Dynamic Threshold Based Rate Adaptation for HTTP Live Streaming

Lan Xie[1], Chao Zhou[1], Xinggong Zhang[1,2], Zongming Guo[1,2,*]

[1]Institute of Computer Science & Technology, Peking University, Beijing, P. R. China, 100871
[2]Cooperative Medianet Innovation Center, Shanghai, China

***Abstract*—The Dynamic Adaptive Streaming over HTTP (DASH) is specified to cope with the changing network conditions and provide an adaptive bit-rate HTTP-based streaming solution. While there have been many researches of rate adaptation algorithms on adaptive HTTP streaming, much of the work is focused on Video on Demand (VoD) service—which is not same as live streaming. It is generally preferred to minimize the end-to-end delay and make full use of the bandwidth for live services. In this paper, we propose a buffer-based rate adaptation algorithm with dynamic threshold which can decrease the rate transitions and provide a seamless playback under a low latency requirement. The rate adaptation metrics not only take into account the momentary value of bandwidth but also consider its fluctuation as the recognition of bandwidth is crucial over small buffer. Experiments demonstrate that our proposed rate adaptation scheme outperforms the methods using fixed threshold or instant throughput.**

Keywords*—*DASH, live streaming, rate adaptation algorithm, dynamic buffer threshold

I. Introduction

In recent years, Dynamic Adaptive Streaming over HTTP (DASH) has been widely used for video streaming service over the Internet [1]. In DASH, an HTTP streaming provider should generate multiple versions of an original video that contains various bit-rate or resolution. Moreover, each video version file is further partitioned into small video segments, which normally contains a few seconds of video content. In terms of live streaming, the segments are generated periodically, i.e. a new segment becoming available shortly after it has been recorded and encoded completely. A DASH client requests the available video segments according to the network condition and the playback buffer occupancy. This process is referred to as rate adaptation which is one of the most essential components to improve the streaming quality. By far, many rate adaptation schemes have been designed for VoD services, including bandwidth-based schemes [2] and buffer-based schemes [3], [4]. However, the research on rate adaptation scheme for live streaming is still limited.

To cope with throughput fluctuations in video streaming, in startup phase, a client should download some amount of video data before it can start playing. Obviously, if the amount of buffered data is large, the client can better cope with future fluctuations. However, this initial buffering delay may negatively affect the quality of experience, especially for live streaming. For on-demand streaming, a good strategy is to use a low bit-rate to achieve fast start-up, and apply a very large buffer size to accommodate bandwidth fluctuation. For live streaming, even if one tries to download low bit-rate video in order to fill up buffer quickly, the amount of buffered media is still limited because the client can only download the segments having been generated. Truong [5] has investigated typical adaptation methods in the context of live streaming under small initial buffering. The results show that the bandwidth based method brings about fluctuating bit-rate. Fixed threshold buffer based method provides smoother media but it is unsuitable when sudden drops of throughput occur frequently.

The performance of the threshold buffer based method is influenced by the value of dual-thresholds. A small underflow-threshold leads to smooth video bit-rate while increasing the risk of buffer underflow, and vice versa. In this paper, we propose a rate adaptation scheme based on dynamic threshold. It focuses on continuous playback if the bandwidth fluctuates dramatically, while a smooth quality playback can be ensured when the network condition is stable. Specifically, the contribution of this paper can be concluded as:

- We present the client buffer model for live adaptive streaming, where we find out, in live streaming, the buffer will never suffer overflow since the buffer has a upper bound which equals to the initial buffering delay.
- Using the client buffer model, we propose a buffer based rate adaptation approach with a dynamic underflow threshold. The threshold considers both the network fluctuation information, which is determined by the coefficient variance of throughput, and client buffer occupancy.
- Experiments on real throughput dataset demonstrate that our approach reduces the number of rate switches meanwhile maintains a seamless streaming which increases the quality of experience for live viewer under low latency requirement.

This paper is organized as follows. In Section II, we present the buffer model for streaming live content. In Section III, we describe the dynamic threshold buffer based rate adaptation scheme. In section IV, the performance of different adaptation methods is evaluated including fixed/dynamic threshold buffer based rate adaptation. Finally, conclusions are provided in Section V.

II. Client Buffer Model of Live Streaming

A DASH system on live streaming contains a server, clients and a content provider, where media are segmented as certain

[*] Corresponding author. E-mail:guozongming@pku.edu.cn

978-1-4673-6853-7/17/$31.00 ©2017 IEEE

duration, namely T, of data in order to deliver. The client continues receiving segments from server. As the occupancy of client buffer is generally tracked in *seconds of media*, we build a dynamic model of the playback buffer for storing live content.

During the startup phase for live streaming, the client requests q_0 seconds video started from the latest segment without playing it. At playback time t, the buffer receives data at a specific video rate, denoted as $R(t)$, meanwhile the perceived bandwidth is represented as $C(t)$. Therefore, the buffer consumes one second of media in every second but obtains $C(t)/R(t)$ seconds of video. Fig. 1 illustrates the snapshot of the dynamic model of playback buffer when the startup phase has been just completed:

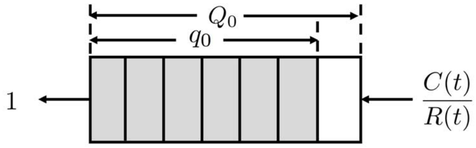


Fig. 1: the snapshot of the dynamic model of playback buffer when the startup phase has been just completed.

Here, Q_0 ($Q_0 > q_0$) is the initial buffering delay, standing for the time duration for downloading q_0 seconds video. The client then proceeds with a steady-state phase as soon as the initial buffering finishes. Thus, the buffer occupancy, denoted as $q(t)$, can be calculated as:

$$q(t) = q_0 + \int_0^t (\frac{C(t)}{R(t)} - 1)dt. \tag{1}$$

Conventionally, the client buffer risks suffering both underflow and overflow for on-demand streaming as the rate adaptation scheme may choose improper video bit-rate. However, in terms of live streaming, the client buffer will never overflow due to the live content generation constraint. As the server generates only Q_0 seconds video at the startup phase and the client will not download the media data more than the server generated, we have:

$$q_0 + \int_0^t \frac{C(t)}{R(t)} dt < Q_0 + t. \tag{2}$$

Combined with (1), $q(t) < Q_0$ always holds for every playback time t. Hence, the client buffer has an upper bound of Q_0 seconds of video under live streaming.

III. BUFFER BASED RATE ADAPTATION WITH DYNAMIC THRESHOLD

The buffer-based methods with fixed dual-threshold decide the bit-rate mainly by the buffer occupancy and the estimated bandwidth. The upper threshold, namely the overflow-threshold, intents to prevent buffer from overflow while achieving high bandwidth utilization and the lower one, namely the underflow-threshold, mainly focus on continuous playback. This approach can maintain a seamless and smooth streaming utilizing high underflow-threshold with buffer size as tens of seconds [3]. However, in live streaming, large buffer size, i.e. Q_0, will introduce long initial buffering delay, leading to dissatisfactions of users. On the other hand, the difference between the two thresholds causes dilemma: if the difference is great, interruptions may occur when the bandwidth fluctuates remarkably although smooth quality playbacks are promised; when the value is small, the risk of buffer starvation decreases while the quality may switch too frequent, harming user experiences.

Aiming at providing a low latency service while balancing the tradeoff between the quality switches and buffer starvation, we propose a buffer based rate adaptation algorithm utilizing dynamic underflow-threshold. When the buffer occupancy is lower than the dynamic underflow-threshold θ, to avoid buffer depleting and provide a continuous playback, the video bit-rate should be selected no higher than the estimated bandwidth. As is mentioned in section II, the client will never suffer from overflow. However, an upper threshold is still significant for achieving high bandwidth utilization. Obviously, high upper threshold is recommended to provide smooth quality. Thus, the upper threshold might as well be set as $Q_0 - T$. When the buffer occupancy is higher than this value, a video bit-rate higher than the estimated bandwidth can be selected. Besides, when the buffer occupancy is located between the two thresholds, the risk of buffer underflow is low, and the video bit-rate should be kept unchanged so that smooth video quality is provided.

Specifically, assume that a video content is encoded into N different video bit-rates, each of which is called as a *version*, and let $\mathcal{R}_i$ be the bit-rate of i-th level, satisfying $\mathcal{R}_i < \mathcal{R}_j, \forall i < j$. We define t_k as the start time of downloading k-th segment, then $q(t_k)$ stands for the buffer occupancy when starting to download segment k. In live streaming, it is worthy to note that if k-th segment is already available at the server, than t_k equals to the finish time of downloading segment $k-1$; otherwise, it should wait for a while until the k-th segment is generated completely, that is, $t_k = k \cdot T$. Estimating available bandwidth $\hat{c}(k)$ is beyond the scope of this paper and we simply use the average throughput of the previous 5 segments. Therefore, the video bit-rate of segment k, denoted by $r(k)$, can be calculated as:

$$r(k) = \begin{cases} \max\limits_{\{1 \le i \le N\}} \{\mathcal{R}_i | \mathcal{R}_i \le \hat{c}(k)\}, & \text{if} \quad q(t_k) < \theta \\ \min\limits_{\{1 \le i \le N\}} \{\mathcal{R}_i | \mathcal{R}_i \ge \hat{c}(k)\}, & \text{if} \quad q(t_k) > Q_0 - T \\ r(k-1), & \qquad \text{otherwise} \end{cases} \tag{3}$$

Our goal is to dynamically adjust the underflow-threshold θ, so that when the bandwidth fluctuates remarkably, it would be raised up to avoid buffer underflow and if the bandwidth is stable, θ should be turn down to maintain smooth quality. In the following sections, we analysis how to decide the threshold.

A. Dynamic Threshold

Modifying θ by every second would consume large computation resources and unnecessary. According to (3), if buffer occupancy is over the upper threshold at the time starting to download the k-th segment, i.e. $q(t_k) > Q_0 - T$, the client will request video with bit-rate higher than the estimated bandwidth, leading to greater probability of underflow. Assume the video bit-rate $r(k)$ is $\mathcal{R}_j$. Hence, we update the underflow-threshold θ in this condition.

To smooth the video quality, as defined in (3), the client will request the same bit-rate $\mathcal{R}_j$ till buffer occupancy becomes smaller than the underflow-threshold. We further suppose that the buffer occupancy will continue decreasing under quality $\mathcal{R}_j$ during a period of time in the near future, denoted by τ. In this situation, the network capacity is supposed to be lower than $\mathcal{R}_j$, namely:

$$\frac{1}{\tau}\int_{t_k}^{t_k+\tau} C(t)dt \leq \mathcal{R}_j, \tag{4}$$

Meanwhile, as the video rate stays at the same value $\mathcal{R}_j$ for τ, the buffer dynamics can be derived as:

$$q(t_k) - \theta = -\int_{t_k}^{t_k+\tau} (\frac{C(t)}{\mathcal{R}_j} - 1)dt. \tag{5}$$

We take into account the network throughput in the near future τ. Since we cannot obtain the precise throuput, we use the history network throughput, i.e. $\int_{t_k-\tau}^{t_k} C(t)dt$, to approximately calculate this value. Last, as $\theta < 0$ makes no sense, we set one segment duration as lower bound for θ to prevent buffer from underflow:

$$\theta = \max\{T, \int_{t_k}^{t_k+\tau} (\frac{C(t)}{\mathcal{R}_j} - 1)dt + q(t_k)\}. \tag{6}$$

B. Tradeoff between Smoothness and Continuity

Our model is in consideration of the tradeoff between smoothness and continuity. Specifically, the rate adaptation scheme should focus on continuous playback if the bandwidth fluctuates dramatically, while a smooth quality playback needs to be ensured when the network condition is stable. In response, we could vary τ according to the network condition.

To track the fluctuation level of bandwidth, we apply coefficient of variance λ, a standardized measure of dispersion of a frequency distribution which is defined as the ratio of the standard deviation to the mean. Using the throughput of preivous n segments, λ can be derived as:

$$\lambda = \frac{\sqrt{n\sum_{i=1}^{n} c_{k-i}^2 - (\sum_{i=1}^{n} c_{k-i})^2}}{\sum_{i=1}^{n} c_{k-i}} \tag{7}$$

where c_i is the throughput of segment i. A high value of λ stands for a fluctuating network condition while a small one represents a stable bandwidth fluctuation.

Intuitively, a more dramatic fluctuating network condition is corresponding to a smaller τ. Thus, τ is a monotonically decreasing function of the bandwidth fluctuation. Approximately, we can envision a constant bandwidth $\mathcal{C}$ with the buffer decreasing from $q(t_k)$ to zero. According to (5), the max value of τ can be deduced as:

$$\tau_{\max} = \frac{q(t_k)}{1 - \mathcal{C}/\mathcal{R}_j} \tag{8}$$

There could be many potential function to describe the relation between τ and the network condition. We take exponential function as an example:

$$\tau = \tau_{\max} \cdot \alpha^{\lambda} \tag{9}$$

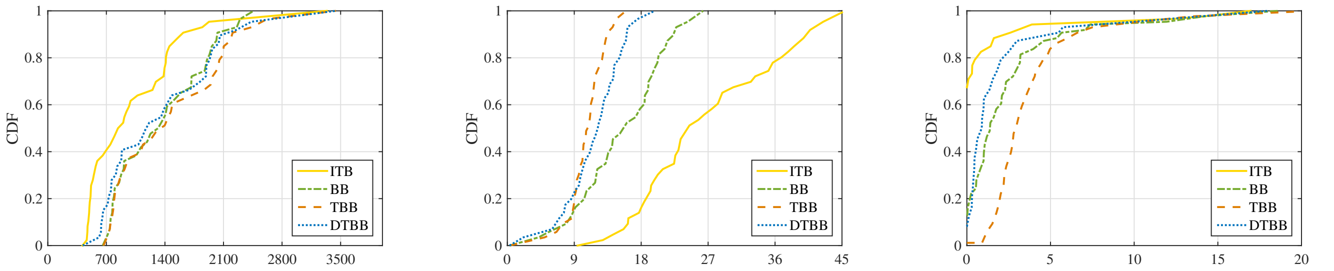
where α is a constant between 0 and 1. Besides, the value of λ can be calculated at the same time when estimating the available bandwidth.

IV. Experiments

In this section, we evaluate the proposed rate adaptation scheme using existing bandwidth trace dataset HSDPA [6]. Same as Netflix, the server provides five different versions of video bit-rate $\mathcal{R} = \{300, 700, 1500, 2500, 3500\}$(Kbps). According to DASH, each version of video is divided into equal-length video segment. In this paper, we fix the length as 1s.

We extend the instant throughput based (ITB) method [2], buffer-based (BB) method [4] and fixed threshold buffer based (TBB) method [7] into live streaming. To make fair comparison on the performance of different methods in the steady state while obtaining a low startup delay in live streaming, all methods are set to request the lowest bitrate in the startup phase. In the steady state, for ITB method, the bit-rate is picked as the maximal value which is just less than $\mu = 0.9$ times the measured segment throughput of the last segment. As for BB method, we employ the function suggested by Huang et al. [4]. For TBB, the underflow threshold is chosen as 1s, focusing mainly on quality smoothness. These three methods are in comparison with our dynamic threshold buffer based (DTBB) method with $\alpha = 0.5$, in terms of average bit-rate, quality switch ratio (the percentage of times of quality switches over total downloaded segments) and playback freeze ratio (the percentage of the duration of playback freezes over the total video streaming period).

We first evaluate the rate adaptation performance under $q_0 = 6s$. Fig. 2 shows the CDF curve of different metrics over the bandwidth trace dataset. As seen in Fig. 2(a), since ITB method always chooses video bit-rate less than the measured throughput, the playback buffer filling rate is faster than its depleting rate. Due to the constraint of live segment generation, the client will always choose a low bit-rate segment and wait for the new one. It brings about the lowest average bit-rate being nearly 500Kbps lower than other methods in 80% of the conditions. In addition, as shown in Fig. 2(b), ITB method is sensitive of bandwidth fluctuation, which makes it suffer from the most fluctuating bit-rate. On the contrary, the TBB method tries to maintain a high bit-rate as much as the buffer allows since it will not switch to a lower quality version unless the buffer occupancy is lower than the fixed underflow-threshold, leading to fewer quality switches. Meanwhile, TBB is so aggressive that playback freezes happen most frequently as seen in Fig. 2(c). Our proposed method will not increase bit-rate switches much, e.g. it only generates less than 14% bit-rate switch ratio under about 80% of the conditions, but the method can reduce playback freezes compared with other buffer-based methods while holding similar average bit-rate. It is necessary to realize that the bandwidth in HSDPA dataset can be less than the lowest bit-rate in the dataset, so playback freezes cannot be avoided completely.

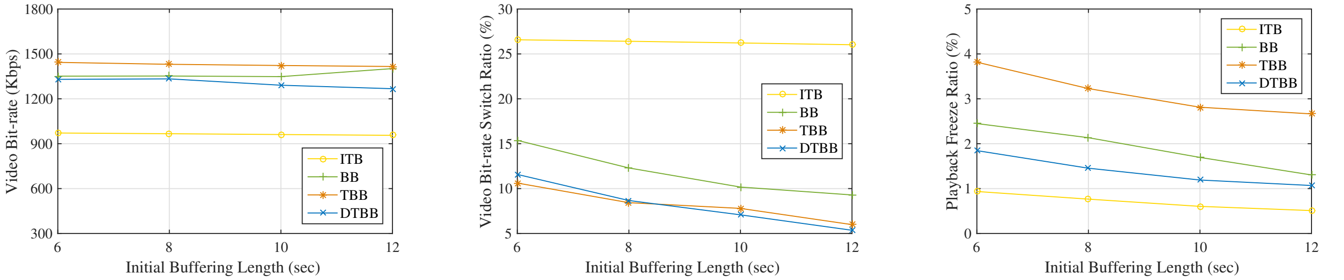


(a) Video Bit-rate (Kbps)

(b) Video Bit-rate Switch Ratio (%)

(c) Playback Freeze Ratio (%)

Fig. 2: Experiment results of different rate adaptation methods



(a) Average Bit-rate (Kbps)

(b) Average Bit-rate Switch Ratio (%)

(c) Average Playback Freeze Ratio (%)

Fig. 3: Experiment results of different rate adaptation methods under different initial buffering length ($q_0 = \{6, 8, 10, 12\}(sec)$)

Then, we evaluate the impact of initial buffering length since it influences time delay in live streaming. As illustrated in Fig. 3, ITB method remains nearly constant on video bit-rate and bit-rate switch ratio due to its independence of buffer occupancy. Apart from this method, longer initial buffering length will produce fewer quality switches and playback freezes since it introduces larger buffer size which is able to tolerate throughput fluctuations. However, it increases end-to-end latency. We can also observe that our proposed method performs well in terms of the above factors.

V. Conclusion

In the new trend of HTTP adaptive streaming, various adaptation methods have been developed. In this paper, we have specifically studied the rate adaption problem under Live streaming with small initial buffering delay. We first present the client buffer model for live adaptive streaming. We find out, in live streaming, the buffer will never suffer overflow since the buffer has a upper bound which equals to the initial buffering delay. Using this model, we proposed a buffer based rate adaptation approach with a dynamic threshold considering both the network fluctuation information and client buffer occupancy. The approach reduces the number of rate switches and maintains a seamless streaming which increases the quality of experience for live viewer under low latency requirement. The extensive experiments demonstrate that the proposed approach outperforms the existing schemes in terms of smoothness and continuity.

VI. Acknowledgment

This work was supported by National Natural Science Foundation of China under contract No. 61471009 and Culture Development Funding under Grant No.2016-288.

References

[1] T. Stockhammer, “Dynamic adaptive streaming over http–: standards and design principles,” in *Proceedings of the second annual ACM conference on Multimedia systems (MMSys’11)*, 2011, pp. 133–144.

[2] T. C. Thang, Q.-D. Ho, J. W. Kang, and A. T. Pham, “Adaptive streaming of audiovisual content using mpeg dash,” *IEEE Transactions on Consumer Electronics*, vol. 58, no. 1, pp. 78–85, 2012.

[3] C. Zhou, X. Zhang, L. Huo, and Z. Guo, “A control-theoretic approach to rate adaptation for dynamic http streaming,” in *IEEE Visual Communications and Image Processing (VCIP)*, 2012, pp. 1–6.

[4] T.-Y. Huang, R. Johari, N. McKeown, M. Trunnell, and M. Watson, “A buffer-based approach to rate adaptation: Evidence from a large video streaming service,” *SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 4, pp. 187–198, 2015.

[5] T. C. Thang, H. T. Le, A. T. Pham, and Y. M. Ro, “An evaluation of bitrate adaptation methods for http live streaming,” *IEEE Journal on Selected Areas in Commun. (JSAC)*, vol. 32, no. 4, pp. 693–705, 2012.

[6] H. Riiser, T. Endestad, P. Vigmostad, C. Griwodz, and P. Halvorsen, “Video streaming using a location-based bandwidth-lookup service for bitrate planning,” *ACM Trans. Multimedia Comput. Commun. Appl (TOMCCAP)*, vol. 8, no. 3, p. 24, 2012.

[7] C. Müller, S. Lederer, and C. Timmerer, “An evaluation of dynamic adaptive streaming over http in vehicular environments,” in *Proceedings of the 4th Workshop on Mobile Video*. ACM, 2012, pp. 37–42.