QoE Control for Dynamic Adaptive Video Streaming Over HTTP at Access Point

Yukun Yuan*, Shan Lin*, Gang Zhou†
*Stony Brook University, †College of William and Mary
Email: {yukun.yuan, shan.x.lin}@stonybrook.edu, {gzhou}@cs.wm.edu

***Abstract*—Recent research shows that when multiple dynamic adaptive streaming over HTTP (DASH) clients stream Internet videos simultaneously via a wireless access point (AP), they may experience poor quality of experience (QoE). This is because DASH clients in a wireless local area network (WLAN) independently estimate current available bandwidth and make bitrate selections. Lacking coordination in the WLAN causes competition for downlink bandwidth, introducing unnecessary bandwidth fluctuations especially when some clients start streaming. Such bandwidth fluctuations lead to cascading video bitrate changes on each stream, resulting in dramatic video quality variations, in the worst case, for example, video stalls. To address this problem, it is essential to mitigate the bandwidth competition by coordinating DASH clients in the WLAN. In this paper, we propose a QoE control framework at the AP to dynamically allocate bandwidth for each DASH client based on their real-time streaming performance feedback. To deal with uncertainty in Internet traffic besides streaming traffic, we design robust control algorithms to minimize stalls while balancing bandwidth sharing among clients. We implement and evaluate our solution with real systems under various settings. Extensive evaluation results show that our design significantly reduces average video stalling duration by 57.1% and improves the average video bitrate by 42.0% for all clients when compared to state-of-the-art solutions.**

I. Introduction

Internet video traffic accounts for an overwhelming majority of Internet traffic. A recent study shows that 82.0% of all IP traffic will be video in 2021 [1]. Existing video streaming services, such as YouTube and Netflix, rely on the Dynamic Adaptive Streaming over HTTP (DASH) protocol to transfer data from video content servers to clients over the Internet. The DASH protocol enables high-quality video content delivery, adapting to changing network conditions via end-to-end bitrate adjustment.

Recent research shows that when multiple DASH clients stream Internet videos simultaneously via a wireless access point (AP), they may experience poor quality of experience [2], [3]. This is because co-existed DASH clients independently estimate current available bandwidth and make bitrate selections. The lack of coordination in the wireless local area network also leads to bandwidth fluctuations in each stream, especially when clients start or finish streaming. Such rapid bandwidth fluctuations usually cause cascading video bitrate selection changes at these DASH clients, resulting in drastic video quality variations, in the worst case, for example, video stalls. Although DASH has very effective end-to-end adaptation mechanisms that allow it to be flexible and robust to various Internet traffic, it is not designed to address such local bandwidth competitions due to their independent and asynchronous adaptation.

To address this problem, it is essential to resolve bandwidth competition among DASH clients in the wireless local area network. Traditional research on DASH focuses on designing adaptive bitrate selection algorithms [4] [5] [6] [7] [8] [9], but such solutions are usually based on independent end-to-end closed-loop adaptation between one client and the video content server, which does not coordinate among multiple clients. Some recent research uses different approaches to achieve better coordination and shows promising results. [10] introduces a traffic shaping algorithm that works at the content server, whereas [5] and [11] present control-theoretic designs on the client-side. In [12], authors directly modify TCP to improve fair sharing. These works require direct modification on either the server or the client software, so they cannot be easily deployed with various commercial DASH players and platforms with their own private implementations. [13] assumes that every client is aware that the network is shared and does not increase its video quality if others cannot do, and [14] demonstrates that heuristic-based traffic shaping at home network gateway can reduce unstable conditions among multi-clients. Our work takes a step further to provide a quality of experience control framework at the wireless access point for coordinating multiple DASH clients, which requires no direct modification on data traffic between servers and clients.

In our design, a controller located at the AP dynamically optimizes the quality of experience (QoE) metrics of DASH clients based on their real-time performance feedback. The objectives of the controller include but are not limited to minimize the stall duration and balance the bandwidth sharing among users. Our control design employs DASH models to predict the future bitrate and bandwidth of each client based on their real-time bitrate selection and buffer health data generated by DASH clients. With the prediction, the controller adjusts the bandwidth allocation for each user at the AP. This design also allows a network manager to specify QoE metrics and sharing policies for different users according to their needs so that they can better regulate multimedia streaming traffic in the local network without unfairly compromising bandwidth for individual users. To deal with various Internet traffic besides streaming traffic from AP users, we design a robust control algorithm to balance bandwidth sharing among clients while minimizing video stall duration under uncertain bandwidth.

Our solution has three major benefits: i) our solution is transparent to video content servers and can be integrated into the wireless local area networks without direct modifications on the data content and DASH player software on the user side. ii) As recent research suggests the possibility to predict bandwidth variations [15] and DASH parameters [16], our model predictive control algorithms have the potential to adapt to different streaming clients and applications. iii) Our design allows local network controller to accurately regulate bandwidth usage among users without dramatic interruptions of their online streaming videos.

Our control solution is implemented in real systems and our design is compared with existing AP based control design [14] and other heuristic solutions under different settings, e.g., different available bandwidths, different number of clients (DASH and non-DASH), different control parameters, etc. The experimental results show that our solution can significantly reduce average video stall duration by 57.1% and improves average video bitrate by 42.0% for all users.

The contributions of this work are summarized as follows:

- Our control framework utilizes real-time performance data from each client to dynamically mitigate bandwidth competition and optimize the quality of experience of multiple DASH clients in a local area network at the AP.
- Our multi-objective control formulation considers both QoE metrics and bandwidth sharing policies, which allows a network manager to flexibly regulate local network bandwidth and meet various application needs.
- Our robust control algorithm design optimizes streaming bandwidth sharing under uncertain Internet traffic in the local area wireless network.
- Extensive real system evaluations show that our design significantly reduces average video stall duration by 57.1% and improves the average video bitrate by 42.0% for all users when compared to state-of-the-art solutions.

The rest of the paper is organized as follows: Section II introduces the background knowledge of DASH and motivation of our research problem. Section III describes the framework of the model predictive based QoE control system. We also propose the problem formulation and a model predictive control (MPC) algorithm. The experiment results are shown in Section IV, and more discussions about our design are shown in Section V. Related works are discussed in Section VI. Main results are summarized in Section VII.

II. Background and Motivation

A. DASH Model

DASH is an adaptive bitrate streaming technology which enables users to stream videos over the Internet delivered from conventional HTTP web servers [17]. Basically, a video is split into multiple segments (chunks) with a uniform interval of playback time. Every segment (chunk) is encoded with different discrete video bitrate that determines the size and quality of video: the higher the bitrate, the better the quality and the larger the file size [18]. As shown in Figure 1, all

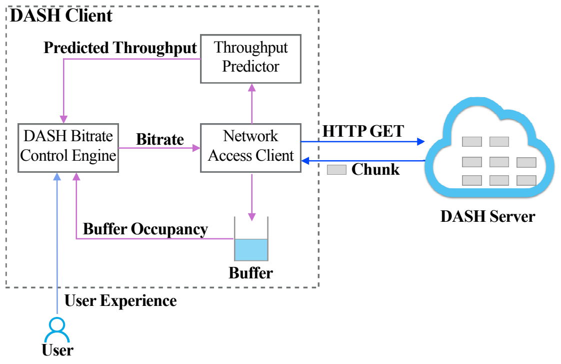

Fig. 1: Abstract model of DASH client

the video chunks and multiple copies of every chunk are stored on the DASH server. When one DASH client wants to download one chunk, it firstly sends one HTTP GET request to the DASH server, in which the bitrate of the chunk is also specified. Then the DASH player replies the corresponding copy of the chunk to the client. After receiving all the data of the current chunk, the HTTP access client sends one new request for the next chunk to the server. The DASH bitrate control engine in the client automatically and adaptively chooses the bitrate level of the next chunks to be downloaded with two inputs: buffer occupancy and predicted throughput [19] [6].

B. Motivation

We conducted an experiment to study the performances of Internet video streaming with multiple DASH clients in a wireless local area network. In this experiment, there are two users with two wireless devices and one access point (AP). The first user connected to the AP and watched a DASH video, and then the second user connected with the same AP and watched the same video 15 seconds later. Both of them used the same DASH player GPAC [20] and the default setting is provided in Section IV-A. We measured the downlink throughput, buffer length and video chunk bitrate of both DASH clients and plotted them in Figure 2. From this figure, we have the following observations: (i) the first user experienced two stalls during time interval (28,31) and (48,54) as highlighted by the red rectangle in the middle sub-figure; (ii) there exists bandwidth competition between two users as shown in the first sub-figure, which results in the stalls as highlighted by the red rectangle in the upper sub-figure; (iii) Due to the bandwidth competition, there exists unfair video quality of two users even though they use the same video player and watch the same video streaming after the 40th second.

To reduce the impact of bandwidth competition at the AP, we tested a simple load balancing solution (LBS) in the same settings. The experimental results are shown in Figure 3. From this figure, we have the following observations: (i) stalls still exist (highlighted by the red rectangles) since the allocated bandwidth doesn't match the downloading bitrate, e.g., the first stall exists when the chunk bitrate is 1850 kbps and the available bandwidth is 1100 kbps; (ii) only the existing user is affected with stalls and cascading decrease and the

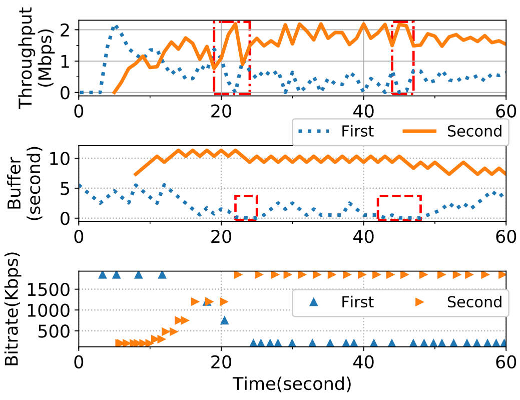

Fig. 2: Two DASH Clients Share a Wireless Link

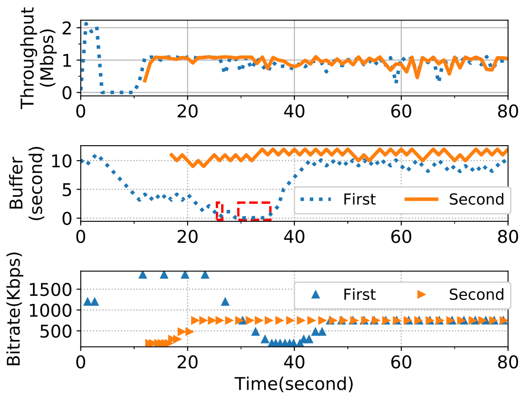

Fig. 3: Two DASH Clients with load balance solution

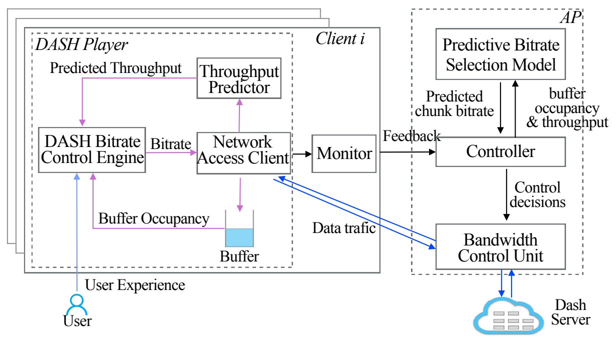

Fig. 4: System architecture overview

new user doesn't get affected. The existing client experienced stalls since its bandwidth dropped rapidly as the bandwidth share of the other client increased. The new client did not experience any stalls because the first chunk was requested at a low bitrate, and its bitrate then increased gradually to the higher ones below its bandwidth share; (iii) if bandwidth allocation can be dynamically adjusted according to real-time bitrate selection, it will reduce stall duration. For example, at the second stall, the first chunk bitrate is 1200 kbps which is larger than 1100 kbps and there is not enough content in the buffer, so stalling exists. Meanwhile, the buffer length of the second user is healthy, around 10 seconds. If more bandwidth could be allocated to the first user during the second stall, the stalling duration will be reduced, and it will not influence the overall bitrate of the second user.

Stalls happen mostly in unstable conditions such that the prediction of future throughput is inaccurate. For example, when one or more new users start streaming, and when the downlink bandwidth at the AP is varying dramatically. It is noted that this type of problems can occur in wired local area networks too.

III. QoE Control Framework at AP

A. *Framework overview*

The overview of our control system design is shown in Figure 4. The main idea is to allocate downlink bandwidth among clients simultaneously at the AP by one controller, which coordinates bandwidth adaptations of all clients to optimize their QoE in terms of stall duration and provide fair sharing of bandwidth. Here we show the feedback control loop between the AP and one representative client i. There are two controllers in our system: the first one is the DASH bitrate control engine, which is specified by DASH players, and the second one is designed for deciding the bandwidth for every client and implemented at the AP. Our design allows close interactions between two controllers: (i) Bandwidth controller affects the decisions of bitrate control engine implicitly, i.e., making bitrate control engine select bitrate level passively by adjusting downlink bandwidth. The downlink bandwidth determines the predicted throughput and buffer occupancy at DASH player, which are two keys input parameters of bitrate control engine to select video bitrate level. (ii) Bitrate control engine influences the bandwidth allocation decisions explicitly. According to the chunk size decided by the bitrate control engine, our bandwidth control should actively determine bandwidth allocation to avoid stalls and reduce the cost of unfair bandwidth sharing.

In our system, there is no modification of the users' DASH players. Therefore, to make bitrate control engine of DASH players select the chunk bitrate passively, controller at the AP should understand the selection logic behind the players' bitrate controller. One predictive bitrate selection model is proposed and embedded at the AP to estimate the chunk bitrate based on the two inputs: control decisions related buffer occupancy and throughput. A monitor program is deployed at each client to collect their real-time information as feedback to the controller, i.e., current buffer level and whether one new chunk request is sent, and the requested bitrate. This monitor collects the information of DASH players in the application layer, e.g., YouTube's IFrame API provides functions for accessing this information [21], [22], [23]. This feedback information is used to correct the prediction error of both buffer occupancy and current chunk bitrate, which is beneficial to clients for getting better QoE control. A bandwidth control unit (BCU) receives the downlink data traffic of all clients connected to the AP and then allocates the bandwidth of every client based on the control decisions.

B. *System design*

In this subsection, we first explain why there exist stalls for DASH players. Then we introduce the multi-objective control

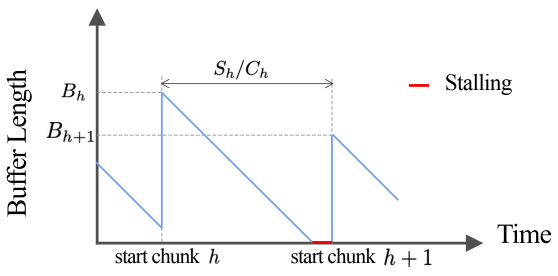

Fig. 5: Illustration of DASH player stall

problem formulation. Finally, we describe the model-based control solution, which periodically computes a solution for the control problem at the beginning of each update period based on real-time performance feedback.

1) Why there exist stalls?: Based on our introduction of DASH protocol in the previous section, video content is divided into multiple chunks and DASH player downloads chunk one by one. Therefore, stalls occur when the previous video chunk has finished playing while the next chunk has not been downloaded fully. As shown in Figure 5, at the beginning of downloading chunk h, the buffer length is B_h. The size of chunk h is denoted as S_h and C_h represents the average throughput during the download of h-th chunk. DASH player selects bitrate of the next chunk based on the historical throughput and current buffer occupancy.

However, in our network scenario, a DASH client does not know the throughput information of all the other local clients. The decrease of C_h due to bandwidth competition significantly increases downloading time S_h/C_h for the next chunk. If $S_h/C_h > B_h$, there will be a stall. Such an explanation is consistent with the observation of the stall from the 22nd to 24th second in Figure 2.

2) Multiple-objective control problem formulation: Quality of experience (QoE) of DASH video services are usually evaluated by four major metrics: video bitrate level, video bitrate variations, stalls (rebuffering) and startup delay. [6] shows that there exists a trade-off among these four metrics when the bitrate selection engine decides the bitrate level of the next chunk. It proposes one model predictive control-based bitrate selection algorithm to decide the bitrate level for future several chunks to maximize the sum of weighted four metrics. According to the experiment results in Section II, we use stall duration as the representative metric to evaluate the QoE, as stalls are the most noticeable events in video streaming.

Suppose there are N client and the total available downlink bandwidth at the AP is W. The time horizon we consider consists of M time slots and the length of one time slot is T. The control variable is formulated as $C_i[k]$ representing the bandwidth allocated to i-th client during time slot k. Let $B_i[k]$ denote the buffer occupancy of i-th client at the beginning of time slot k. $S_i[k]$ represents the data needed to be downloaded for i-th client's current chunk at the beginning of k-th time slot. Normally, different chunks of the same video can be played for the same time period, and we assume that one chunk can be decoded into one L-seconds video.

The total bandwidth allocated to clients during any time slot k should not exceed the downlink bandwidth constraint $W[k]$, and we formulate it as: $\sum_{i=1}^{N} C_i[k] \leq W[k], \quad 1 \leq k \leq M$

Stall duration of one client: We use $q_i[k]$ to denote the stall duration of i-th client during k-th time slot. It is clear that $q_i[k]$ is related to $C_i[k]$, $B_i[k]$, $S_i[k]$ and T, and we have the following analysis:

- If $B_i[k] \geq T$, $q_i[k] = 0$. If the buffer length at the start of time slot k is greater than T, there is no stall.
- If $B_i[k] < T$ and $C_i[k] \cdot B_i[k] \geq S_i[k]$, $q_i[k] = 0$. It means if current chunk can be downloaded fully before the buffer is run out, there is also no stall.
- If $B_i[k] < T$ and $C_i[k] \cdot B_i[k] < S_i[k]$, $q_i[k] = \min\{S_i[k]/C_i[k], T\} - B_i[k]$. It means that the buffer is used up before the download of current chunk is finished.

Based on the above analysis, we conclude that:

$$q_i[k] = \Big(I(T - B_i[k]) \cdot \Big(\min\{S_i[k]/C_i[k], T\} - B_i[k] \Big) \Big)_+ \quad (1)$$

where $(x)_+ = \max\{x, 0\}$ and $I(x) = 1$ if $x > 0$, otherwise, it is 0. Therefore, the total stall duration of N clients during time horizon, M time slots is: $J_s = \sum_{i=1}^{N} \sum_{k=1}^{M} q_i[k]$.

Then we study the relation between $B_i[k]$ and $B_i[k+1]$, and we analyze it as follows:

- If $S_i[k] > T{\cdot}C_i[k]$ and $q_i[k] = 0$, $B_i[k{+}1] = B_i[k]{-}T$. It means that during k-th time slot, the download of current chunk is not finished and there is no stall.
- If $S_i[k] > T \cdot C_i[k]$ and $q_i[k] > 0$, $B_i[k+1] = 0$. If there exists stall and current chunk isn't downloaded fully, the buffer level for the time slot $k+1$ is 0.
- If $S_i[k] < T \cdot C_i[k]$ and $q_i[k] = 0$, $B_i[k+1] = B_i[k] - T + L$, meaning there is no stall and the download of current chunk is finished, the buffer level increases L.
- If $S_i[k] < T \cdot C_i[k]$ and $q_i[k] > 0$, $B_i[k+1] = L - (T - B_i[k] - q_i[k]) = L - T + B_i[k] + q_i[k]$. If there exists stall, meanwhile, the chunk is downloaded fully, only $(T - S_i[k]/C_i[k])$ seconds of video is played.

We conclude that:

$$B_i[k{+}1] = \Big(B_i[k] {+} q_i[k] {-} T {+} L {\cdot} I(T {\cdot} C_i[k] {-} S_i[k]) \Big)_+ \quad (2)$$

where the initial value $B_i[1]$ $(1 \leq i \leq N)$ is updated based on the feedback from clients' player.

Finally, we model the relation between $S_i[k]$ and $S_i[k+1]$. If current chunk is not fully downloaded during time slot k, meaning $S_i[k] > C_i[k] \cdot T$, there is only remainder of data that needs to be downloaded, and we have $S_i[k{+}1] = S_i[k]{-}C_i[k]{\cdot}T$. Otherwise, one new chunk is requested. Let $P_i[k]$ be the bitrate of one chunk if it is requested by i-th client during time slot k. Under such condition, the relation is:

$$\begin{aligned} S_i[k+1] &= P_i[k] \cdot L - (T - S_i[k]/C_i[k]) \cdot C_i[k] \\ &= S_i[k] - C_i[k] \cdot T + P_i[k] \cdot L \end{aligned} \quad (3)$$

According to the above discussion, the relation between $S_i[k]$ and $S_i[k+1]$ is concluded as:

$$S_i[k+1] = S_i[k] - C_i[k] \cdot T + P_i[k] \cdot L \cdot I(T \cdot C_i[k] - S_i[k]) \quad (4)$$

where $S_i[1]$ $(1 \leq i \leq N)$ is updated according to the feedback from the clients, such as when one chunk request is sent and what the bitrate is. It is noted that $P_i[k]$ is decided by the DASH player. According to background introduction in Section II, a player chooses the bitrate of the next chunk with two inputs: historical throughput and buffer level. Here, $P_i[k]$ is estimated by one model, which predicts the player's bitrate selection based on the two inputs, and we describe the model in the following part in detail.

Cost of unfair sharing of bandwidth: Note here we consider fair sharing of bandwidth among clients as a basic objective. We could extend our framework to support other sharing policies. Let $p_i[k]$ denote the cost of unfair sharing of bandwidth for i-th client during time slot k, and it is defined as:

$$p_i[k] = \Big| C_i[k] - W[k]/N \Big| \quad (5)$$

To simplify the equation, let $U[k] = W[k]/N$. Therefore, the total cost of unfair sharing of bandwidth is:

$$J_u = \sum_{i=1}^{N} \sum_{k=1}^{M} p_i[k] = \sum_{i=1}^{N} \sum_{k=1}^{M} \Big| C_i[k] - U[k] \Big| \quad (6)$$

Since there exists a trade-off between two objectives as discusses in Section II, we define a weight parameter α when summing up the costs related to both objectives as follows:

$$\begin{aligned} \min_{C_i[k]} \quad & J_s + \alpha J_u = \sum_{i=1}^{N} \sum_{k=1}^{M} \Big(q_i[k] + \alpha \cdot |C_i[k] - U[k]| \Big) \quad (7) \\ \textbf{s.t.} \quad & \sum_{i=1}^{N} C_i[k] \leq W[k], \quad 1 \leq k \leq M \\ & C_i[k] \geq 0, \qquad (1), (2), (4) \end{aligned}$$

Although the above problem is a non-convex and non-linear optimization problem, the state space is limited due to the discrete bitrate of each client, the limited number of clients and compaction of bandwidth via binning. Given one initial state of the system, we can get the optimal control decisions by enumeration within 1$\sim$2 minutes. In the system implementation, we first build the decisions table for a limited number of possible system states and then control bandwidth dynamically for each client via looking up the table.

3) Robust Optimization with Uncertain Network Traffic: In the previous problem formulation, we assume that all network traffic is for video streaming applications. However, other light network traffic, e.g., reading news, searching and online shopping, etc, may coexist with video streaming traffic through the same access point. Due to such uncertain network traffic, we do not have perfect knowledge of available bandwidth for video streaming in the local wireless network. Hence, we discuss a formulation of the robust bandwidth allocation problem for video streaming with uncertain network traffic. It is noted that the bandwidth controller at the AP may allocate fixed bandwidth for heavy network traffic, e.g., FTP, BitTorrent and teleconference to avoid over-consumption.

Both video and non-video traffic will go through the same AP simultaneously, which disturbs the bandwidth control of video streaming applications. Since the network characteristic of video and non-video streaming traffic is different [15], our solution first collects the data-trace of every client, e.g., average throughput during each time slot to differentiate video and non-video clients in the AP. Then we learn the pattern of uncertain network traffic disturbing the available bandwidth for video streaming, e.g., burst traffic from web browsing. With such uncertain traffic pattern, we can consider the effects by setting uncertainty parameters. Let $W'_l[k]$ and $W'_h[k]$ be the historical minimum and maximum bandwidth used by non-video applications, and the available bandwidth for video streaming is denoted by an inequality: $W_l[k] \leq W[k] \leq W_h[k]$ $(W_l[k] = W - W'_h[k], W_h[k] = W - W'_l[k])$.

By introducing interval uncertainty to available bandwidth, we have the following robust optimization problem:

$$\begin{aligned} \min_{C_i[k]} \max_{W[k]} \quad & J_s + \alpha J_u \quad (8) \\ = & \sum_{i=1}^{N} \sum_{k=1}^{M} \Big(q_i[k] + \alpha \cdot \big| C_i[k] - U[k] \big| \Big) \\ \textbf{s.t.} \quad & W_l[k] \leq W[k] \leq W_h[k], \ \sum_{i=1}^{N} C_i[k] \leq W[k] \end{aligned}$$

In the above robust optimization formulation, the $\max$ function denotes the worst-case performance when available bandwidth is within the range, and we try to minimize the worst-case performance by the $\min$ function to achieve a robust optimization. Since for fixed $C_i[k]$, according to [24], the maximization expression is equal to

$$\begin{aligned} \sum_{i=1}^{N} \sum_{k=1}^{M} \Big(& q_i[k] + \alpha \cdot \max\{C_i[k] - W_l[k]/N, C_i[k] - W_h[k]/N, \\ & W_l[k]/N - C_i[k], W_h[k]/N - C_i[k], \quad (9) \\ & C_i[k] - \sum_{i=1}^{N} C_i[k]/N, \sum_{i=1}^{N} C_i[k]/N - C_i[k]\} \Big) \quad (10) \end{aligned}$$

Then the optimization problem can be reformulated with slack variables $t_i[k] = |C_i[k] - U[k]|$ as:

$$\begin{aligned} \min_{C_i[k]} \quad & \sum_{i=1}^{N} \sum_{k=1}^{M} \Big(q_i[k] + \alpha \cdot t_i[k] \Big) \\ & t_i[k] \geq C_i[k] - W_l[k]/N, t_i[k] \geq C_i[k] - W_h[k]/N, \\ & t_i[k] \geq W_l[k]/N - C_i[k], t_i[k] \geq W_h[k]/N - C_i[k]\} \\ & t_i[k] \geq \sum_{i=1}^{N} C_i[k]/N - C_i[k], t_i[k] \geq C_i[k] - \sum_{i=1}^{N} C_i[k]/N\} \end{aligned}$$

Therefore, the robust optimization problem can be solved like the regular optimization problem with close computation cost.

4) Model Predictive QoE Control Algorithm: Firstly, we describe the predictive bitrate selection model to estimate the chunk bitrate one client may select according to two inputs: buffer occupancy and historical throughput. Next, we introduce one model predictive control (MPC) algorithm to compute the bandwidth control decisions for every client. Finally, we demonstrate how to update control parameters in our system.

Predictive Bitrate Selection Model: As explained previously, the bandwidth controller needs to understand the control logic behind the DASH adaptation bitrate selection algorithms so that it estimates the possible selection of chunk bitrate based on its control decisions. As shown in Equation (3), during the optimization time horizon, the controller update $S_i[k+1]$ based on the relation between $S_i[k]$ and $S_i[k+1]$, which is related to the bitrate selection model $P_i[k]$. $P_i[k]$ is used to estimate the bitrate of one chunk the DASH player may select so that enough bandwidth is allocated to avoid stalling or specified bandwidth is allocated to reduce the bitrate of the chunk and minimize the unfair sharing of bandwidth.

Now, different players use different bitrate adaptation algorithms, which have two inputs: estimation of network capacity and playback buffer occupancy [7]. Then we propose the following bitrate selection model of different bitrate adaptation algorithms:

$$P_i[k] = \arg\max_{u_j \in U} \Big\{ u_j \leq \Big(F_b(B_i[k]) \cdot F_r(\hat{\mathbf{R}}_i[k]) \Big) \Big\} \quad (11)$$

where U is the set of bitrates of the video, and $\hat{\mathbf{R}}_i[k]$ is one series with K inputs: $R_i[k-1]$,..., $R_i[k-K]$. $F_b(B_i[k])$ is one adjustment function based on the playback buffer occupancy. It is noted that for different bitrate adaptation algorithm, $F_b(B_i[k])$ varies. For example, for the throughput-based algorithm, $F_b(B_i[k]) = 1$ for any $B_i[k]$. Moreover, for buffer-based algorithm, $F_b(B_i[k]) < 1$ if $B_i[k]$ is less than the low-level threshold, otherwise, it is 1. $F_r(\hat{\mathbf{R}}_i[k])$ estimates the future available bandwidth based on the historical downlink throughput, and it's formulated as one linear model:

$$F_r(\hat{\mathbf{R}}_i[k]) = \beta_0 + \beta_1 R_i[k-1] + \cdots + \beta_K R_i[k-K] \quad (12)$$

We model that the chunk bitrate is the maximum level that is no more than $F_b(B_i[k]) \cdot F_r(\hat{\mathbf{R}}_i[k])$. To obtain function $F_b(\cdot)$ and $F_c(\cdot)$, we collect the data trace of DASH players in terms of buffer level, selected bitrate and throughput over time, and then determine model parameters to fit in with the data-trace.

Model predictive control algorithm: Here we propose one model predictive control (MPC) based QoE control algorithm to compute the control decisions of every client during each time slot. MPC utilizes an explicit process model to estimate the future response of a system if taking some decisions [25], [26], [27]. The benefit of MPC is that it not only optimizes the current time slot but also keeps future time slots in the account. By considering the possible buffer level, data needed to be downloaded and predictive bitrate selection results in the future time slots according to the control decisions, it optimizes the objectives in a complex dynamical system.

Algorithm 1: ***Model predictive control algorithm for real-time bandwidth control***

```
Input: Time horizon $M$ time slots; length of one time slot
    $T$; number of clients $N$; total available downlink
    bandwidth $W$; time length of video that one chunk can
    be played $L$ seconds.
Output: Control decision: $C_i$, $1 \leq i \leq N$
 1: while At the beginning of every time slot $T$, denoted as
    $k$-th time slot do
 2:   Update the historical throughput: $R_i[k-1]$; update the
      buffer level of every client: $B_i[k]$, update the data
      needed to be downloaded: $S_i[k]$;
 3:   if there exists non-video traffic then
 4:     Solve the robust optimization problem during fixed
        time horizon $M$: (8)
 5:   else
 6:     Solve the optimization problem during fixed time
        horizon $M$: (7)
 7:   end if
 8:   Send the control decision $C_i[k]$ to the bandwidth
      control unit.
 9: end while
10: return Control decision
```

Let us introduce the control algorithm of the controller in detail, whose pseudo-code is shown in Algorithm 1. The main idea is at the beginning of every time slot, denoted as time slot k, the controller updates the current buffer level of every client $B_i[k]$ and data needed to be downloaded for current chunk, $S_i[k]$ based on the feedback from clients' monitor. Then the controller detects whether there exists non-video traffic, if so, it solves the robust optimization problem (8), otherwise, it solves the problem (7) to get the optimal solution $C_i[k']$, $k \leq k' \leq k+M-1$. Finally, only the optimal solution $C_i[k]$ is forwarded to the BCU. In our model predictive control algorithm, we predict the possible adaptive bitrate selection of every client based on the allocated bandwidth during the time horizon. It means making the clients select bitrate passively based on controlling bandwidth and the feedback scheme of DASH.

Update $S_i[k]$ and $B_i[k]$: Controller updates the buffer level and data size that needed to be downloaded for the current chunk at the start of every time slot. DASH players can be classified into two categories: open source players, such as GPAC [20] and DASH-IF [28] and commercial players, such as YouTube player [29]. DASH players belonging to both of the two categories can provide the information, such as current buffer level, when one chunk is requested and what the bitrate is. Therefore, our controller can update $B_i[k]$ based on the real-time feedback from every client.

Let $I_i^n[k]$ denote whether one new chunk is requested by client i during time slot k. If so, it is 1, otherwise, it is 0. $d_i[k]$ represents the corresponding chunk (segment) size. For given bitrate selected during time slot k, the client monitor replies future chunk size $d_i[k]$ by averaging the size of previous chunks which have the same bitrate level. Then $S_i[k]$ is

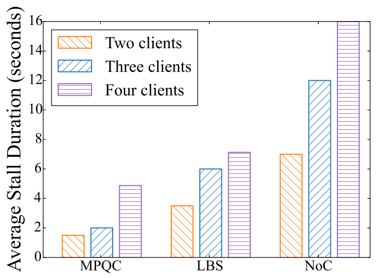

Fig. 6: Stall duration of three solutions

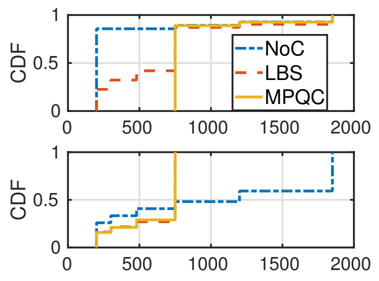

Fig. 7: CDF of video quality with two clients

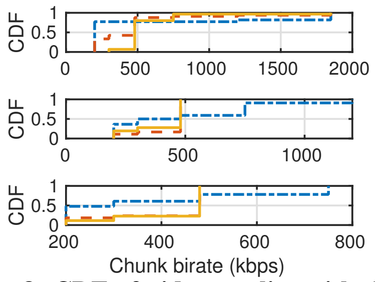

Fig. 8: CDF of video quality with three clients

	MPQC	LBS	NoC
Convergence time (second)	12	24	26
# of bitrate increase	4	6	7
# of bitrate decrease	3	5	2

TABLE I: QoE of three solutions with three clients

updated according to the following equation:

$$S_i[k] = S_i[k-1] - R_i[k-1]T + I_i^n[k-1]d_i[k-1] \quad (13)$$

IV. Evaluation

A. Implementation and experiment setup

With the consideration of computational overhead for solving the one-iteration problem, we adopt the method mentioned in [6] to implement our model predictive QoE control (MPQC). We first enumerate potential scenarios captured for each value and then compute the control decision offline, stored as one table. At the access point, the controller updates the real-time system parameters and looks up the table to get the bandwidth control decisions.

In the evaluation, the MPQC is implemented in one wireless router (Linksys WRT1900ACS) using Linux traffic control to allocate bandwidth among clients. The same player GPAC [20] is installed on the Ubuntu and macOS to watch the video streaming[1]. The buffer size of GPAC is 10 seconds and the segment length of the video is 2 seconds. The possible bitrate levels are 200, 300, 480, 750, 1200, 1850, 2850 and 4300 kbps. The experiment is conducted in the regular school building with other wireless networks. Three control strategies are evaluated:

- Model Predictive QoE Control (MPQC): this strategy is proposed in this paper.
- Load Balance Solution (LBS) [14]: multiple clients share the downlink bandwidth by one fixed ratio according to their target bitrate level. In this evaluation, clients have the same target bitrate, so bandwidth is shared among them equally.
- No Control (NoC): there is no bandwidth control at the AP. It demonstrates the baseline of the system.

The three strategies are evaluated with these metrics: (i) average stall duration: it is the average stall duration of all the clients over their playback period; (ii) video quality variations: the number of chunk bitrate increase and decrease; (iii) video quality: the bitrate of chunks; (iv) bandwidth fairness: it's defined as Equation (6); (v) convergence time: the length of time duration from joining new users to reaching stable states that there is no stall or bitrate change for all clients. We run a program in each client to collect the QoE information, i.e., video quality and buffer length.

[1]https://dash.akamaized.net/envivio/EnvivioDash3/manifest.mpd

We also test robust MPQC, MPQC, LBS and NoC under the scenario that there is uncertain web browser traffic from one client. The default settings of our evaluation are listed as follows: there are two clients and the bottleneck of the downlink is 2.2 Mbps. The update period is 0.3 seconds, the time horizon is 1.2 seconds and the value of α is 0.1.

B. Comparison of three strategies

Figure 6 plots the average stall duration of three solutions with a different number of clients. When there are two clients in the system, compared to LBS and NoC, MPQC achieves 57.1% and 78.6% less average stall duration, respectively. This is because that MPQC dynamically controls the bandwidth such that the existing client decreases its requested chunks' bitrate gradually, meanwhile it provides enough bandwidth to prevent using up the buffer before finishing downloading current chunk. It is observed with the more clients in the system, the higher average stall duration is introduced since more clients introduce fiercer competition of bandwidth among clients. Whereas, as shown in Figure 6, given the same number of clients, MPQC always outperforms LBS and NoC.

Figure 7 and 8 show the video quality of each client by three solutions under the two-client and three-client case, respectively. There are two sub-figures in Figure 7, where the upper one describes the video quality of the first client and the bottom one demonstrates that of the second client. In each subfigure, there are three curves for three comparison solutions. We can observe that for the second client, the video quality does not vary by LBS and MPQC, since its first chunk was requested at a low bitrate, and its bitrate then increased gradually to the higher ones below its bandwidth share. The first user experienced better video quality by MPQC compared with LBS and NoC, e.g., all chunks' bitrate is no less than 750 kbps, whereas, 42.0% and 83.0% of chunks' bitrate is below 750 kbps by LBS and NoC, respectively. QoE of three clients by different solutions is also shown in Table I. It can be observed that it costs 50.0% and 53.8% less time to achieve the final stable state by MPQC, compared with LBS and NoC. It is explained that the first client gradually reduces its chunks' bitrate from the higher one to the stable one due to dynamic

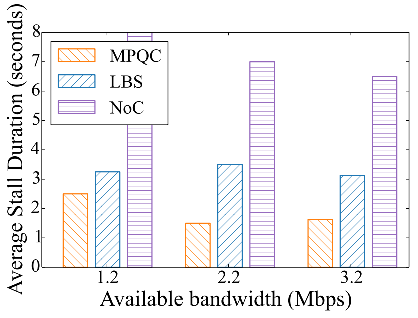

Fig. 9: Average stall duration under different available bandwidth

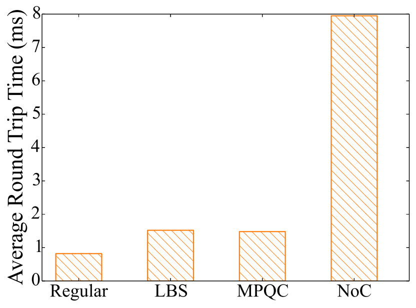

Fig. 10: Average round trip time under different cases

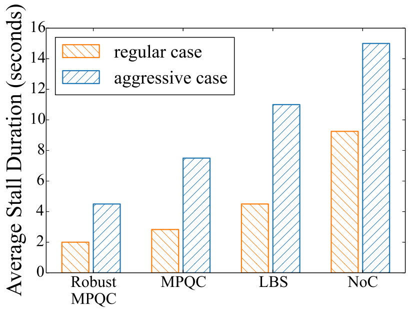

Fig. 11: Average stall duration with traffic uncertainty

bandwidth control, whereas by LBS and NoC, the first client's bitrate firstly decreases to the lowest one and then increases.

Impact of different available bandwidth: Figure 9 plots the average stall duration of three solutions with different total available bandwidth. It can be observed that given the same total available bandwidth, MPQC performs better than LBS and NoC do, e.g., when total available bandwidth is 3.2 Mbps, the average stall duration of MPQC is 48.1% and 75.0% less than that of the other two solutions respectively. When the available bandwidth changes from 1.2 to 2.2 and 3.2 Mbps, MPQC performs better since the existing client gets more bandwidth to avoid stalls while the new client does not request so much bandwidth.

Impact on Round Trip Time: We evaluate round trip time (RTT) under different solutions and show the results in Figure 10. To measure RTT, we use tcpdump to capture network packets on the client side and then calculate the time difference between sending one packet and receiving the acknowledgment packet. The left bar represents the regular case that only one client connects to the access point, which is used as the baseline. Compared to the regular case, round trip time of MPQC increases from 0.82 to 1.48 milliseconds, which is obvious that less bandwidth increases the waiting time in the queue for transmission from the access point to the client. MPQC and LBS have close average RTT since MPQC also tries to provide fair bandwidth sharing among clients. However, RTT of MPQC is 81.4% less than that of NoC, since uncontrolled bandwidth competition results in much fewer bandwidth for the client.

Impact of clients' mobility: We measure how clients' mobility affects the performance of MPQC and LBS. Here we consider two cases: (i) clients are far/near to AP; (ii) clients are fixed/moving. Due to the space limitation, we state the results without figures. The stall duration increases from 1.50 seconds to 1.67 seconds averagely when clients are 1 and 5 meters away from AP respectively. When clients are moving, the average stall duration is 1.83 seconds, with an increase of 0.33 seconds on average. The stall duration of LBS is 3.83 and 4.125 seconds when clients are far or moving respectively. It is concluded that MPQC always outperforms LBS.

C. Impact of uncertain Internet traffic

In this subsection, we measure the performance of fours solutions under the regular Internet application scenario that there exists uncertain Internet traffic from non-video applications. Figure 11 shows the performance of four solutions with non-video application traffic, where two users watch video streaming and one user uses a web browser to read the news on the Internet. There is no bandwidth control of the non-video flow, so the third user may compete bandwidth with the other two streaming clients. We consider two cases: a regular case and an aggressive case. In the regular case, the third user loads webpages with only texts and in the aggressive case, the third user loads webpages with both figures and texts. Robust MPQC achieves the best performance in both two cases since it considers the dynamic change of available bandwidth rather than one static threshold during the optimization horizon once non-video traffic is detected. Robust MPQC performs worse in the aggressive case because the bandwidth demand of uncertain network flow is out of the estimation. Table II shows the convergence time with uncertain network traffic, in which robust MPQC also converges quickly compared with the other three solutions in both two cases.

Convergence time (second)	Regular	Aggressive
Robust MPQC	17	27
MPQC	25	38
LBS	29	41
NoC	39	51

TABLE II: Convergence time with traffic uncertainty

D. Performance of MPQC with different parameters

We study how the performance of MPQC changes with different parameter settings, where the update period is 0.3 seconds and the time horizon is 1.2 seconds. In Figure 12, the value of α is changed from 0.1 to 2. There is no doubt that with the increase of α, the stall duration of MPQC increases and bandwidth fairness decreases, e.g., the average stall duration increases 14.9% and bandwidth fairness decreases 87.5% when α changes from 0.1 to 2.

Figure 13 plots the performance of MPQC with different control update periods: 0.4, 0.5 and 1.0 seconds. The prediction time horizon is set to be 2.0 seconds. We can see that the short control update period increases the performance of MPQC, as it allows more frequent control decisions for updating buffer level, bitrate selection and data to be downloaded: when the update period is 0.4, it improves the performance by 28.4% and 37.5% compared with time period lengths of 0.5

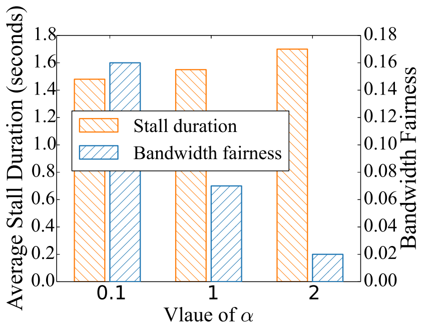

Fig. 12: Performance of MPQC with different α

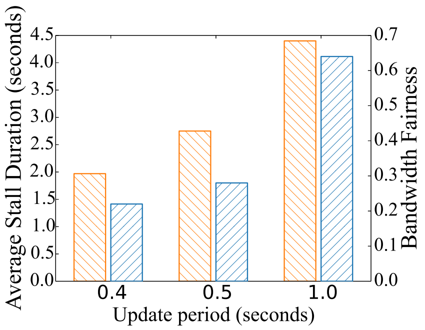

Fig. 13: Performance of MPQC with different update period

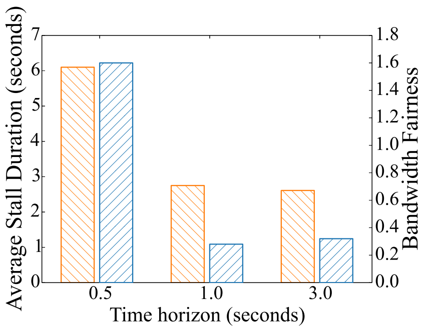

Fig. 14: Performance of MPQC with different time horizon

and 1.0 seconds respectively. Meanwhile, bandwidth fairness also increases with the growth of the update period.

Figure 14 plots the performance of MPQC with different prediction time horizon: 0.5, 1.0 and 3.0 seconds. The update period length is set to 0.5 seconds. The observation is that with the increase of future time horizon, the average stall duration and the bandwidth fairness improves 54.9% and 82.5% respectively, since considering the longer future time horizon introduces an opportunity to conduct better control.

V. Discussion

Different types of DASH clients: There are two main categories of DASH clients, open source, and private commercial players. Understanding and modeling the bitrate selection logic of different DASH clients is one key challenge of our design to allocate the bandwidth properly. In this work, we propose one general model to estimate the bitrate selection of different players. Nevertheless, to improve the accuracy of the prediction model, the specified model for different players can be proposed used, which is easy to be learned for open source players, but not for private commercial players, such as YouTube, Netflix, and Hulu. Studying the black-box bitrate selection model is one future research direction.

The design of DASH adaptation algorithms forms a rich literature [5] [6] [7] [8] and they aim to maximize one user's QoE via designing novel bitrate selection algorithms. However, [4] has already shown that there exist unfairness and instability among multiple commercial players due to uncertain bandwidth competition. Our solution provides the opportunity to solve the problem due to the bandwidth competition among multiple DASH players with a centralized method to coordinate them at the AP.

Concern of clients' privacy: our design does not increase the risk of leaking clients' privacy, such as which player one client is using and what video one client is watching. This privacy information is protected by the application's encryption mechanism and our design does not require any knowledge of such information to realize the objectives.

User priority: Considering user priority is a future direction, e.g., some users have higher priority over normal users due to their payment for on-demand service. Whereas, our work also provides some useful insights for priority-based bandwidth allocation, e.g., for a group of users paying for the same service plan with the same priority.

VI. Related Work

We classify the related work into the following categories: bandwidth control and traffic shaping, bitrate adaptation algorithm design, and buffer management and packet scheduling.

Bandwidth control and traffic shaping: A category of related work studies how to optimize the quality-of-experience (QoE) of one or multiple clients by bandwidth control or traffic shaping [30] [10] [31] [14] [32] and some of them consider the similar application scenario with ours [11] [4] [33] [34] [35]. [11] considers optimizing the multi-client quality-of-experience fairness problem from a control theory perspective, but it requires control of clients' bitrate selection process. [4] considers how to optimize efficiency, fairness, and stability when multiple bitrate-adaptive players share a bottleneck link. However, it solves the problem via redesigning the bitrate selection algorithms of clients, which introduces a high overhead to implement it for all the DASH players. [33] [34] [35] propose Software Defined Network (SDN) based in-network solution to dynamically allocate network resource. Whereas, such solutions either require the modification of clients' chunk requests or simply assume that clients can reach the highest bitrate level below the allocated bandwidth resource. [36] and [14] demonstrate that heuristic-based traffic shaping can reduce unstable conditions, but it has to intercept data traffic from each client at the AP.

Bitrate adaptation algorithm design: There are several pieces of work focusing on designing the bitrate adaptation algorithms to optimize the application level performance [5] [6] [7] [8]. [6] develops a formal control-theoretic model of the bitrate adaptation problem and then proposes a model predictive control algorithm which optimizes the quality of experience, such as average video quality, quality variations, rebuffering, and startup delay. However, such solutions are usually based on end-to-end closed-loop adaptation between the client and the video content server, which does not coordinate among multiple clients.

Buffer management and packet scheduling: some related work, such as [37] [38] [39] [40] concentrates on offering relative differentiated services among different service classes or different traffic types. [38] proposes a token-based scheduling scheme for wireless local area networks to provide guaranteed priority access to voice traffic and service differentiation for data traffic. However, the existing works only classify the application into different classes and provide differential service for every class. In our problem, all the video applications belong to the same service class and they cannot offer a specific service for every application.

VII. Conclusions

This paper presents our framework design of model predictive QoE control for multi-client Internet video streaming in a wireless local area network. Our control design has four main advantages over existing solutions: 1) it enables local network administrator to effectively control the quality of experience in Internet video streaming for local network clients; 2) the cross-layer design can directly plug-and-play in the network layer at the access point, which does not require explicit modification of the implementation of DASH players; 3) this solution utilizes real-time performance data from clients as feedback to better optimize the quality of experience for video clients; 4) extensive evaluation shows that our solution outperformed load balance solution, significantly reducing average video stall duration by 57.1% and improving the average video bitrate by 42.0% for all clients.

Acknowledgment

This work was funded in part by NSF CNS 1553273 (CAREER) and NSF CNS 1253506 (CAREER).

References

[1] Cisco, *VNI Global Fixed and Mobile Internet Traffic Forecasts*, 2017, https://www.cisco.com/c/en/us/solutions/service-provider/visual-networking-index-vni/index.html#~stickynav=1.

[2] S. Akhshabi, L. Anantakrishnan, A. C. Begen, and C. Dovrolis, "What happens when http adaptive streaming players compete for bandwidth?", in *NOSSDAV*. 2012, ACM.

[3] T. Huang, N. Handigol, B. Heller, N. McKeown, and R. Johari, "Confused, timid, and unstable: Picking a video streaming rate is hard", in *IMC*. 2012, ACM.

[4] J. Jiang, V. Sekar, and H. Zhang, "Improving fairness, efficiency, and stability in http-based adaptive video streaming with festive", *IEEE/ACM Transactions on Networking (ToN)*, 2014.

[5] Y. Sun, X. Yin, J. Jiang, V. Sekar, F. Lin, N. Wang, T. Liu, and B. Sinopoli, "Cs2p: Improving video bitrate selection and adaptation with data-driven throughput prediction", in *SIGCOMM*. 2016, ACM.

[6] X. Yin, A. Jindal, V. Sekar, and B. Sinopoli, "A control-theoretic approach for dynamic adaptive video streaming over http", in *SIGCOMM*. 2015, ACM.

[7] T. Huang, R. Johari, N. McKeown, M. Trunnell, and M. Watson, "A buffer-based approach to rate adaptation: Evidence from a large video streaming service", in *SIGCOMM*. 2014, ACM.

[8] G. Tian and Y. Liu, "Towards agile and smooth video adaptation in dynamic http streaming", in *CoNEXT*. 2012, ACM.

[9] Z. Akhtar, Y. Nam, R. Govindan, S. Rao, J. Chen, E. Katz-Bassett, B. Ribeiro, J. Zhan, and H. Zhang, "Oboe: auto-tuning video abr algorithms to network conditions", in *SIGCOMM*. ACM, 2018.

[10] S. Akhshabi, L. Anantakrishnan, C. Dovrolis, and A. C. Begen, "Server-based traffic shaping for stabilizing oscillating adaptive streaming players", in *NOSSDAV*. 2013, ACM.

[11] X. Yin, M. Bartulović, V. Sekar, and B. Sinopoli, "On the efficiency and fairness of multiplayer http-based adaptive video streaming", in *American Control Conference (ACC), 2017*. IEEE.

[12] P. Mehra, C. De Vleeschouwer, and A. Zakhor, "Receiver-driven bandwidth sharing for tcp and its application to video streaming", *IEEE Transactions on Multimedia*, 2005.

[13] M. Seufert, N. Wehner, and P. Casas, "A fair share for all: Tcp-inspired adaptation logic for qoe fairness among heterogeneous http adaptive video streaming clients", *IEEE Transactions on Network and Service Management*, 2019.

[14] R. Houdaille and S. Gouache, "Shaping http adaptive streams for a better user experience", in *MMSys*. 2012, ACM.

[15] A. Rao, A. Legout, Y. Lim, D. Towsley, C. Barakat, and W. Dabbous, "Network characteristics of video streaming traffic", in *CoNEXT'11*. 2011, ACM.

[16] D. Tsilimantos, T. Karagkioules, A. Nogales-Gómez, and S. Valentin, "Traffic profiling for mobile video streaming", in *ICC*. IEEE, 2017.

[17] Wikipedia, *Dynamic Adaptive Streaming over HTTP*, 2019, https://en.wikipedia.org/wiki/Dynamic_Adaptive_Streaming_over_HTTP.

[18] L. Brown, *What is video bitrate and why it matters?*, 2019, https://filmora.wondershare.com/video-editing-tips/what-is-video-bitrate.html.

[19] X. Xie, X. Zhang, S. Kumar, and L. E. Li, "pistream: Physical layer informed adaptive video streaming over lte", in *MobiCom '15*. 2015, ACM.

[20] GPAC, *GPAC Multimedia Open Source Project*, 2017, https://gpac.wp.imt.fr/.

[21] TubularInsights, *YouTube Brings Us Stats for Nerds*, 2013, http://tubularinsights.com/youtube-stats-for-nerds/.

[22] YouTube, *YouTube Player API Reference for iframe Embeds*, 2019, https://developers.google.com/youtube/iframe_api_reference.

[23] M Hammad Mazhar and Zubair Shafiq, "Real-time video quality of experience monitoring for https and quic", in *INFOCOM*. IEEE, 2018.

[24] F. Miao, S. Lin, S. Munir, J. A. Stankovic, H. Huang, D. Zhang, T. He, and G. J. Pappas, "Taxi dispatch with real-time sensing data in metropolitan areas: A receding horizon control approach", in *ICCPS*. 2015, ACM.

[25] James Blake Rawlings and David Q Mayne, *Model predictive control: Theory and design*, Nob Hill Pub. Madison, Wisconsin, 2009.

[26] S. Qin and T. Badgwell, "A survey of industrial model predictive control technology", *Control engineering practice*, 2003.

[27] M. Morari and J. Lee, "Model predictive control: past, present and future", *Computers & Chemical Engineering*, 1999.

[28] DASH-IF, *DASH Industry Forum*, 2017, http://dashif.org/.

[29] YouTube, *YouTube HTML5 Video Player*, 2017, https://www.youtube.com/html5.

[30] X. Chen, J. Hwang, C. Wu, S. Yang, and C. Lee, "A qoe-based app layer scheduling scheme for scalable video transmissions over multi-rat systems?", in *ICC*. IEEE, 2015.

[31] R. Rejaie and J. Kangasharju, "Mocha: A quality adaptive multimedia proxy cache for internet streaming", in *NOSSDAV*. ACM, 2001.

[32] A. El Essaili, D. Schroeder, E. Steinbach, D. Staehle, and M. Shehada, "Qoe-based traffic and resource management for adaptive http video delivery in lte", *IEEE Transactions on Circuits and Systems for Video Technology*, 2015.

[33] P. Georgopoulos, Y. Elkhatib, M. Broadbent, M. Mu, and N. Race, "Towards network-wide qoe fairness using openflow-assisted adaptive video streaming", in *Proceedings of the 2013 ACM SIGCOMM Workshop on Future Human-centric Multimedia Networking*. 2013, FhMN '13, ACM.

[34] M. Taha, L. Garcia, J. M Jimenez, and J. Lloret, "Sdn-based throughput allocation in wireless networks for heterogeneous adaptive video streaming applications", in *Wireless Communications and Mobile Computing Conference (IWCMC)*. IEEE, 2017.

[35] G. Cofano, L. De Cicco, T. Zinner, A. Nguyen-Ngoc, P. Tran-Gia, and S. Mascolo, "Design and experimental evaluation of network-assisted strategies for http adaptive streaming", in *MMSys*. ACM, 2016.

[36] A. Mansy, M. Fayed, and M. Ammar, "Network-layer fairness for adaptive video streams", in *IFIP Networking Conference (IFIP Networking), 2015*. IEEE, 2015.

[37] C. Dovrolis and P. Ramanathan, "A case for relative differentiated services and the proportional differentiation model", *IEEE Network*, 1999.

[38] P. Wang and W. Zhuang, "A token-based scheduling scheme for wlans supporting voice/data traffic and its performance analysis", *IEEE Transactions on Wireless Communications*, May 2008.

[39] G. Panza, S. Grilli, E. Piri, and J. Vehkaper, "Qos provisioning by cross-layer feedback control", in *IEEE 21st Symposium on Communications and Vehicular Technology in the Benelux (SCVT)*, Nov 2014.

[40] S. Wittevrongel, S. De Vuyst, C. Sys, and H. Bruneel, "A reservation-based scheduling mechanism for fair qos provisioning in packet-based networks", in *2014 26th International Teletraffic Congress (ITC)*. IEEE, 2014.