Poster: Content-aware Retransmission for Ultra-low-latency Video Streaming

Wei Li
wlidq@connect.ust.hk
Hong Kong University of Science and Technology
Hong Kong

Zili Meng*
zilim@ust.hk
Hong Kong University of Science and Technology
Hong Kong

CCS Concepts

• **Networks** → **Transport protocols**; • **Information systems** → **Multimedia streaming**.

Keywords

loss recovery, video streaming

ACM Reference Format:
Wei Li and Zili Meng. 2024. Poster: Content-aware Retransmission for Ultra-low-latency Video Streaming. In *ACM SIGCOMM 2024 Conference (ACM SIGCOMM Posters and Demos '24), August 4–8, 2024, Sydney, NSW, Australia.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3672202.3673746

1 Introduction

Ultra-low-latency video streaming, such as videoconferencing, has been rapidly developed and widely used by users in recent years. Especially for emerging applications like cloud gaming and virtual reality, they require both extremely low latency (tens of milliseconds even at the 99th percentile) and good video quality to achieve great quality of experience [9].

Packet loss and loss recovery are consequently critical to achieve such a goal. On the Internet, packet loss can unpredictably happen at any time due to wireless interference [8] or network congestion [3]. The packet loss is also believed to result in the degradation of video quality. To ensure good video quality, existing efforts will try their best to reliably deliver packets and recover the lost packets, with retransmissions [2, 6, 7] or forward error correction (FEC) [4, 9, 10].

However, none of the solutions are perfect. FEC-based methods will probably over-protect the video stream with considerable bandwidth costs (up to 10x of the actual loss rate [9]), and still have to rely on retransmission when the recovery fails. Retransmission, on the contrary, introduces the delay of at least one RTT. Making matters worse, the retransmitted packet will also have the head-of-line blocking effect for subsequent packets since the video decoder has to decode the whole frame together. In this case, the loss of one packet will block the decoding of the whole video frame and even subsequent frames. This has become more and more severe with the increase of video resolution and frame size recently.

(a) Normal Frame. **(b) One lost packet (index=0.75).**

Figure 1: Some packets are not imporatant to the user experience – the SSIM between two pictures are 0.97.

We identify that one missing question in the packet loss recovery is *whether these lost packets are really necessary to recover for the application.* That is to say, although the transport layer strives to reliably deliver every packet, including both TCP-based or UDP-based [8], some of the recovery efforts might be unnecessary for the application. Especially for conventional video codecs such as `libx264`, the video decoder supports the error concealment with intra-frame neighbor contents[1] – the decoder can recover the missing regions caused by packet loss using other auxiliary information. In this case, the loss of some packets has minimal influence on the video quality.

We present two decoded images from [11] in Fig. 1 to further demonstrate the case. We manually create a packet loss when transmitting Fig. 1a, pad the missing bits with zero, and decode it into Fig. 1b. We enable the error concealment in `libx264` at the decoder. The visual differences between Fig. 1a and Fig. 1b are very slight. It indicates that the lost packet is not significant to the user experience. Then we conduct an experiment to further explore the significance of each packet in a frame. As shown in Fig. 2, 40% of packet losses in our preliminary experiments are not necessary to recover with the degradation on SSIM[2] of less than 0.05, only 20% can lead to video quality degradation of more than 0.1.

Our key insight is that we will only retransmit the lost packets when they are really necessary to the quality degradation. In this way, we can reduce unnecessary retransmissions and quickly decode the video frames to minimize the impact. However, as there are numerous video codecs in use nowadays (e.g., VP8, VP9, AV1, H264), how to scale our design to different codecs is also challenging. In response, we will not modify the video codecs – instead, we will only modify the retransmission manager at the transport layer. Moreover, determining which packets are critical to the video quality is non-trivial since the significance is related to the video contents. We therefore define a metric combining user experience with load time to measure the significance of a frame to the video

*Zili Meng is the corresponding author.

ACM SIGCOMM Posters and Demos '24, August 4–8, 2024, Sydney, NSW, Australia

ACM ISBN 979-8-4007-0717-9/24/08
https://doi.org/10.1145/3672202.3673746

[1] Recently, GRACE [5] proposed to codesign the video codecs with loss recovery, which requires coordination between the encoder and decoder and brings considerable decoding overhead (NVIDIA A40 GPU).
[2] Structural similarity (SSIM [12]) is a typical metric to evaluate the video quality.

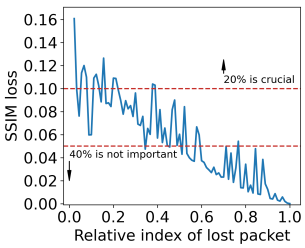

Figure 2: The loss of different packets has different significance to the user experience.

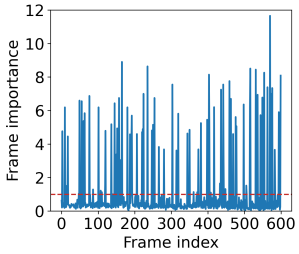

Figure 3: The metric is able to distinguish the importance of different frames.

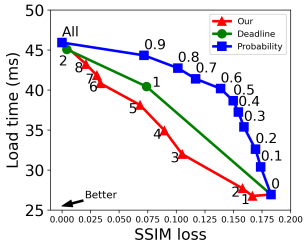

Figure 4: Compare our method with two baselines and burst length is 8 frames.

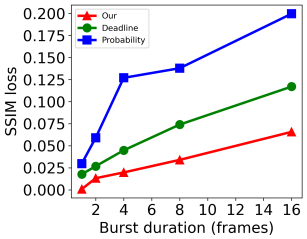

Figure 5: Compare different methods with different duration of bursts.

quality. Then we differentiate retransmission packets according to the significance. We preliminarily evaluate our idea with `libx264` and an open-source video dataset, and find that our method can reduce the end-to-end latency by up to 1/3 with negligible SSIM loss and push forward the Pareto frontier [1] compared with baselines.

2 Design

We define the load time of a frame as the total time required to receive all the packets. If we retransmit a packet, we may need more time to receive the packet. Therefore, by making decisions about retransmission for each packet, we can effectively reduce the number of retransmissions. However, the action space for decision-making is exponential growth to the total number of packets, which is deemed unacceptable. Ideally, we can retransmit only the important packets and we find that the significance of packets is determined by two factors:

Video quality. As shown in Fig. 2, the SSIM loss decreases when the index of the lost packet increases. This is also related to the mechanism of video codecs – one video frame contains tens to hundreds of packets, and packets at the beginning of a frame are more critical than packets at the end. In Fig. 2, 40% of the packet losses will only incur an SSIM loss of less than 0.05, which is negligible to the quality degradation.

Retransmission time. If the retransmitted packet is at the beginning of a frame, the time cost is negligible because we can receive other packets when waiting for the packet. On the contrary, when the last packet of a frame is lost, we have to wait for at least one RTT before we can decode the frame. In other words, the time cost of retransmission in a frame is inversely proportional to the index of the lost packet.

We define the expectation of SSIM loss as $\mathbb{E}[S_i|r]$, for frame i under a specific loss rate r, assuming no retransmission. This expectation represents the significance of frame i to the overall video quality. Additionally, we consider the expected time cost of retransmission, which is influenced by the frame size f_i. The ratio $Q_i = \mathbb{E}[S_i|r]/f_i$ is the metric to measure the importance of a frame. Furthermore, we define Q_a as the average of Q_i values across the entire video and λ as the coefficient to adjust the threshold. Then we adopt the following approach:

- If λQ_i is higher than Q_a, it indicates that frame i is crucial for video quality, requiring retransmission of the lost packets.
- If λQ_i falls below the average level, it suggests that those packets of frame i is not worth retransmitting.

3 Evaluation

We estimate $\mathbb{E}[S_i|r]$ using the Monte Carlo method. Subsequently, we perform an experiment to demonstrate the variability of Q_i among different frames and its effectiveness in differentiating them. In Fig. 3, red line represents Q_a. Notably, certain frames exhibit peak values significantly higher than Q_a. This observation indicates that the metric serves as a high signal-to-noise ratio signal, enabling easy identification of important frames.

To evaluate the effectiveness of our solution, we compare it against two baselines: the deadline-aware and the probabilistic loss recovery method. Deadline-aware baseline sets the maximum number of retransmissions π per lost packet and will give up retransmission after π retransmissions. The probabilistic method is the probability p of retransmission per lost packet. We also test the different values of $p = \{0, 0.1, 0.2, ..., 0.9\}$, $\pi = \{0, 1, 2\}$, $\lambda = \{0, 1, 2, ..., 8\}$, and the case that retransmits all packets.

We implement a simple simulator using `libx264`. In the simulation, we suppose RTT is 15 ms, bandwidth is 40 Mbps, the loss rate is 10% and the frame rate of the video is 30fps, taking from the typical network conditions [9]. We suppose only one burst happens during the whole video. Then we use the video from YT-UGC Dataset [11] as the test video, each with a duration of 20 seconds. In Fig 4, the burst length is 8 frames and the simulation result shows that our method reduces the SSIM loss with the same load time. For example, if we limit the load time to 40ms, the SSIM loss of our, deadline-aware and probabilistic method are 0.025, 0.075, and 0.14.

Then we set burst length from 1 frame to 16 frames in the experiment following [9]. Fig 5 shows the performances of these three methods with different burst lengths. We limit the load time to 40ms and compare the SSIM loss results. Our method can achieve the best result in all scenarios and reduce the SSIM loss with baselines by 50%-95%.

4 Conclusion and Future Work

This poster proposes a content-aware loss recovery mechanism that enables receivers to optimize the QoE when packet loss occurs by only retransmitting important frames. The simulation experiment demonstrates that our method breaks the trade-off between user experience and load time.

For future work, one challenge is how to collect load time and content information efficiently for live video streaming. Another challenge is addressing the handling of more frequent and longer bursts. Currently, we only consider the scenario where bursts occur every 20 seconds.

References

[1] Pareto front-wikipedia. https://en.wikipedia.org/wiki/Pareto_front.

[2] Mark Allman, Vern Paxson, and Ethan Blanton. Tcp congestion control. IETF RFC 5681, 2009.

[3] Gaetano Carlucci, Luca De Cicco, Stefan Holmer, and Saverio Mascolo. Congestion control for web real-time communication. *IEEE/ACM Transactions on Networking*, 2017.

[4] Ke Chen, Han Wang, Shuwen Fang, Xiaotian Li, Minghao Ye, and H. Jonathan Chao. Rl-afec: Adaptive forward error correction for real-time video communication based on reinforcement learning. In *Proc. ACM MMSys*, 2022.

[5] Yihua Cheng, Ziyi Zhang, Hanchen Li, Anton Arapin, Yue Zhang, Qizheng Zhang, Yuhan Liu, Kuntai Du, Xu Zhang, Francis Y. Yan, Amrita Mazumdar, Nick Feamster, and Junchen Jiang. GRACE: Loss-Resilient Real-Time video through neural codecs. In *Proc. USENIX NSDI*, pages 509–531, Santa Clara, CA, April 2024. USENIX Association.

[6] Yuchung Cheng, Neal Cardwell, Nandita Dukkipati, and Priyaranjan Jha. The rack-tlp loss detection algorithm for tcp. IETF RFC 8985, 2021.

[7] Sadjad Fouladi, John Emmons, Emre Orbay, Catherine Wu, Riad S Wahby, and Keith Winstein. Salsify: Low-latency network video through tighter integration between a video codec and a transport protocol. In *Proc. USENIX NSDI*, 2018.

[8] Zili Meng, Yaning Guo, Chen Sun, Bo Wang, Justine Sherry, Hongqiang Harry Liu, and Mingwei Xu. Achieving Consistent Low Latency for Wireless Real Time Communications with the Shortest Control Loop. In *Proc. ACM SIGCOMM*, 2022.

[9] Zili Meng, Xiao Kong, Jing Chen, Bo Wang, Mingwei Xu, Rui Han, Honghao Liu, Venkat Arun, Hongxin Hu, and Xue Wei. Hairpin: Rethinking packet loss recovery in edge-based interactive video streaming. In *Proc. USENIX NSDI*, 2024.

[10] Michael Rudow, Francis Y Yan, Abhishek Kumar, Ganesh Ananthanarayanan, Martin Ellis, and KV Rashmi. Tambur: Efficient loss recovery for videoconferencing via streaming codes. In *Proc. USENIX NSDI*, pages 953–971, 2023.

[11] Yilin Wang, Sasi Inguva, and Balu Adsumilli. Youtube ugc dataset for video compression research. In *2019 IEEE 21st International Workshop on Multimedia Signal Processing (MMSP)*, pages 1–5. IEEE, 2019.

[12] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.