MIT 6.829: Computer Networks Fall 2017

Lecture 18: Video Streaming

Scribe: Zhihong Luo, Francesco Tonolini

1 Overview

This lecture is on a specific networking application: video streaming. In particular, we discussed Pensieve, a reinforcement learning based video streaming technology [1].

2 Dynamic Adaptive Streaming over HTTP (DASH)

Because video is stored in chunks (few seconds each) at different discretized bitrates (240p, 360p, 720p) at the server, and Dynamic Adaptive Streaming over HTTP (DASH) is responsible for getting the next chunk at an appropriate bitrate according to the network throughput. If the chosen bitrate is larger than the network throughput, the playback buffer will be drained and cause rebuffering in the future. If the chosen bitrate is too small, extra bandwidth will be wasted, and users can not enjoy a better video quality. Therefore, client-side video players employ adaptive bitrate (ABR) algorithms to optimize user quality of experience (QoE). Figure 1 is a illustration of what ABR algorithms try to achieve.

User-perceived quality-of-experience (QoE) is critical in Internet video applications as it impacts revenues for content providers and delivery systems. The ultimate goal of bitrate adaptation is to improve the QoE of users in order to achieve higher long-term user engagement. The widely-adapted QoE formulation is as follow:

$$QoE = \sum_{k=1}^{K} q(R_k) - \lambda \sum_{k=1}^{K-1} |q(R_{k+1}) - q(R_k)| - \mu \sum_{k=1}^{K} (\frac{d_k(R_k)}{C_k} - B_k) - \mu_s T_s$$

We model a video as a set of consecutive video segments or chunks, each of which contains L seconds of video. Each chunk is encoded at different bitrates. Let R be the set of all available bitrate levels. The video player can choose to download chunk k at bitrate $R_k \in R$. Let $d_k(R_k)$ be the size of chunk k encoded at bitrate R_k. The higher bitrate is selected, the higher video quality is perceived by the user. Let $q() : R \to \mathbb{R}$ be a non-decreasing function which maps selected bitrate R_k to video quality perceived by user $q(R_k)$. T_s represents the startup delay

Figure 2 helps illustrate the conceptual operation of the video player. At time t_k, the video player starts to download chunk k. The download time for this chunk will be $d_k(R_k)/C_k$; i.e., it depends on the size of selected chunk with bitrate R_k, as well as average download speed C_k experienced during this download process.

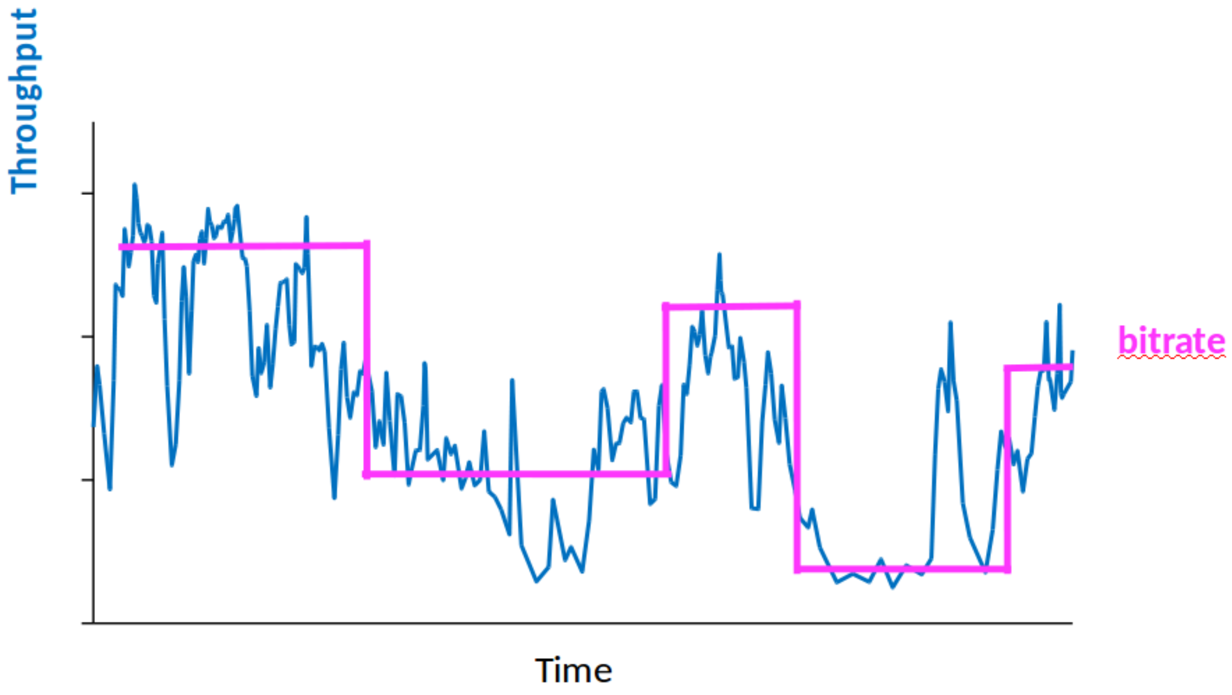

Figure 1: Adaptive Bitrate (ABR) Algorithms

Different terms in this QoE formulation represent different goals that need to be achieved. Here λ, μ, μ_s are non-negative weighting parameters corresponding to video quality variations, rebuffering time and startup delay, respectively.

Average Video Quality: The average per-chunk quality over all chunks: $\frac{1}{K}\sum_{k=1}^{K} q(R_k)$

Rebuffer: For each chunk k rebuffering occurs if the download time $\frac{d_k(R_k)}{C_k}$ is higher than the playback buffer level when the chunk download started (i.e., B_k). Thus, the total rebuffer time is: $\sum_{k=1}^{K}(\frac{d_k(R_k)}{C_k} - B_k)$

Minimize startup delay: Assume $T_s \ll B_{max}$

Average Quality Variations: This tracks the magnitude of the changes in the quality from one chunk to another: $\frac{1}{K-1}\sum_{k=1}^{K-1} |q(R_{k+1}) - q(R_k)|$

There are four types of bitrate adaptation algorithms: rate-based algorithms, buffer-based algorithms, hybrid algorithms and learning-based algorithms. Rate-based algorithms use a throughput predictor to predict throughput in the future. The benefits of doing this is to be able to deliver high play-back rate if the predictor is right. However, if the predictor fails to predict accurately, these algorithms will fail to pick up the optimal bitrate. A previous state of art ABR algorithm belongs to this category[2], which repeatedly solve the optimization function to select the best bit rate according to the predicted throughput. However, MPCs performance relies on an accurate

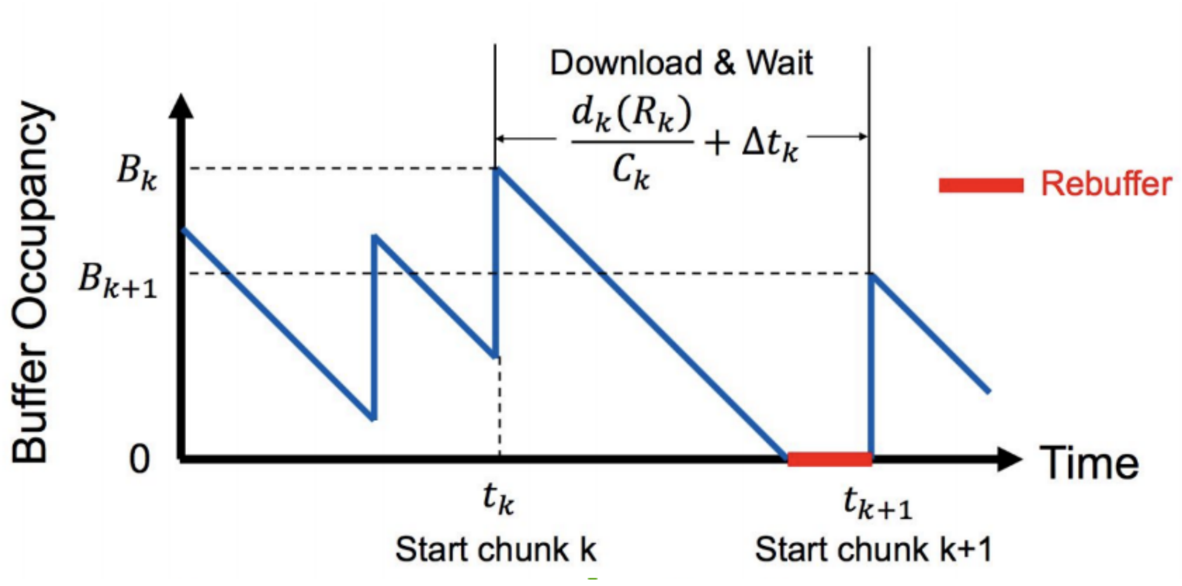

Figure 2: Illustration of Buffer Dynamics

model of the system dynamicsparticularly, a forecast of future network throughput. This makes MPC sensitive to throughput prediction errors.

Buffer-based algorithms pick bitrate based on buffer occupancy and try to keep buffer at a constant level, which is good for preventing rebuffering events. But these algorithms usually make conservative decisions, and sacrifice user experience as a result. Hybrid algorithms use both throughput prediction and buffer occupancy to make decisions. According to Pensieve, rate-based, buffer-based and hybrid methods all require significant tuning and do not generalize to different network conditions and QoE objectives[1]. Therefore, Pensieve uses a novel learning based method, which picks appropriate bit rates by learning from previous experience.

3 Pensieve

From a high-level point of view, Pensieve trains a reinforcement learning neural network model that selects bitrates for future video chunks based on observations collected by client video players. Pensieve does not rely on pre-programmed models or assumptions about the environment. Instead, it learns to make ABR decisions solely through observations of the resulting performance of past decisions. As a result, Pensieve automatically learns ABR algorithms that adapt to a wide range of environments and QoE metrics.

3.1 Reinforcement Learning

As shown in Figure 3, RL considers a general setting in which an agent interacts with an environment. At each time step t, the agent observes some state s_t, and chooses an action a_t. After applying the action, the state of the environment transitions to s_{t+1} and the agent receives a reward

r_t. The goal of learning is to maximize the expected cumulative discounted reward: $\mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r_t]$, where $\gamma \in (0, 1]$ is a factor discounting future rewards.

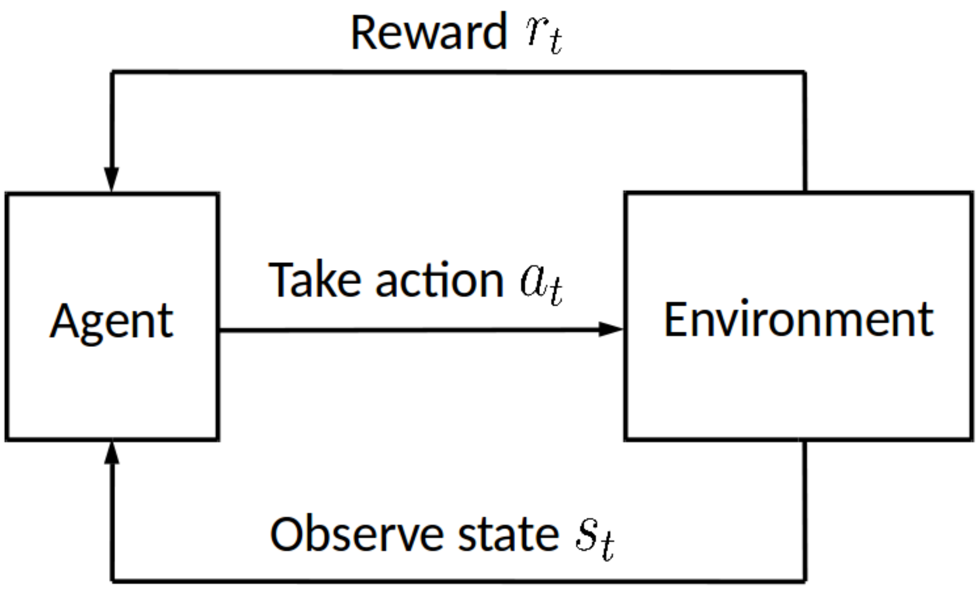

Figure 3: Reinforcement Learning Agent

3.2 RL Agent of Pensieve

As shown in Figure 4, the decision policy guiding the ABR algorithm is not handcrafted. Instead, it is derived from training a neural network. The ABR agent observes a set of metrics including the client playback buffer occupancy, past bitrate decisions, and several raw network signals (e.g., throughput measurements) and feeds these values to the neural network, which outputs the action, i.e., the bitrate to use for the next chunk. The resulting QoE is then observed and passed back to the ABR agent as a reward. The agent uses the reward information to train and improve its neural network model.

3.3 Training Methodology of Pensieve

To accelerate the video download and update process, Pensieve trains the agent in a simple simulation environment that models the dynamics of video streaming with real applications. Policy gradient descent is used to make sure that the agent adapts to a better policy in training. Multi-agent parallel training enables training on different processors or different machines, which greatly accelerates the training process.

References

[1] Mao, H., Netravali, R. and Alizadeh, M., 2017. Neural Adaptive Video Streaming with Pensieve. *Proceedings of the 2017 ACM SIGCOMM Conference*

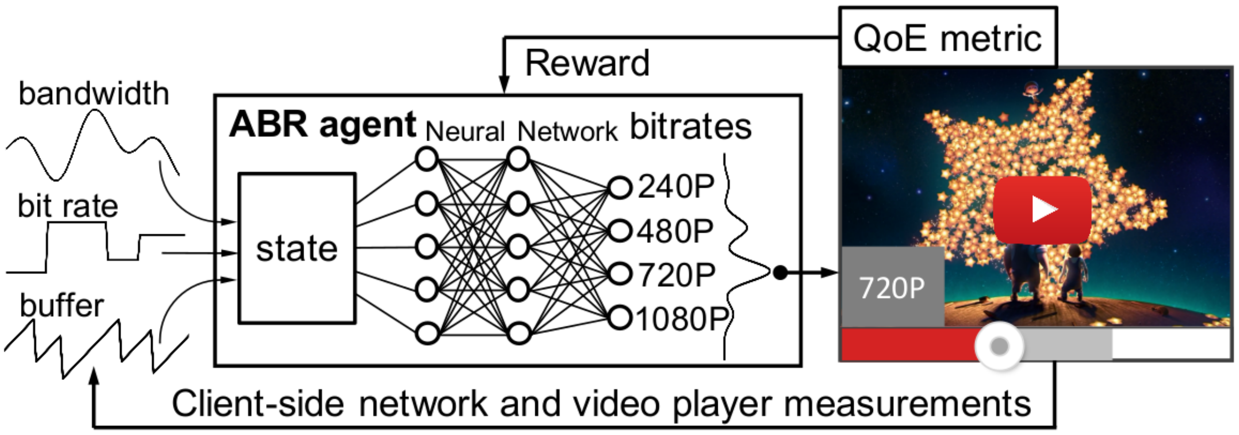

Figure 4: Applying reinforcement learning to bitrate adaptation

[2] Yin, X., Jindal, A., Sekar, V. and Sinopoli, B., 2015. A control-theoretic approach for dynamic adaptive video streaming over http. ACM SIGCOMM Computer Communication Review, 45(4), pp.325-338.