APL: Adaptive Preloading of Short Video with Lyapunov Optimization

Conference Paper · December 2020

DOI: 10.1109/VCIP49819.2020.9301886

8 authors, including:

Yixuan Ban
Peking University
7 PUBLICATIONS **237** CITATIONS
SEE PROFILE

Xinggong Zhang
Peking University
69 PUBLICATIONS **833** CITATIONS
SEE PROFILE

Zongming Guo
Yantai Nanshan University
211 PUBLICATIONS **3,520** CITATIONS
SEE PROFILE

Zhimin Xu
Peking University
9 PUBLICATIONS **276** CITATIONS
SEE PROFILE

APL: Adaptive Preloading of Short Video with Lyapunov Optimization

Haodan Zhang[1], Yixuan Ban[1], Xinggong Zhang[1,*], Zongming Guo[1]
Zhimin Xu[2], Shengbin Meng[2], Junlin Li[2], Yue Wang[2]
[1]Wangxuan Institute of Computer Technology, Peking University, Beijing, P.R. China
[2]Beijing Bytedance Technology Co., Ltd., Beijing, P.R. China
{pkuzhd, banyixuan, zhangxg, guozongming}@pku.edu.cn

***Abstract*—Short video applications, like TikTok, have attracted many users across the world. It can feed short videos based on users' preferences and allow users to slide the boring content anywhere and anytime. To reduce the loading time and keep playback smoothness, most of the short video apps will preload the recommended short videos in advance. However, these apps preload short videos in fixed size and fixed order, which can lead to huge playback stall and huge bandwidth waste. To deal with these problems, we present an Adaptive Preloading mechanism for short videos based on Lyapunov Optimization, also called APL, to achieve near-optimal playback experience, i.e., maximizing *playback smoothness* and minimizing *bandwidth waste* considering users' sliding behaviors. Specifically, we make three technical contributions: (1) We design a novel short video streaming framework which can dynamically preload the recommended short videos before the current video is downloaded completely. (2) We formulate the preloading problem into a playback experience optimization problem to maximize the playback smoothness and minimize the bandwidth waste. (3) We transform the playback experience optimization problem during the whole viewing process into a single-step greedy algorithm based on the Lyapunov optimization theory to make the online decisions during playback. Through extensive experiments based on the real datasets that generously provided by TikTok, we demonstrate that APL can reduce the stall ratio by 81%/12% and bandwidth waste by 11%/31% compared with no-preloading/fixed-preloading mechanism.**

***Index Terms*—short video preload, adaptive streaming, Lyapunov optimization**

I. Introduction

The mobile short video applications, such as TikTok, have developed rapidly and become the time killers. According to [1], TikTok was the most downloaded non-game app worldwide for January 2020, with more than 104.7 million installs. More and more people choose to use the TikTok on their commute [2]. TikTok allows users to record short videos (usually about 15 seconds [3]) and then upload them to the Internet to share with others. It can also recommend short videos to users and allow them to slide among the recommended videos [4]. To reduce the loading time and keep playback smoothness, TikTok used to preload the short videos on the *recommendation queue*.

*Corresponding author. E-mail:zhangxg@pku.edu.cn.

This work was supported by the National Key RD program of China (2019YFB1802700, 2019YFF0302903), Toutiao Research Funding(CT20200414000744).

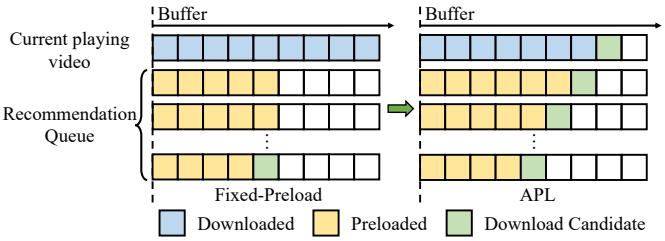


Fig. 1: Preloading mechanism of APL.

However, the existing preload mechanism can only preload short videos when the current playing video is downloaded completely, and it can only preload short videos with fixed size and fixed order. When the current video is slid, the unplayed bits will be wasted, and this bandwidth waste will significantly increase the risk of stall. Besides, different users have different sliding behaviors. Some users prefer to finish watching each video, whereas some users prefer to slide over videos to find the interesting content. Preloading videos with the fixed size and fixed order would cause stall for users who tend to finish watching each video and cause bandwidth waste for users who tend to slide.

To tackle these problems, we propose an adaptive preloading approach for short video streaming, called APL, to adaptively choose which video to preload before the current video is fully downloaded. As shown in Fig. 1, assuming there are N videos on the recommendation queue, APL can choose $N+1$ videos, including the current short video to download, according to the buffer status and each user's predictive viewing duration. Especially, the existing mechanism's download order is fixed, while the APL can adaptively choose the current video as well as the recommended ones to download. To maximize the user's playback experience, we consider two preloading objectives on our framework. Firstly, for short video streaming, providing *playback smoothness* for users is the first priority. Besides, reducing the user's traffic cost on 4G/LTE network as well as the content provider's delivery cost on Content Delivery Network (CDN), i.e., reducing *bandwidth waste* is also crucial. Thus we firstly formulate the preloading problem into a global playback smoothness maximization and bandwidth waste minimization problem. Then we utilize the Lyapunov optimization theory [5] to transform the global optimization into a single-

978-1-7281-8068-7/20/$31.00

step greedy algorithm in a provable near-optimal way to make online decisions. To the best of our knowledge, it is the first paper focusing on short video preloading approaches and the first paper to evaluate the effectiveness on the real datasets from TikTok. The results of the evaluation show that APL can reduce the stall ratio by 81%/12% and bandwidth waste by 11%/31% compared with no-preloading/fixed-preloading mechanism.

II. System Overview

In this section, we introduce the main objective of our framework and present the problem formulation.

A. Short Video Streaming Objective

We consider the playback smoothness as well as the bandwidth saving as our framework's objectives. In order to represent the playback smoothness S, we denote R as the total video duration that watched by the user and denote T_e as the total playback time including the stall time and the video watching time. Then the playback smoothness S can be defined as:

$$S = \frac{R}{T_e}. \tag{1}$$

For the same watching time R, longer playback time T_e means longer stall. By maximizing S, the stall time can be minimized.

Intuitively, achieving the optimal playback smoothness under limited bandwidth means almost all the downloaded bits are watched by users. In other words, we can approximately optimize the bandwidth waste by optimizing playback smoothness S individually.

B. Problem Formulation

We divide the download process into non-overlapping consecutive slots k and choose which short video to download at each slot's start time t_k, where $k \in \{1, 2, ..., K\}$. A slot ends when the download ends, or when the user slides away the downloaded video. Assuming there are L short videos watched by the user in total, and the user is watching video l_k at slot k, where $l \in \{1, 2, ..., L\}$. The client can only download one video for each slot, denoted as $a_{k,l}$, where $a_{k,l} = 1$ means to download short video l for p seconds and $a_{k,l} = 0$ otherwise. Thus total downloaded duration of video l at the end of slot k can be represented by $A_{k,l} = \sum_{i=1}^{k} a_{i,l}$. We denote each short video's expected viewing duration at the end of slot k as $R_{k,l}$, which can be formulated as

$$R_{k,l} = min\{A_{k,l}, P_{k,l}\}, \tag{2}$$

where $P_{k,l}$ represents predicted viewing duration of video l at slot k. The expected viewing duration $R_{k,l}$ can only take the minimum of $A_{k,l}$ and $P_{k,l}$. Especially, we set $R_{0,l} = 0$. Based on that, the total viewing duration R in (1) can be represented by the sum of each slot's viewing duration increment I_k:

$$R = \sum_{k=1}^{K} I_k = \sum_{k=1}^{K} \sum_{l=1}^{L} (R_{k,l} - R_{k-1,l}), \tag{3}$$

where I_k can be formulated by the sum of each video's viewing duration increment $(R_{k,l} - R_{k-1,l})$ on slot k.

To represent the download process, we denote the recommendation queue can accommodate N short videos, and thus there are $N + 1$ short videos that can be downloaded at each slot, including the current playing short video. Moreover, we denote the buffer length of each video as $Q_{k,l}$ seconds in slot k. At each slot, $\sum_{l=l_k}^{l_k+N} a_{k,l} \leq 1$ always holds, where the $\sum_{l=l_k}^{l_k+N} a_{k,l} = 0$ means any short video will not be downloaded, and this no-download slot will last τ seconds each. Based on that, the total playback time T_e can be represented by the sum of each slot's duration T_k as follows:

$$T_e = \sum_{k=1}^{K} T_k, \tag{4}$$

$$T_k = \begin{cases} \frac{\sum_{l=l_k}^{l_k+N} a_{k,l} \cdot C_l \cdot p}{B_k}, & \text{if } \sum_{l=l_k}^{l_k+N} a_{k,l} = 1 \\ \tau, & \text{otherwise,} \end{cases} \tag{5}$$

where C_l is the average bitrate of the short video l and B_k is the average bandwidth of slot k.

Given (3) and (4), our objective (1) can be formulated as

Problem 1.

$$\arg\max_{\{a_{k,l}\}} \frac{\sum_{k=1}^{K} I_k}{\sum_{k=1}^{K} T_k}. \tag{6}$$

However, solving **Problem 1** need all the bandwidths and the viewing duration of each video in advance, which is impossible. In order to solve **Problem 1** online in near-optimal, we leverage the Lyapunov optimization theory to design a single-step greedy algorithm in the following section.

III. Lyapunov Optimization

In this section, we introduce our online adaptive preloading algorithm, which utilizes the buffer status to achieve near-optimal playback experience. We firstly formulate the buffer length evolvement into a mathematical representation. Then we transform **Problem 1** into a single-step greedy algorithm to make decisions online.

A. Buffer Evolvement

Let $\boldsymbol{Q}_k = (Q_{k,1}, ..., Q_{k,L})$ denote the buffer length of all videos in slot k. If the client decides to download video l, the buffer l will be enlarged by one chunk's duration p. Assuming that the user watches video l for $T_{k,l}^{play}$ seconds in slot k. Then the buffer evolvement of each video in each slot k can be formulated as:

$$Q_{k+1,l} = max\{0, Q_{k,l} - T_{k,l}^{play}\} + a_{k,l} \cdot p \tag{7}$$

We define the Lyapunov function $\Gamma(\boldsymbol{Q}_k)$ at t_k as follows:

$$\Gamma(\boldsymbol{Q}_k) = \frac{1}{2} \sum_{l=1}^{L} w_{l-l_k} Q_{k,l}^2. \tag{8}$$

It is a non-negative value and is equal to zero only when all of the buffers are empty. $\{w_{l-l_k}\}$ is a collection of weights

of video buffers. Users usually prefer to watch recommended videos sequentially, and thus the upcoming videos have higher priority, i.e., $\{w_{l-l_k}\}$ satisfies:

$$w_{1-l_k} = ... = w_0 \leq w_1 \leq ... \leq w_N = ... = w_{L-l_k}. \quad (9)$$

We define the one-slot *Lyapunov drift* $\Delta \boldsymbol{Q}$ as follows to represent the buffer change:

$$\begin{aligned}\Delta \boldsymbol{Q}_k =& \mathbb{E}\{\Gamma(\boldsymbol{Q}_{k+1}) - \Gamma(\boldsymbol{Q}_k)|\boldsymbol{Q}_k\} \\ =& \frac{1}{2}\sum_{l=1}^{L} \mathbb{E}\{w_{l-l_{k+1}} Q_{k+1,l}^2 - w_{l-l_k} Q_{k,l}^2 |\boldsymbol{Q}_k\} \\ =& \frac{1}{2}\sum_{l=l_k}^{l_k+N} \mathbb{E}\{w_{l-l_{k+1}} Q_{k+1,l}^2 - w_{l-l_k} Q_{k,l}^2 |\boldsymbol{Q}_k\}.\end{aligned} \quad (10)$$

Only video $l \in \{l_k, l_k+1, ..., l_k+N\}$ can be downloaded, so the drift of video $l \notin \{l_k, l_k+1, ..., l_k+N\}$ is zero. As shown in appendix document, using (7), we can minimize (10) by minimizing $\sum_{l=l_k}^{l_k+N} w_{l-l_k} \cdot Q_{k,l} \cdot a_{k,l}$.

B. Online Adaptation Algorithm

According to the *drift-plus-penalty* theory in Lyapunov optimization [5], we can greedily maximize the ratio of the increment I_k minus drift to the slot's duration T_k over each slot to achieve near-optimal playback experience. Based on that, the **Problem 1** can be rewritten as follows:

Problem 2.

$$\arg\max_{\{a_{k,l}\}} \frac{\lambda I_k - \sum_{l=l_k}^{l_k+N} w_{l-l_k} \cdot Q_{k,l} \cdot a_{k,l} \cdot p}{T_k}. \quad (11)$$

λ is a control parameter representing the trade-off between playback smoothness and buffer length. As shown in (11), the smaller w means the higher priority on I_k, i.e., high priority on downloading new video content on video l, this is why we set $\{w_{l-l_k}\}$ as (9). By solving (11), we can achieve a playback smoothness deviating by at most $O(\frac{1}{\lambda})$ from optimality of **Problem 1** while maintaining buffer length bound of $O(\lambda)$. The following theorem provides a theoretical performance guarantee. The proof is in [6].

Theorem 1. *The playback smoothness in APL satisfies:*

$$S^\dagger \geq S^* - \frac{p^2 + \Psi}{2\lambda} \sum_{n=0}^{N} w_n \quad (12)$$

where $S^\dagger$ is the playback smoothness of APL, S^* is the optimal playback smoothness in theory.

In practice, we will loop all short video l, where $l \in \{l_k, ..., l_k + N\}$, and select the best short video l^* that makes (11) the largest. Specifically, if downloading nothing ($\sum_{l=l_k}^{l_k+N} a_{k,l} = 0$) can achieve the maximum, the client will wait for τ seconds to switch to the next slot.

IV. Performance Evaluation

In this section, we evaluate the performance of APL by comparing it with three approaches under simulation experiments over real network traces and playback traces. The result shows our APL can achieve lower stall and less waste.

A. Experimental Setup

Trace Selection. To evaluate our APL's performance, we select 11 real 4G/LTE network traces from the public dataset in [7] and use playback traces provided by TikTok (8300 traces). Specifically, each playback trace includes information of videos (bitrate, duration, and viewing duration) and information of users' behavior (when to slide away).

Parameter Setting. In APL, we set λ as 20, p as 1, and N as 5, which means we can choose from 6 videos to download at each slot, including the current playing video. Accordingly, the parameter set $\{w\}$ can be set to a 6-dimensional vector, $\{2, 2, 3, 4, 5, 6\}$.

Comparing Algorithms. To demonstrate APL's effectiveness, we implement three approaches for comparison:

- **Perfect** downloads the video sequentially based on the ground-truth of each video's viewing duration.
- **No-Preload** only downloads the current playing video. It stops when the current playing video download ends.
- **Fixed-Preload** downloads the current playing video first. When the current playing video download ends, it preloads the videos on the *recommendation queue* in order, with a maximum of 800KB for each video.

Viewing Duration Prediction. For experiments, we use a sliding window to predict the viewing duration. Specifically, the average viewing proportion of the previous m ($\min\{5, l_k - 1\}$) videos before the current video is taken as the prediction proportion of the current video and videos on the *recommendation queue*. If the user has not watched any video yet, we use 50% as the prediction proportion. For the current video, if the prediction duration is less than the current viewing duration plus 5s, we will use the current viewing duration plus 5s as the prediction duration.

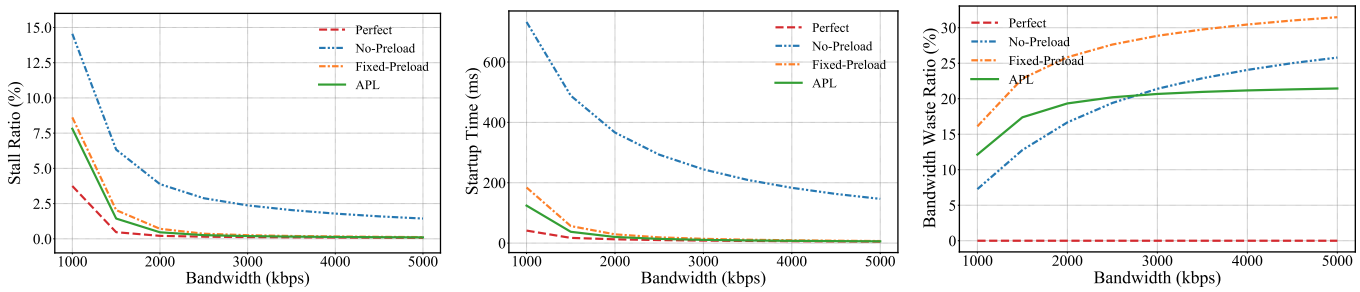
We use stall ratio and average startup time to evaluate the playback smoothness. The ratio of unviewed bytes to downloaded bytes is used to represent the waste ratio.

B. Performance under fixed bandwidth

We evaluate APL under fixed bandwidth to explore the potential performance across different network conditions. Specifically, the fixed bandwidth ranges from 1000kbps to 5000kbps with an interval of 500kbps. For each bandwidth level, the playback experiment metrics are averaged by 8300 traces.

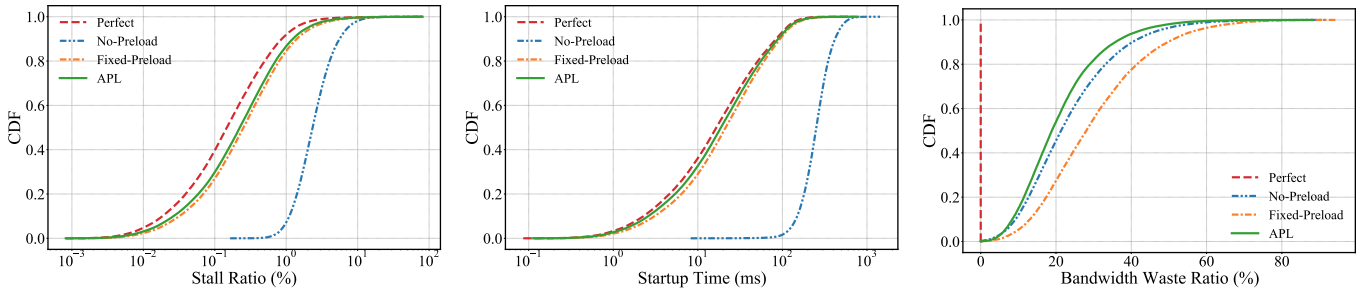
As shown in Fig. 2, for all approaches, the stall ratio and startup time decrease as the bandwidth increases, while the waste ratio increases.

In terms of playback smoothness, No-Preload is the worst approach because it doesn't have any preloading mechanism. Perfect is the best approach because its preloading can reach exactly how long the user will see it. When the bandwidth is between 1000kbps and 2000kbps, APL is notably better than Fixed-Preload. When the bandwidth is higher than 2500kbps, the gap between the two approaches is smaller, and they are both close to Perfect. This is because when the bandwidth is large enough (higher than the bitrate of any video), there will be almost no stall and very low startup time.



(a) Stall Ratio (b) Startup Time (c) Bandwidth Waste Ratio

Fig. 2: Performance under fixed network bandwidth.



(a) Stall Ratio (b) Startup Time (c) Bandwidth Waste Ratio

Fig. 3: Performance distribution on 4G/LTE traces.

As for the waste ratio, Perfect has no waste because it will not download any bytes that the user will not watch. Fixed-Preload has the largest waste ratio because it preloads fixed size, whether users will see them. Our APL takes viewing duration prediction into account and treats waste as one of the objectives, so it has lower waste than No-Preload and Fixed-Preload. As bandwidth increases to 3000kbps, APL's waste is even lower than No-Preload's.

C. Performance under real bandwidth traces

We evaluate APL with 11 real 4G/LTE traces and 8300 playback traces. We calculate each playback's metrics to plot the Cumulative Distribution Function (CDF) in Fig. 3.

No-Preload has an inferior performance in playback smoothness. APL has a 12% less stall ratio and 8% less startup time compared to Fixed-Preload. Compared with No-Preload, the two metrics of APL are decreased by 81% and 87% respectively. Meanwhile, APL's waste ratio is 31% less than Fixed-Preload's and 11% less than No-Preload's.

The result shows that APL can reduce stall ratio and startup time while causing less waste of bandwidth. We have reason to believe that APL can improve users' experiments and reduce waste.

V. Conclusion

In this paper, we propose an adaptive preloading mechanism for short video based on Lyapunov optimization, called APL. We formulate the multi-video streaming problem into a global smoothness maximization and waste minimization problem to optimize the playback experience. Moreover, the Lyapunov optimization theory is utilized to transform the problem into a single-step greedy algorithm and demonstrate its near-optimality. Through extensive experiments based on real datasets that generously provided by TikTok, we demonstrate that APL can reduce the stall ratio by 81%/12% and bandwidth waste by 11%/31% compared with not-preloading mechanism/fixed-preloading mechanism.

References

[1] "https://sensortower.com/blog/top-apps-worldwide-january-2020-by-downloads."

[2] Q. Li, Y. Zhang, H. Huang, and J. Yan, "Deep learning-based short video recommendation and prefetching for mobile commuting users," in *Proceedings of the ACM SIGCOMM 2019 Workshop on Networking for Emerging Applications and Technologies*, 2019, pp. 49–55.

[3] X. Lu and Z. Lu, "Fifteen seconds of fame: A qualitative study of douyin, a short video sharing mobile application in china," in *International Conference on HCI*. Springer, 2019, pp. 233–244.

[4] D. Li and W. Qi, "User experience research of short video app based on feed flow," in *2019 12th CISP-BMEI*. IEEE, 2019, pp. 1–6.

[5] M. J. Neely, "Stochastic network optimization with application to communication and queueing systems," *Synthesis Lectures on Communication Networks*, vol. 3, no. 1, pp. 1–211, 2010.

[6] "Proof of apl," https://github.com/pkuzhd/APL/blob/master/Proof_of_APL.pdf.

[7] J. van der Hooft, S. Petrangeli, T. Wauters, R. Huysegems, P. R. Alface, T. Bostoen, and F. De Turck, "HTTP/2-Based Adaptive Streaming of HEVC Video Over 4G/LTE Networks," *IEEE Communications Letters*, vol. 20, no. 11, pp. 2177–2180, 2016.