the loop for labeling the sensitive frame, the authors established crowd-sourcing works on the MTurker platform. They also offer methods to eliminate the cost of crowd-sourcing but ensure accuracy.

Although the result of the experiments showing SENSEI has much higher performance compared to state-of-art methods, this paper has certain limitations. For instance, the bias that arises from differences in the perspectives of workers and real-world viewers when labeling videos. To address this, one potential solution is to assign weights to the labeled results based on the user study. Another limitation is that this method may not be applicable to less-viewed videos and live-streaming videos.

2.3 Multi-Path QUIC Transport

With the recent rising popularity of short-form videos on mobile devices, the need to improve the delivery speed, responsiveness, and other user-perceived QoE becomes more of an important discussion on mobile devices. Building on top of the popular transport layer protocol QUIC, the authors designed a multi-path transporting algorithm that increased the delivery speed of video content. Multi-path transport has been well-researched and used in industry, for instance, the widely used multi-path protocol MPTCP. However, previous multi-path transport solutions do not apply well to the short-form video application settings for three main reasons: 1) MPTCP requires operation system-level support in smartphones and thus is too costly and almost infeasible for most mobile application providers. 2) Multi-path solutions over wireless links can perform worse than single-path solutions in practice due to multi-path head-of-line (HoL) blocking. 3) Multi-path HoL blockings are hard to tackle in wireless settings even with the current state-of-the-art scheduling algorithms which make predictions about network characteristics due to the variability in wireless links. 4) Multi-path solutions are often used in conjunction with packet re-injections, which leads to more traffic and even slower delivery in practice. To tackle the current problems in the multi-path transport of short-form videos and driven by the recent trends of cross-layer network design, the author designed XLINK a multi-path transporting framework taking advantage of the user-space properties of QUIC to capture user-perceived video QoE which is then used to control multi-path scheduling and management [4].

Big picture-wise, XLINK records user-perceived QoE signals and uses a multi-path acknowledgments extension frame to carry the QoE signals to a remote video server to control its scheduling. XLINK uses packet re-injection coupled with multi-path transport but relies on the client's QoE feedback to control how aggressively the re-injections are happening from the server's side. Re-injections only happen when the packages are unacknowledged likely from multi-path HoL blacking. As seen in the graph below, XLINK's re-injection with QoE control decreases the frequency of re-injections and decreases the filled level of the buffer.

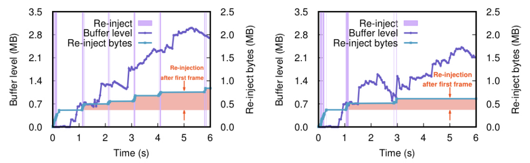


Figure 6: Re-injection without QoE (left) vs Re-injection with QoE (right) [4]

The algorithm that XLINK uses to control re-injection is named Double thresholding control. The algorithm estimates video play time left and determines if the is sufficient data on the cache. If the available play-time left exceeds the estimated maximum delivery time of in-flight packets XLINK determines that re-injection should be turned on to accelerate the packet delivery using another path [4].

By using re-injection controlled by QoE, XLINK is able to achieve a noticeable improvement in video delivery speed. The evaluation of XLINK is done through A/B testing through the deployed Taobao (one of the most prominent online shopping platforms in China) Android application. Comparison of XLINK is done against single-path QUIC, multi-path QUIC, and multi-path QUIC with re-injection but no consideration of QoE. The XLINK A/B testing reached more than 100K participants and delivered over 3 million short-form videos. Through the evaluation, the authors found promising results of XLINK making 19-50% improvement in the 99-th percentile video-chunk request completion time, 32% improvement in the 99-th percentile first-video-frame latency, and 23-67% improvement in the re-buffering rate at the expense of 2.1% redundant traffic [4].

2.4 Cross-layer Optimization for Imperfect Transmission

One hurdle that is difficult to overcome for video deliveries is the limitation of network conditions. When network conditions are bad and video frames are not delivered in time, stall playbacks happen and undermine user experience with the video. To prevent stall playbacks, previous research on dealing with varying network conditions has either been focusing on adjusting the transport layer protocol or using adaptive bitrate (ABR) algorithms on the client side. Effort on adjusting the transport layer protocol includes: tolerating a certain amount of packet loss, adding video frame deadlines-awareness, supporting real-time delivery, and using retransmissions. The client-side improvements of ABR algorithms include dynamically choosing bitrates based on available bandwidth and buffer occupancy. The authors of VOXEL presented a cross-layer ABR optimization solution that can improve the user-perceived experience when network conditions are sub-optimal for practical use. VOXEL measures the QoE importance of video frames on the server side and deliberately drops video frames that are less impactful to QoE when network conditions are suboptimal. The authors created VOXEL to be open-source and deployment friendly because it is fully compatible with DASH and is even backward compatible with VOXEL-unaware clients [1].

VOXEL is built on the insight that not all video frames require reliable delivery and frame-drops does not always result in a compromised user-perceived QoE. VOXEL builds on top the QUIC extensions to offer partially reliable transport based on this insight. VOXEL introduces a novel QoE-metric-based frame-importances measurement and ranking done on the server side. The metric VOXEL choose to evaluate QoE by is the FFmpeng's SSIM metric, but the authors show that VOXEL achieved a QoE-metric-agnostic state meaning that it also scored highly using other QoE metrics such as VMAF and PSNR compared to other state-of-the-art video delivery algorithms. The state-of-the-art algorithms VOXEL is compared to include BETA, which also pursues server-sider analysis of frame-drop importance, and VOXEL successfully outperforms BETA with its more detailed implementation and deployment support and QoE [1].

Thus, the authors generate referencing graphs across frames, which shows how each frame relates to the previous graphs and indicate each frame's importance based on the new elements they add. With this ordering, dropping the least important 10-20% frames can still maintain an SSIM score of 0.99 (The SSIM metrics have a perfect score of 1.0) which is great QoE and has a noticeable frame drops to end users. In its implementation, VOXEL re-orders the less important frames to be delivered at each segment's tail– if the less important frames do not arrive in time due to network issues, they are covered for by previous frames. VOXEL also designed its own ABR algorithm to aid its delivery. VOXEL's ABR optimizes for QoE as opposed to a utility function based on bitrate, supports partial-segment downloads, and offers options for segment abandonment. All of the adjustments are designed to work with VOXEL's focus on delivering higher QoE content with less-than-ideal network conditions [1].

VOXEL's evaluation is done against three state-of-the-art ABR algorithms (BOLA, MPC, and BETA) and a naive throughput-based ABR algorithm. In the evaluations, VOXEL experienced substantially lower buffering compared to the other algorithms. VOXEL outperforms the other algorithms in bitrates under various network conditions tested. Below is one of the result graphs in the comparisons. The authors also conducted a real user survey with VOXEL compared to BOLA which included 54 participants and 84% of the participants indicated that they rather watch the VOXEL streamed videos [1].

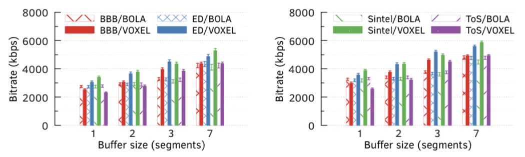


Figure 7: Comparison of VOXEL vs BOLA bitrates under different network conditions [1]

3 CONCLUSION

Current developments in video delivery include those done on designing ABR algorithms to improve QoE such as those seen in the work done by Francis Y. et. al. [2] and Xu Z. et. al. [3]. Another popular direction is to improve the transportation of video segments using techniques such as multi-path transportation. Other efforts include using cross-layer knowledge to make decisions about video frame transportation and re-transmissions.

There still remains a lot of research opportunity to tackle the transportation of videos in various network conditions–especially with the rising popularity of satellite networks. Future research also could be done on further investigating multi-path transportation of videos, congestion control, and smarter re-injection of packets.

REFERENCES

[1] Mirko Palmer, Malte Appel, Kevin Spiteri, Balakrishnan Chandrasekaran, Anja Feldmann, and Ramesh K. Sitaraman. 2021. VOXEL: Cross-Layer Optimization for Video Streaming with Imperfect Transmission. In *Proceedings of the 17th International Conference on Emerging Networking EXperiments and Technologies (CoNEXT '21)*. Association for Computing Machinery, New York, NY, USA, 359–374. https://doi.org/10.1145/3485983.3494864

[2] Francis Y Yan, Hudson Ayers, Chenzhi Zhu, Sadjad Fouladi, James Hong, Keyi Zhang, Philip Alexander Levis, and Keith Winstein. 2020. Learning in situ: a randomized experiment in video streaming.. In *NSDI*, Vol. 20. 495–511.

[3] Xu Zhang, Yiyang Ou, Siddhartha Sen, and Junchen Jiang. 2021. SENSEI: Aligning Video Streaming Quality with Dynamic User Sensitivity. In *18th USENIX Symposium on Networked Systems Design and Implementation (NSDI 21)*. USENIX Association, 303–320. https://www.usenix.org/conference/nsdi21/presentation/zhang-xu

[4] Zhilong Zheng, Yunfei Ma, Yanmei Liu, Furong Yang, Zhenyu Li, Yuanbo Zhang, Jiuhai Zhang, Wei Shi, Wentao Chen, Ding Li, Qing An, Hai Hong, Hongqiang Harry Liu, and Ming Zhang. 2021. XLINK: QoE-Driven Multi-Path QUIC Transport in Large-Scale Video Services. In *Proceedings of the 2021 ACM SIGCOMM 2021 Conference (SIGCOMM '21)*. Association for Computing Machinery, New York, NY, USA, 418–432. https://doi.org/10.1145/3452296.3472893