Buffer-based Smooth Rate Adaptation for Dynamic HTTP Streaming

Chao Zhou*, Chia-Wen Lin†, Xinggong Zhang‡, Zongming Guo*
*Institute of Computer Science & Technology, Peking University, Beijing, China
E-mail:{zhouchaoyf, zhangxinggong, guozongming}@pku.edu.cn
†Department of Electrical Engineering and the Institute of Communications Engineering,
National Tsing Hua University, Hsinchu, Taiwan
E-mail: cwlin@ee.nthu.edu.tw
‡Corresponding author. Institute of Computer Science & Technology, Peking University, Beijing, China
E-mail:zhangxinggong@pku.edu.cn

***Abstract*—Recently, *Dynamic Streaming over HTTP* (DASH) has been widely deployed in the Internet. However, it is still a challenge to play back video smoothly with high quality in the time-varying Internet. In this paper, we propose a buffer based rate adaptation scheme, which is able to smooth bandwidth variations and provide a continuous video playback. Through analysis, we show that simply preventing buffer underflow/overflow in the greedy rate adaptation method may incur serious rate oscillations, which is poor quality-of-experience for users. To improve it, we present a novel control-theoretic approach to control buffering size and rate adaptation. We modify the buffered video time model by adding two thresholds: an overflow threshold and an underflow threshold, to filter the effect of short-term network bandwidth variations while keeping playback smooth. However, the modified rate adaptation system is nonlinear. By choosing operating point properly, we linearize the rate control system. By a Proportional-Derivative (*PD*) controller, we are able to adapt video rate with high responsiveness and stability. We carefully design the parameters for the *PD* controller. Moreover, we show that reserving a small positive/negative bandwidth margin can greatly decrease the opportunities of buffer underflow/overflow incurred by the bandwidth prediction error. At last, we demonstrate that our proposed control-theoretic approach are highly efficient through real network trace.**

I. Introduction

In recent years, *Dynamic Adaptive Streaming over HTTP* (DASH) has been widely adopted for providing uninterrupted video streaming service to users with dynamic network conditions and heterogeneous devices [1]–[3]. Contrary to the past RTP/UDP, the use of HTTP over TCP is easy to configure and in particular, it greatly simplifies the traversal of firewalls and network address translators (NAT). Besides, it is cheap to be deployed since it employs standard HTTP servers and it also can be easily deployed within Content Delivery Networks (CDN). In DASH, a video content is encoded into multiple versions at different rates. Each encoded video is further divided into small video segments, each of which normally contains seconds or tens of seconds worth of video. Using HTTP protocol, a client downloads video segments sequentially by sending HTTP "GET" requests to a server. Upon network condition changes, a client dynamically switches video version for the segments to be downloaded. Dynamic HTTP streaming is preferred by more and more content providers, including Microsoft smooth streaming [4], Apple HTTP live steaming [5], Adobe HTTP dynamic streaming [6], Netflix [7], [8], etc.

Different from the traditional video streaming algorithms, DASH does not directly control the video transmission rate. Transmission rate of a segment is totally controlled by TCP protocol, which reacts to network bandwidth along the server-client path. Intuitively, if bandwidth is high, the client can choose a video with higher rate to give user better video quality; otherwise, it should switch to a low video rate to avoid playback freezes. To maximally utilize bandwidth and avoid video freezes, video rate adaptation should be responsive to network conditions. On the other hand, TCP congestion control incurs inherent rate fluctuations, and cross-traffic has both long-term and short-term variations. Adapting video rate to short-term TCP throughput fluctuations will significantly degrade user video experience. It is therefore desirable to adapt video rate smoothly.

In this paper, we present a novel control-theoretic approach to switch video rates in dynamic adaptive HTTP streaming. Our approach use the client-side buffered video time as feedback signal. Through analysis, we find that though traditional buffer based rate adaptation method by simply preventing buffer underflow/overflow can ensure continuous video playback, it may cause video rate oscillation. To address this problem, we propose a control-theoretic approach to select the best bitrate considering both the heterogeneous bandwidth and feedback buffered video time. We use two thresholds as the operating points to filter the effect of bandwidth variations on rate adaptation. By theoretical analysis, we model and linearize the rate control logic. A Proportional-Derivative (PD) controller is further involved to improve rate adaptation performance. By carefully designing the parameters for the PD controller, we balance the needs for video bitrate smoothness and bandwidth utilization. At last, our proposed rate adaptation approach is shown to be highly efficient and robust in fluctuant network conditions via real-trace based experiments.

The rest of the paper is organized as follows. Section II describes the related work. Bandwidth prediction is presented in Section III. In Section IV, we present the buffer based rate adaptation logic. In Section V, we propose a *PD* controller to improve the the video rate adaptation performance. We show real trace based experimental results in Section VI, and conclude the paper in Section VII.

II. Related Work

Although DASH is a relatively new application, due to its popularity, it has generated lots of research recently. More specially, dynamic rate adaptation is one of the most hot research topics since it is able to automatically throttle the video quality to match the available bandwidth, so that the user receives the video at the maximum possible quality.

The existing rate adaptation techniques can be classified into two main categories: *1*)bandwidth-based [9], and *2*)buffer-based [10], [11]. In the bandwidth-based rate adaptation technique, it switches up or down the rate with estimated network bandwidth. Most of the rate adaptation schemes adopted in some commercial vendors belong to this category. However, the inherent time-varying bandwidth would lead to short-term rate oscillation and deteriorate user experience of streaming services. This is demonstrated by experimental evaluation [12]. In [12], the authors compared rate adaptation of three popular DASH clients: Netflix client, Microsoft Smooth Streaming [4], and Adobe OSMF. They concluded that none of them is good enough. They are either too aggressive or too conservative. Some clients even just jump between the highest video rate and the lowest video rate. Also, all of them have relatively long response time under network congestion level shift. Besides, it is still challenging to accurately predict network bandwidth due to the complex network conditions, and playback freezes and buffer overflow may be caused by bandwidth prediction error.

For buffer-based rate adaptation technique, the rate selection decision is made to provide a continuous video playback through preventing buffer underflow/overflow. There are very few work about buffer-based rate adaptation technique. Akamai [10], [11] adopt buffer-based rate adaptation technique, but it adjusts the video rate at the server side. This makes it has limitation in supporting the large-scale multimedia delivery since it will dramatically increase the burden on the web server or cache. Moreover, though preventing buffer underflow/overflow can ensure video playback continually, it may cause dramatic video rate changes and lead to inferior user quality-of-experience [13].

III. Bandwidth Prediction

ARMA/GARCH model can be used for exact prediction applications [14]. In this paper, we use ARMA model [14] for conditional mean (expectation conditioned on the history) prediction and the GARCH model [15] for conditional variance prediction of the network bandwidth. We briefly describe these statistical models here. Interested readers are referred to [14], [15] for details.

The ARMA model is a tool for understanding and, perhaps, predicting future values in time series. The model consists of two parts, an autoregressive (AR) part and a moving average (MA) part. The notation ARMA(r, m) refers to the model with r autoregressive terms and m moving-average terms. Let $\mu(k)$ as the average bandwidth when downloading segment k, we have

$$\hat{\mu}(k) = \mu_0 + \varepsilon_k + \sum_{i=1}^{r} \phi_i \mu(k-i) + \sum_{i=1}^{m} \theta_i \varepsilon_{k-i} \tag{1}$$

where $\hat{\mu}(k)$ is the predicted value of $\mu(k)$, μ_0 is a constant, ε_k is white noise, ϕ_i is the parameters of the MA model and θ_i is the parameter of the AR model. Both ϕ_i and θ_i are derived by training.

On the other hand, it is impossible to completely eliminate the prediction errors. Therefore, we propose to use GARCH model to predict the conditional variance of $\mu(k)$. Applying GARCH(g, h) model (where g is the order of the GARCH terms and h is the order of the ARCH terms), the variance is written as

$$\hat{\sigma}^2(k) = \gamma_0 + \sum_{i=1}^{h} \gamma_i^2 \varepsilon_{k-i}^2 + \sum_{i=1}^{g} \lambda_i^2 \sigma^2(k-i) \tag{2}$$

where $\hat{\sigma}(k)$ is the predicted value of $\sigma(k)$, γ_i and λ_i are the model parameters also derived by training.

Now, we have obtained the average bandwidth and its variance for segment k. In section V, we propose to compensate the average bandwidth prediction error by its variance and propose to reserve a bandwidth margin for conservative rate adaptation.

IV. Overview of Buffer based Rate adaptation

A. Buffered Video Time Model

To sustain continuous playback, a video streaming client normally maintains a video buffer to absorb temporary mismatch between video download rate and video playback rate. In conventional single-version video streaming, video buffer is measured by the size of buffered video, which can be easily mapped into buffered video playback time when divided by the average video playback rate. In DASH, different video versions have different video playback rates. Since a video buffer contains segments from different versions, there is no longer direct mapping between buffered video size and buffered video time. To deal with multiple video versions, we use buffered video time to directly measure the length of video playback buffer.

Buffered video time process, denoted as $q(t)$, can be modeled as a single-server queue with constant service rate of 1, i.e., with continuous playback, in each unit of time, a piece of video with unit playback time is played and dequeued from the buffer. The enqueue process is driven by the video download rate and the downloaded video version. Specifically, for a video content, there are L different versions, with different playback rates $V_1 < V_2 < \cdots < V_L$. The set of these different versions is denoted as $\mathcal{V}$. All versions of the video are partitioned into segments, each of which has the same playback time of Δ. Without loss of generality, a client starts to download segment k at time instant $t_k^{(s)}$. Let $t_k^{(e)}$ be the time instant when segment k is downloaded completely, then we have

$$\int_{t_k^{(s)}}^{t_k^{(e)}} R\left(t\right)dt{=}v\left(k\right)\Delta \tag{3}$$

where $R(t)$ is the bandwidth at time t, $v(k)$ is the video rate for segment and $v(k) \in \mathcal{V}$. We denote $\tilde{R}\left(k\right)$ as average bandwidth when downloading segment k, from Eq.(3), we derive the time needed to downloaded segment k as

$$t_k^{(e)}{-}t_k^{(s)}{=}\frac{v\left(k\right)\Delta}{\tilde{R}\left(k\right)} \tag{4}$$

When a segment is completely downloaded, enqueue time is the duration of a segment Δ, dequeued time is the time consumed to downloaded the segment $t_k^{(e)}{-}t_k^{(s)}$. Then we have

$$\begin{aligned} q\left(t_k^{(e)}\right) &{=}q\left(t_k^{(s)}\right)+\Delta{-}\left(t_k^{(e)}{-}t_k^{(s)}\right) \\ &{=}q\left(t_k^{(s)}\right)+\Delta\left(1{-}\frac{v\left(k\right)}{\tilde{R}\left(k\right)}\right) \end{aligned} \tag{5}$$

and the rate of change for buffered video time, i.e. the first derivative of $q(t)$ is written as

$$q'\left(t\right)=\frac{q\left(t_k^{(e)}\right)-q\left(t_k^{(s)}\right)}{t_k^{(e)}{-}t_k^{(s)}}{=}\frac{\tilde{R}\left(k\right)}{v\left(k\right)}{-}1{,}t\in\left(t_k^{(s)}{,}t_k^{(e)}\right] \tag{6}$$

B. Underflow and Overflow Control

From Eq.(5), if the rare of the requested video version is higher than $\tilde{R}\left(k\right)$, the buffered video time decreases, and video playback freezes whenever $q\left(t_k^{(e)}\right)$ may go down to zero; if the requested video rate is lower than $\tilde{R}\left(k\right)$, the buffered video time ramps up and buffer overflow may happen, it suggests that user gets stuck at low video rate even though his connection supports higher rate. A responsive video rate adaptation scheme should control video rate that no video playback freeze and buffer overflow happens.

In order to ensure no video freeze happens, we must select a video version guaranteeing that $q\left(t_k^{(e)}\right) \geq 0$, i.e.

$$v\left(k\right) \leq \tilde{R}\left(k\right){+}\frac{\tilde{R}\left(k\right)}{\Delta}q\left(t_k^{(s)}\right) \tag{7}$$

Eq.(7) indicates an upper bound of $v(k)$ to prevent buffer underflow and the upper bound is denoted as $v^{(u)}\left(k\right)$.

On the other hand, in order to ensure no buffer overflow happens, we must have $q\left(t_k^{(e)}\right) \leq S$, i.e.

$$v\left(k\right) \geq \tilde{R}\left(k\right)+\frac{\tilde{R}\left(k\right)}{\Delta}\left(q\left(t_k^{(s)}\right){-}S\right) \tag{8}$$

where S is the buffer size. Eq.(8) indicates an lower bound of $v(k)$ to prevent buffer underflow and the lower bound is denoted as $v^{(l)}\left(k\right)$.

C. Greedy Video Version Selection

It is therefore in order to ensure continuous video playback, the selected video rate $v(k)$ must satisfy the constraints in inequality (7) and (8) at the same time. On the other hand, a high video rate is preferred to ensure high bandwidth utilization. However, if the lower bound is higher than the highest video version V_L or the upper bound is lower than lowest video version V_1, there is no video version in $\mathcal{V}$ satisfying the constraints in inequality (7) and (8). Considering these extreme cases, the rate decision can be made as follows:

$$\begin{aligned} & v\left(k\right) = \arg\max V_i \\ \text{s.t.}\ & V_i \in \mathcal{V} \\ & V_i \geq \max\left(v^{(l)}\left(k\right){,}V_1\right) \\ & V_i \leq \min\left(v^{(u)}\left(k\right){,}V_L\right) \end{aligned} \tag{9}$$

To maximally utilize bandwidth and ensure continuous video playback, the the maximal discrete video rate satisfying the constraints in (9) is selected. However, this greedy video rate adaptation method has some drawbacks:

- Bandwidth prediction is still challenging and prediction error is inevitable, it is therefore the bounds described in inequality (7) and (8) may not be accurate enough to guarantee buffer underflow/overflow not happens.
- The inherent time-varying bandwidth makes the buffered video size fluctuant. This further affects the bounds and leads to video rate oscillation.
- There may be no video version in $\mathcal{V}$ satisfying the constraints in inequality (7) and (8), the heuristic method employed in (9) may still cause buffer underflow/overflow.

V. Control-Theoretic Approach for Rate adaptation

To address the problem proposed in section IV-C, we first modify the buffer model by adding two buffered video time thresholds, $q_{\min}$ and $q_{\max}$ as the operating points to filter the effect of buffer oscillations on rate decision. Then, we analyze this rate adaptation system and a *PD* controller is proposed to stable the rate adaptation system and improve its performance. Moreover, we propose to adjust the thresholds dynamically and a small positive/negative bandwidth margin is reserved to compensate the bandwidth prediction error. We briefly summariz the rate control logic as follows:

- If $q\left(t_k^{(s)}\right) < q_{\min}$, $q_{\min}$ is used as the operating point to select a video version whose aim is to drag current buffered video time to $q_{\min}$ and ensure no buffer underflow happens.
- If $q\left(t_k^{(s)}\right) > q_{\max}$, $q_{\max}$ is used as the operating point to select a video version with the aim to drag current buffered video time to $q_{\max}$ and ensure no buffer overflow happens.
- If $q_{\min} \leq q\left(t_k^{(s)}\right) \leq q_{\max}$, the video version has the same rate with that of last requested video version. This

avoids video rate oscillation and a smooth video playback is provided.

- When the bandwidth is too low, a rate reset mechanism is adopted to set the video rate to the lowest video version. On the other hand, when the bandwidth is too high, a sleeping mechanism is proposed to consume some buffered video before downloading next segment and no buffer overflow happens.

A. *Rate Control Model and Linearization*

To control video rate oscillation and buffer underflow or overflow, we set two buffered video time thresholds, $q_{\min}$ and $q_{\max}$, to filter the effect of buffer oscillations on video rate decision. More specifically, if $q_{\min} \leq q\left(t_k^{(s)}\right) \leq q_{\max}$, the video version having the same rate with that of last request video version is selected, i.e.

$$v\left(k\right) = v\left(k-1\right) \tag{10}$$

On the other hand, when $q\left(t_k^{(s)}\right) < q_{\min}$, $q_{\min}$ is used as the operating point to selected an appropriate video version to drag current buffered video time $q(t)$ to $q_{\min}$. To ensure no buffer underflow happens, we must have $q\left(t_k^{(e)}\right) \geq q_{\min}$, therefore

$$v\left(k\right) \leq \tilde{R}\left(k\right) + \frac{\tilde{R}\left(k\right)}{\Delta}\left(q\left(t_k^{(s)}\right) - q_{\min}\right) \tag{11}$$

In this case, we prefer to select the highest video version satisfying above inequality to make full use of the bandwidth while guaranteeing no buffer underflow happens.

Similarly, when $q\left(t_k^{(s)}\right) > q_{\max}$, we have

$$v\left(k\right) \geq \tilde{R}\left(k\right) + \frac{\tilde{R}\left(k\right)}{\Delta}\left(q\left(t_k^{(s)}\right) - q_{\max}\right) \tag{12}$$

However, we prefer to select the lowest video version satisfying above inequality in this case. The main reason is that any version satisfying above inequality decrease the buffered video time and guarantees no buffer overflow happens. The buffered video time approaches to $q_{\min}$ quickly when a too high video version is selected, and it causes rate oscillation.

Since the rate of requested video version is set the same with last downloaded segment when $q_{\min} \leq q\left(t_k^{(s)}\right) \leq q_{\max}$, we mainly focus on the rate adaptation in the cases that when $q\left(t_k^{(s)}\right) < q_{\min}$ or $q\left(t_k^{(s)}\right) > q_{\max}$ in the following of this paper. Now, the rate control model can be represented as

$$\begin{cases} v\left(k\right) = \tilde{R}\left(k\right) + \frac{\tilde{R}(k)}{\Delta}\left(q\left(t\right) - q_0\right) \\ q'\left(t\right) = \frac{\tilde{R}(k)}{v(k)} - 1 \\ q_0 = \begin{cases} q_{\min}, & \text{if } q\left(t\right) < q_{\min} \\ q_{\max}, & \text{if } q\left(t\right) > q_{\max} \end{cases} \end{cases} \tag{13}$$

The block diagram of the model described in (13) is shown in Fig.1, where $Q\left(\cdot\right)$ and $R\left(\cdot\right)$ are quantization and reciprocal operations respectively. It is is a nonlinear system, which is not suitable for analysis and implementation.

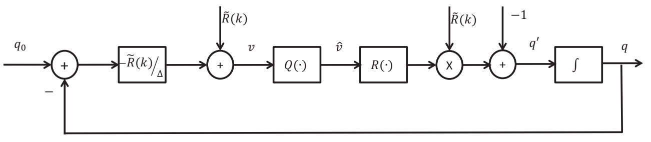


Fig. 1. Block diagram of the buffer based rate adaptation model.

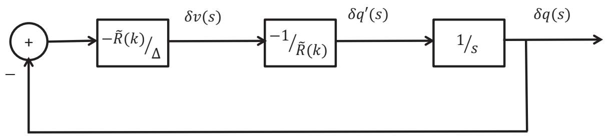


Fig. 2. Block diagram of the linearized model.

We next linearize the model described in (13) around the operating point q_0 satisfying $q'\left(t\right) = 0$. Let the corresponding video rate be v_0, then we have

$$q'\left(t\right) = 0 \Rightarrow v_0 = \tilde{R}\left(k\right) \tag{14}$$

The following linearized state and output equations are obtained for the feedback system:

$$\begin{cases} \delta\left(v\left(k\right)\right) = \frac{\tilde{R}(k)}{\Delta}\delta\left(q\left(t\right)\right) \\ \delta\left(q'\left(t\right)\right) = -\frac{1}{\tilde{R}(k)}\delta\left(v\left(k\right)\right) \end{cases} \tag{15}$$

where $\delta\left(v\left(k\right)\right) = v\left(k\right) - v_0$ and $\delta\left(q'\left(t\right)\right) = q'\left(t\right) - q'_0$. Applying the Laplace transform to the differential equations, we derive the linearized system block as shown in Fig.2. With some manipulation of the block diagram, we obtain the transfer function of the plant as

$$H\left(s\right) = \frac{1}{\Delta s + 1} \tag{16}$$

Remark 1 (Stability): The pole of the linearized system is $s_p = -\frac{1}{\Delta}$. Since $\Delta > 0$, the pole is located in the left phase plane. This indicates that the equilibrium state of the dynamics of the system is stable.

B. *PD Controller for Rate Adjustment*

Generally, a pure proportional controller would cause system oscillations and may not converge. In this subsection, we design an effective controller for the linearized feedback system in (15). We redraw the block diagram in Fig.3, where, from the control system prospective, the rate control module $G_c(s)$ is the controller to be designed, while the combination of the other blocks is the plant to be controlled.

We choose a proportional derivative (*PD*) controller since the proportional cycle can accelerate response time and improve system accuracy, and the derivative element can improve

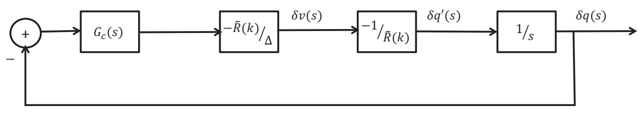


Fig. 3. Block diagram of the feedback control system for rate adjustment.

system dynamic performance. The classic *PD* controller can be designed with the following transfer function:

$$G_c(s) = K_p + K_d s \tag{17}$$

This controller is implemented at the client side. It uses the difference between current buffered video time and operation point q_0, and the variation tread of buffered video time to compute the adjustment of video rate. Then the final video version is selected.

Consider that the video version can be adjusted only when a segment is completely downloaded, then in the time domain, we have the control law as Eq.(18) shows. Now, the target video rate for segment k is written in Eq.(19). Taking into account the finite discrete video rates, the actual requested video rate for segment k is $v(k) = Q(\tilde{v}(k))$ with

$$Q(\tilde{v}(k)) \triangleq \begin{cases} \max\limits_{\{V_i: V_i \le \tilde{v}(k)\}} V_i & \text{if } q\left(t_k^{(s)}\right) < q_{\min} \\ \min\limits_{\{V_i: V_i \ge \tilde{v}(k)\}} V_i & \text{if } q\left(t_k^{(s)}\right) > q_{\max} \\ v(k-1) & \text{if } q_{\min} \le q\left(t_k^{(s)}\right) < q_{\max} \end{cases} \tag{20}$$

where $Q(\cdot)$ is the quantization function.

C. Parameter Analysis

In this subsection, we analyze the parameters K_p and K_d to stabilize the close loop system described in Fig. 3. The settling time of the control system is also analyzed. The overall system transfer function is expressed by

$$H_c(s) = \frac{K_d s + K_p}{(\Delta + K_d)s + K_p} \tag{21}$$

Suppose that it takes no more than τ seconds to switch to the suitable bitrate. Now, we have the following propositions:

Proposition 1. *The feedback system in Fig.3 is stabilized by the PD controller, if the parameters satisfy the following conditions*:

$$\begin{cases} w_c \ge \frac{1}{\tau}\sqrt{\frac{\Delta + K_d}{\Delta - K_d}} \ln \frac{20\Delta}{\Delta + K_d} \\ K_p = \sqrt{\Delta^2 - K_d^2}\, w_c \\ 0 < K_d < \Delta \end{cases} \tag{22}$$

Proof. According to proposition 1, the only pole of (21) is

$$s_p = -\frac{K_p}{\Delta + K_d} < 0 \tag{23}$$

It is located in the left phase plane and the system is stable. ■

Proposition 2. *For a PD controller satisfying Proposition 1, the* 5% *step response settling time* (T_s) *of the feedback control system satisfies*: $T_s \le \tau$.

Proof. The step response of the the system in Fig.3 is

$$U_c(s) = \frac{K_d s + K_p}{(\Delta + K_d)s + K_p}\frac{1}{s} \tag{24}$$

Applying inverse Laplace transform, we have

$$u_c(t) = u(t)\left(1 - \frac{\Delta}{\Delta + K_d} e^{-\frac{K_p}{\Delta + K_d}t}\right) \tag{25}$$

From the constraints in proposition 1, the 5% step response settling time (T_s) of the feedback control system is

$$T_s = \frac{\Delta + K_d}{\sqrt{\Delta^2 - K_d^2}\, w_c} \ln \frac{20\Delta}{\Delta + K_d} \le \tau \tag{26}$$

The last inequality is derived by the first constraint in proposition 1. ■

D. Dynamic Thresholds

Since the bandwidth is time-varying, when the threshold ($q_{\min}$ and $q_{\max}$) are fixed, there may be still no video version in $\mathcal{V}$ satisfies the constraints in formula (11) and (12) and the reasons are given in section (IV-C). To address this problem, we adjust the thresholds dynamically.

1) Dynamic $q_{\min}$*:* In order to guarantee that there always exists available video version in $\mathcal{V}$ ensuring buffer not underflow, the right term of inequality (11) must be no smaller than V_1, then we have

$$q_{\min} \le \hat{q}_{\min} = q\left(t_k^{(s)}\right) + \Delta - \frac{\alpha V_1 \Delta}{\tilde{R}(k)}, (\alpha \ge 1) \tag{27}$$

where $t_k^{(s)} = t_{k-1}^{(e)}$, and parameter α is used to adjust $q_{\min}$ conservatively when $\alpha > 1$. Then $q_{\min}$ is determined as

$$q_{\min} = \min\left(\max(\hat{q}_{\min}, 0), q_{\min}^{(T)}\right) \tag{28}$$

and $q_{\min}^{(T)}$ is the maximal $q_{\min}$ with typically $q_{\min}^{(T)} = \Delta \sim 3\Delta$.

Furthermore, $\hat{q}_{\min} < 0$ denotes that bandwidth is smaller than the lowest video rate and playback freeze is inevitable. In this case, a rate reset mechanism is proposed to set the video rate to the lowest video version, i.e. $v(k) = V_1$.

2) Dynamic $q_{\max}$*:* On the other hand, in order to guarantee that there always exists available video version in $\mathcal{V}$ ensuring buffer not overflow, the right term of inequality (12) must be not bigger than V_L, therefore

$$q_{\max} \ge \hat{q}_{\max} = q\left(t_k^{(s)}\right) + \Delta - \frac{\beta V_L \Delta}{\tilde{R}(k)}, (\beta \le 1) \tag{29}$$

and parameter β is used to adjust $q_{\max}$ conservatively when we set $\beta < 1$. We denote $q_{\max}^{(T)}$ as the minimum $q_{\max}$ with typically $q_{\max}^{(T)} = S - \Delta \sim S - 3\Delta$, then we have

$$q_{\max} = \max\left(\hat{q}_{\max}, q_{\max}^{(T)}\right) \tag{30}$$

However, when $\hat{q}_{\max} > S$, buffer overflow happens even the highest video rate is selected. In order to address this problem, we propose a sleeping mechanism to idle some period to consume the buffered video data since $\hat{q}_{\max}$ is an increasing function of buffered video data $q\left(t_k^{(s)}\right)$, therefore

$$\begin{cases} t_k^{(s)} = t_{k-1}^{(e)} + \Delta_k^{(idle)} \\ q\left(t_k^{(s)}\right) = q\left(t_{k-1}^{(e)}\right) - \Delta_k^{(idle)} \end{cases} \tag{31}$$

where $\Delta_k^{(idle)}$ is the idle time. After idling $\Delta_k^{(idle)}$ seconds, we except that $\hat{q}_{\max} = q_{\max}^{(T)}$ and then we have

$$\Delta_k^{(idle)} = q\left(t_{k-1}^{(e)}\right) + \Delta - \frac{\beta V_L \Delta}{\tilde{R}(k)} - q_{\max}^{(T)} \tag{32}$$

and the existence of available video versions is guaranteed.

$$\delta\left(v\left(k\right)\right)=\frac{\tilde{R}\left(k\right)}{\Delta}K_p\left(q\left(t_k^{(s)}\right)-q_0\right)+\frac{\tilde{R}\left(k\right)}{\Delta}K_d\frac{q\left(t_{k-1}^{(e)}\right)-q\left(t_{k-1}^{(s)}\right)}{t_{k-1}^{(e)}-t_{k-1}^{(s)}} \tag{18}$$

$$\tilde{v}\left(k\right)\approx\begin{cases} v_0+\delta\left(v\left(k\right)\right) & \text{if } q\left(t_k^{(s)}\right)<q_{\min} \text{ or } q\left(t_k^{(s)}\right)>q_{\max} \\ v\left(k-1\right) & \text{if } q_{\min}\leq q\left(t_k^{(s)}\right)\leq q_{\max} \end{cases} \tag{19}$$

E. *Conservative Rate adaptation with Bandwidth Margin*

Generally, $\tilde{R}\left(k\right)$ is set to the predicted average bandwidth in previous works. However, it comes with the risk of playback freezes or buffer overflow without any bandwidth margin, since prediction error is inevitable. To address these problems, we introduce a middle client download scheme.

To avoid buffer underflow, we must select a video rate satisfying the constraint in (11). According to inequality (27), we have $q\left(t_k^{(s)}\right)-q_{\min}\geq\frac{\alpha V_1\Delta}{\tilde{R}(k)}-\Delta>-\Delta$, so the right hand of inequality (11) is decreasing as $\tilde{R}\left(k\right)$ decreases. Therefore, when $q\left(t_k^{(s)}\right)<q_{\min}$, we reserve a small negative bandwidth margin for $\tilde{R}\left(k\right)$ to avoid buffer underflow. The average bandwidth is adjusted by its conditional variance, i.e.$\tilde{R}\left(k\right)=\mu\left(k\right)-\rho\delta\left(k\right)$, where parameter $\rho>0$.

On the other hand, when $q\left(t_k^{(s)}\right)>q_{\max}$, the video rate must satisfy the the constraint in (12) to avoid buffer overflow. The lower bound is increasing as $\tilde{R}\left(k\right)$ increases. Therefore, we reserve a small positive bandwidth margin for $\tilde{R}\left(k\right)$ to avoid buffer overflow, i.e. $\tilde{R}\left(k\right)=\mu\left(k\right)+\rho\delta\left(k\right)$.

The essence of above bandwidth reserving method is to compute confidence intervals of $\tilde{R}\left(k\right)$ and $\rho=3$ typically.

VI. Performance Evaluation

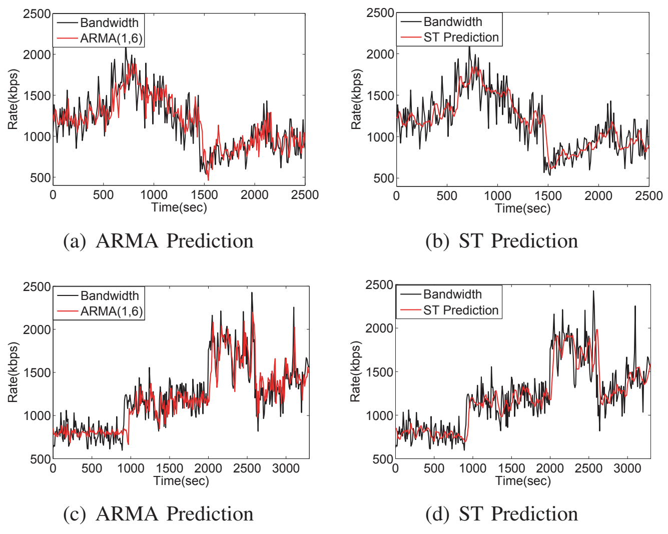
In this section, we evaluate our rate adaptation approach via real network trace. The bandwidth trace is extracted from *www.youku.com*. Considering that the background traffic is various at different times, we choose two traces, one is extracted in the daytime (*Trace I*) with length of about 3000s and the other is extracted in the night (*Trace II*) with length of about 3800 seconds. The first 500 seconds of each trace is used for training. In all our experiments, the server provides five video rate $\mathcal{V}=\{400\text{kbps},800\text{kbps},1200\text{kbps},1600\text{kbps},2000\text{kbps}\}$. The initial buffered video time is set to $\frac{S}{2}$ (40 seconds). The smoothed HTTP/TCP throughput based rate adaptation method (STRA) proposed in [9] and greedy rate adaptation (GRA) described in section IV are implemented for comparison. Table I summarizes the default parameter values.

A. *Bandwidth Prediction Accuracy*

For bandwidth prediction, we use ARMA(1,6) model for conditional mean prediction. We compare it with other history-based bandwidth prediction methods, such smoothed HTTP/TCP throughput (*ST*) [9] prediction. Fig.4 compares the accuracy of *ST* and our ARMA based bandwidth prediction.

TABLE I
Parameter Summary

Parameter	α	β	Δ	Δ_s	K_p	K_d
Default	1	1	10	10	-0.03	0.03
Parameter	$q_{\min}^{(T)}$	$q_{\max}^{(T)}$	S	$q_{\min}$	$q_{\max}$	ρ
Default	10	70	80	10	70	3

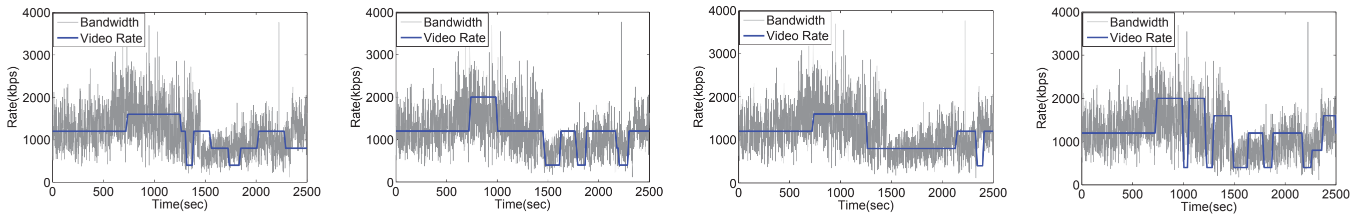


(a) ARMA Prediction (b) ST Prediction (c) ARMA Prediction (d) ST Prediction

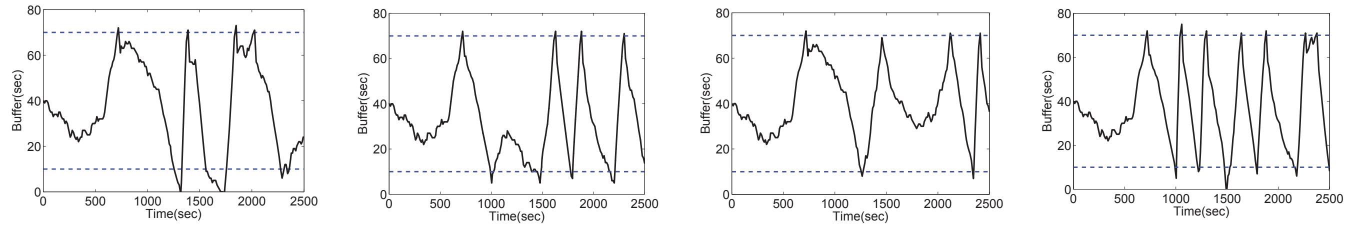
Fig. 4. Bandwidth Prediction Accuracy : (a)-(b):Trace I; (c)-(d):Trace II

We can see that our ARMA based bandwidth prediction is obviously better than *ST* prediction. Especially for some bandwidth peaks, ARMA based prediction matches well with practical bandwidth while ST prediction has a relative very poor performance. This is mainly because ARMA can capture the characteristics of self-similarity and long range dependence which are exhibited by network bandwidth [16].

We also use GARCH(1,1) model for conditional variance prediction which is used to compute confidence intervals of the predicted average bandwidth. Fig.5 plots the $\pm 3\sigma$ intervals for both traces. The figures clearly shows that the bandwidth is confidently located in the intervals at most times. It is therefore when the bounds is used for conservative rate adaptation, the client-side buffer underflow/overflow which is caused by prediction error can be avoided. The results in the next subsection demonstrate this.

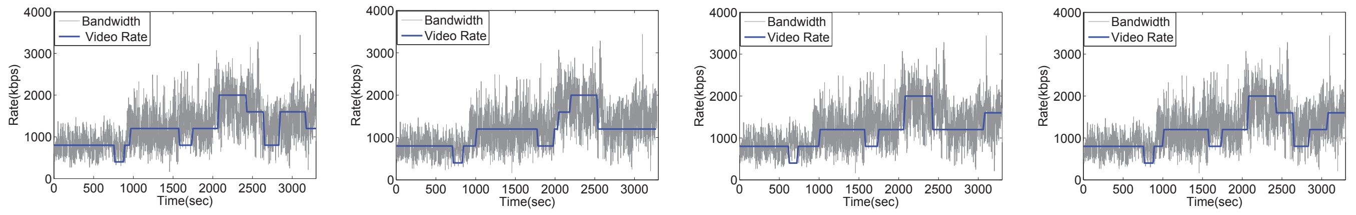


(a) $\rho = 0$ (b) $\rho = 1$ (c) $\rho = 3$ (d) $\rho = 5$

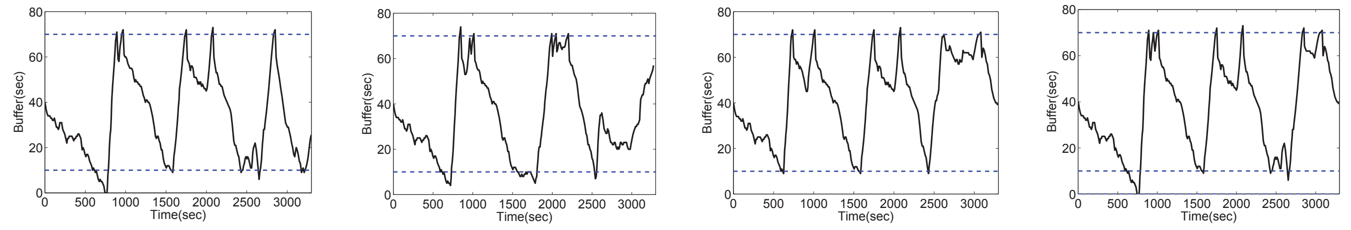


(e) $\rho = 0$ (f) $\rho = 1$ (g) $\rho = 3$ (h) $\rho = 5$

Fig. 6. Proposed Control-Theoretic Approach for Rate adaptation of *Trace I* under Different Bandwidth Margins: (a)-(d):video rate and bandwidth; (e)-(h):buffer size evolution

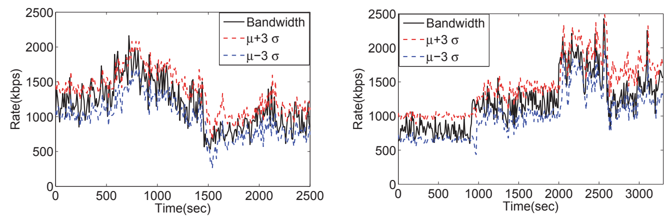


(a) $\rho = 0$ (b) $\rho = 1$ (c) $\rho = 3$ (d) $\rho = 5$



(e) $\rho = 0$ (f) $\rho = 1$ (g) $\rho = 3$ (h) $\rho = 5$

Fig. 7. Proposed Control-Theoretic Approach for Rate adaptation of *Trace II* under Different Bandwidth Margins: (a)-(d):video rate and bandwidth; (e)-(h):buffer size evolution



(a) Trace I (b) Trace II

Fig. 5. Bandwidth Margin for Prediction Error

B. Performance Evaluation under Bandwidth Margin

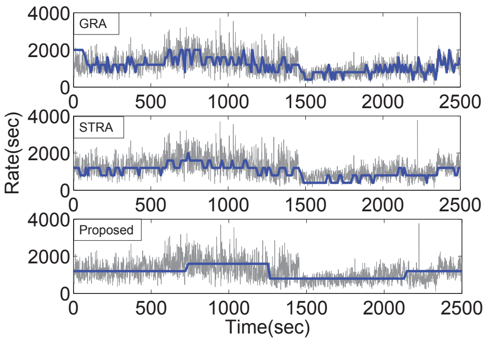
Prediction error is unavoidable. Underestimating or overestimating the bandwidth comes with the risk of playback freezes (buffer underflow) or buffer overflow. Reserving a bandwidth margin can potentially reduce these risks. Fig.6(e) and Fig.7(e) show that playback playback freeze happens two times and one time respectively when there is no bandwidth margin. As the bandwidth margin increases, playback freeze disappears demonstrated by Fig.6(e) to Fig.6(g) and Fig.7(e) to Fig.7(g). Moreover, the video rate at $\rho = 3$ is much more smooth than the rate at $\rho = 0$ and $\rho = 1$. This is because that higher ρ provides more conservative than lower ρ. However, too big bandwidth margin also may cause playback freezes and video rate fluctuations as Fig.6(d), Fig.6(h), Fig.7(d) and Fig.7(h) show. The main reason is that the gap between the adjusted bandwidth and practical bandwidth is too big, and it can not be used for effective rate adaptation. We choose $\rho = 3$ as the default value in our work, and this is coincident with the confidence interval of Gaussian distribution. Though relative smooth video rate is provided by reserving appropriate bandwidth margins, there is still some rate spikes. For example, the short-term negative rate spikes happen around $t = 2350$s in Trace I, $t = 630$s and $t = 1600$s in Trace II and these spikes can be annoying to user. In the next subsection, we show that these short-term rate spikes can be eliminated by dynamically setting buffer thresholds $q_{\min}$ and $q_{\max}$.

C. Performance Evaluation under Dynamic Thresholds

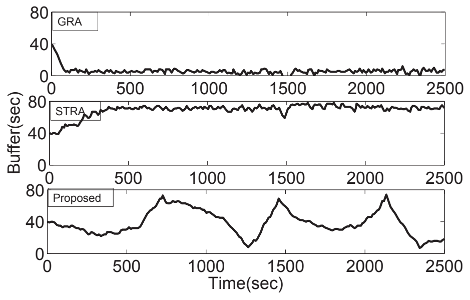
In the above subsection, $q_{\min}$ and $q_{\max}$ are fixed. However, it is not trivial to set the values of these thresholds in practice since the bandwidth is time-varying. Low $q_{\min}$ and high $q_{\max}$ increase the risk of buffer underflow/overflow, this is demonstrated in Fig.8(e) and Fig.9(e). On the other hand, when $q_{\min}$ increases and $q_{\max}$ decreases, the gap between $q_{\min}$ and $q_{\max}$ becomes small and video rate fluctuates drastically as Fig.8(c) and Fig.9(c) shows. Though when $q_{\min} = 10$ and $q_{\max} = 70$, the performances are relative better, there is still some rate spikes. From Fig.9(a) and Fig.9(b), we can clearly find that the two short-term negative rate spikes which happens around $t = 630$s and $t = 1600$s in Fig.9(b) can be eliminated by setting a lower $q_{\min}$. This also demonstrates that more smooth video rate can be provided by setting appropriate values of the thresholds at different times. In this paper, we propose to adjust the thresholds dynamically. Fig.8(d) and Fig.9(d) shows that the video rate performance under dynamic thresholds is much better than that under fixed values. The plots in Fig.8(h) and Fig.9(h) also demonstrate that playback freezes and buffer overflow never happens. More specifically, from Fig.9(h) we find that the buffer becomes very small around $t = 630$s and $t = 1600$s, but it still bigger than $q_{\min}$ which has been set to smaller values dynamically, so the video rate keeps constant and no rate spikes happen. At most times, $q_{\min}$ and $q_{\max}$ keep unchanged, this is because that they are bounded by $q^{(T)}_{\min}$ and $q^{(T)}_{\max}$ as Eq.(28) and Eq.(30) show.

D. Performance Comparison with Existing Schemes

At last, we compare the performance of our proposed control-theoretic approach with STRA and GRA. Fig.10 plots the video rate and buffer size of different schemes. For space limit, we only plot the results under *Trace I*. The figures clearly show that the video rate in our proposed approach is much more smooth than the other methods. On the other hand, no playback freeze and buffer overflow happens in our proposed approach. In STRA, the video rate is switched up/down according to the smoothed throughput. However, since bandwidth is time-varying and ST prediction is not accurate enough, the video rate fluctuate drastically. The performance loss of GRA is mainly because that it only provides a continuous video playback without considering the smoothness of video rate. Moreover, in STRA, the video rate which is closest to the predicted bandwidth and smaller than bandwidth is selected, so the buffer always stays at a high level. On the other hand, GRA selected the highest video rate under the constraint of ensuring buffer not overflow/underflow,



(a) video rate and bandwidth



(b) buffer size evolution

Fig. 10. Performance Evaluation with Existing Schemes: (a):video rate and bandwidth; (b):buffer size evolution

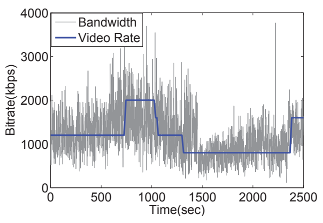
therefore its buffered video time always keeps at a low level and underflow happens at some times due to the prediction error.

VII. Conclusion

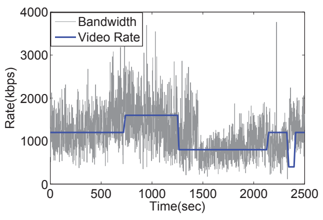
In this paper, we have proposed a novel control-theoretic approach to switch video rates in dynamic adaptive HTTP streaming. We show that though traditional buffer based rate adaptation can ensure video playback continually, it results to serious rate oscillation. To address this problem, we add an underflow threshold and an overflow threshold to filter the effect of short-term network bandwidth variations while keeping playback smooth. Then a novel control-theoretic approach is designed to switch video rates. Based on the properly chosen operating points, we linearize the rate control system and develop a *PD* controller to stabilize the video rate adaptation. Moreover, we show that the opportunities of buffer overflow/underflow incurred by bandwidth prediction error can be greatly decreased by reserving a small positive/negative bandwidth margin. At last, the real network trace based experiments demonstrate the highly effectiveness of our proposed control-theoretic approach.

Acknowledgements

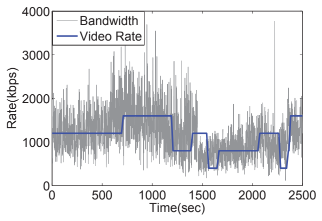
This work was supported by National High-tech Technology R&D Program (863 Program) of China under Grant 2013AA013504National Natural Science Foundation of China under contract No. 61071082 and National Key Technology R&D Program of China under Grant 2012BAH18B03.



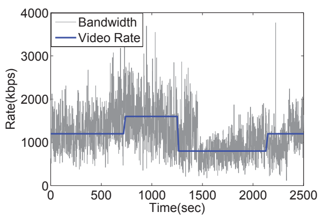
(a) $q_{\min} = 5, q_{\max} = 75$



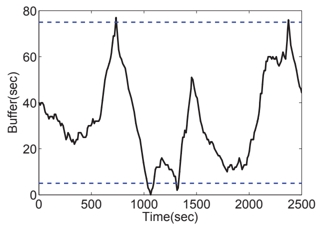
(b) $q_{\min} = 10, q_{\max} = 70$



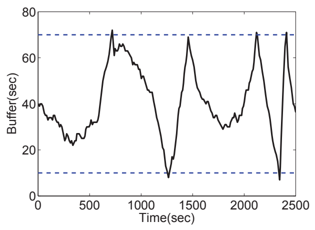
(c) $q_{\min} = 15, q_{\max} = 65$



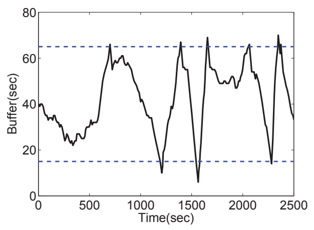
(d) Dynamic



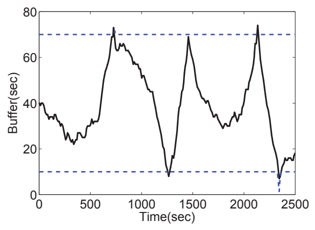
(e) $q_{\min} = 5, q_{\max} = 75$



(f) $q_{\min} = 10, q_{\max} = 70$

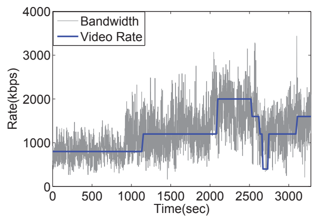


(g) $q_{\min} = 15, q_{\max} = 65$

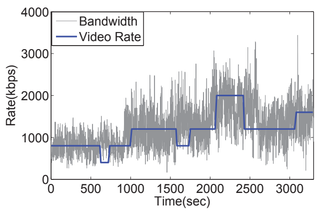


(h) Dynamic

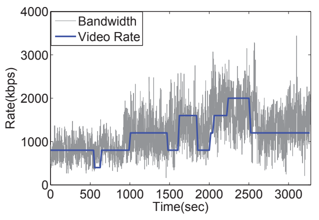
Fig. 8. Proposed Control-Theoretic Approach for Rate adaptation of *Trace I* under Different Buffer Thresholds: (a)-(d):video rate and bandwidth; (e)-(h):buffer size evolution



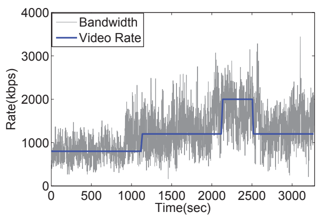
(a) $q_{\min} = 5, q_{\max} = 75$



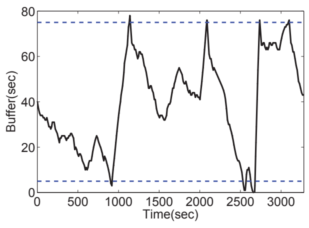
(b) $q_{\min} = 10, q_{\max} = 70$



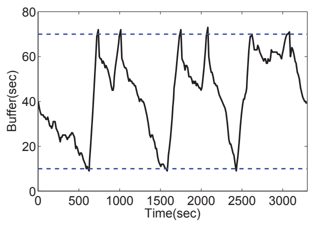
(c) $q_{\min} = 15, q_{\max} = 65$



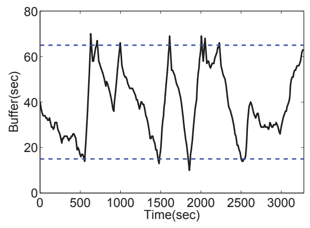
(d) Dynamic



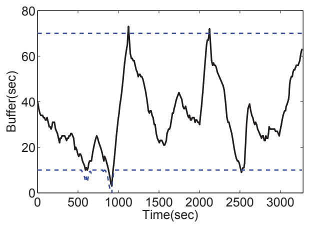
(e) $q_{\min} = 5, q_{\max} = 75$



(f) $q_{\min} = 10, q_{\max} = 70$



(g) $q_{\min} = 15, q_{\max} = 65$



(h) Dynamic

Fig. 9. Proposed Control-Theoretic Approach for Rate adaptation of *Trace II* under Different Buffer Thresholds: (a)-(d):video rate and bandwidth; (e)-(h):buffer size evolution

References

[1] A. Begen, T. Akgul, and M. Baugher, "Watching video over the web: Part 1: Streaming Protocols," *IEEE Internet Comput.*, vol. 15, no. 2, pp. 54–63, Mar. 2011.

[2] R. Kuschnig, I. Kofler, and H. H., "Evaluation of HTTP-based request-response streams for internet video streaming," *in Proc. ACM MMSys11*, pp. 245–256, Feb. 2011.

[3] T. Stockhammer, "Dynamic Adaptive Streaming over HTTP: standards and design principles," *in Proc. ACM MMSys11*, pp. 133–144, Feb. 2011.

[4] A. Zambelli, "IIS smooth streaming technical overview," *Microsoft Corporation*, 2009.

[5] R. Pantos and W. May., "HTTP Live Streaming," *IETF Draft,*, jun. 2010.

[6] D. Hassoun, "Dynamic streaming in flash media server 3.5," *http://www.adobe.com/devnet/ashmediaserver/*.

[7] V. K. Adhikari, Y. Guo, F. Hao, M. Varvello, V. Hilt, M. Steiner, and Z.-L. Zhang, "Unreeling netflix: Understanding and improving multi-CDN movie delivery," *in Proceedings of IEEE INFOCOM*, pp. 1620–1628, Mar. 2012.

[8] J. F. Kurose and K. Ross, "Computer Networking: A Top-Down Approach Featuring the Internet," *6th ed. Addison-Wesley*, 2012.

[9] C. Liu, I. Bouazizi, and M. Gabbouj, "Rate Adaptation for Adaptive HTTP Streaming," *in Proc. ACM MMSys11*, pp. 169–174, Feb. 2011.

[10] "Akamai HD Network Demo," *http://wwwns.akamai.com/hdnetwork/demo/flash*.

[11] L. De Cicco, S. Mascolo, and P. V., "Feedback Control for Adaptive Live Video Streaming," *in Proc. ACM MMSys11*, pp. 145–156, Feb. 2011.

[12] S. Akhshabi, A. C. Begen, and D. C., "An Experimental Evaluation of Rate-Adaptation Algorithms in Adaptive Streaming over HTTP," *in Proc. ACM MMSys11*, pp. 169–174, Feb. 2011.

[13] E. C. R. Mok, X. Luo, and R. Chang, "Qdash: A Qoe-aware DASH system," *in ACM Multimedia Systems*, Feb. 2012.

[14] G. E. P. Box, G. M. Jenkins, and G. C. Reinsel, *Time Series Analysis: Forecasting and Control.* third edition. Prentice-Hall, 1994.

[15] D. Niu, B. Li, and S. Zhao, "Understanding Demand Volatility in Large VoD Systems," *in Proc. the 21st International workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV)*, pp. 39–44, 2011.

[16] M. Crovella and A. Bestavros, "Self-similarity in World Wide Web traffic: evidence and possible causes," *IEEE/ACM Trans. Networking*, vol. 5, no. 6, pp. 835 –846, Dec. 1997.