Rate Adaptation for HTTP Video Streaming to Improve the QoE in Multi-client Environments

Dooyeol Yun and Kwangsue Chung
Department of Electronics and Communications Engineering, Kwangwoon University
447-1, Wolgye-dong, Nowon-gu, Seoul, Korea
[e-mail: dyyoon@cclab.kw.ac.kr, kchung@kw.ac.kr]
*Corresponding author: Kwangsue Chung

Received June 11, 2015; revised August 29, 2015; accepted September 20, 2015; published November 19, 2015

Abstract

Hypertext Transfer Protocol (HTTP) adaptive streaming has become a new trend in video delivery. An HTTP adaptive streaming client needs to effectively estimate resource availability and demand. However, due to the bitrate of the video encoded in variable bitrate (VBR) mode, a bitrate mismatch problem occurs. With the rising demand for mobile devices, the likelihood of cases where two or more HTTP adaptive streaming clients share the same network bottleneck and competing for available bandwidth will increase. These mismatch and competition issues lead to network congestion, which adversely affects the Quality of Experience (QoE). To solve these problem, we propose a video rate adaptation scheme for the HTTP video streaming to guarantee and optimize the QoE. The proposed scheme estimates the available bandwidth according to the bitrate of each segment and also schedules the segment request time to expedite the response to the bandwidth variation. We used a multi-client simulation to prove that our scheme can effectively cope with drastic changes in the connection throughput and video bitrate.

Keywords: HTTP Adaptive Streaming, Multi-client Environment, Variable Bitrate (VBR), Rate Adaptation, Qualit of Experience (QoE)

This work was supported by Institute for Information & Communications Technology Promotion(IITP) Grant funded by the Korea government(MSIP) (No. R0101-15-293, Development of Object-based Knowledge Convergence Service Platform using Image Rcognition in Broadcating Contetnts).

http://dx.doi.org/10.3837/tiis.2015.11.015 **ISSN : 1976-7277**

1. Introduction

In recent years, Hypertext Transfer Protocol (HTTP) adaptive streaming has become increasingly popular due to the abundance of web platforms and broadband connections [1][2]. In HTTP adaptive video streaming, the client needs to continuously estimate the available bandwidth based on the segment information in order to determine the quality level. For certain intervals, the client may decide to change the quality level or to remain at the current level. The quality levels are predefined and are communicated to the client by the server at the beginning of the session. However, the bitrate of a video encoded in variable bitrate (VBR) mode may also vary widely according to the characteristics of the content, and thus the mismatch of both the predefined bitrate and segment bitrate is a significant challenge in video streaming.

With abundant video content and increasing bandwidth demands, it is becoming increasingly likely that two or more adaptive streaming players will a network bottleneck and compete for the available bandwidth. For example, such competition can occur when several people in the same house watch different movies at the same time. The residential broadband access link is likely to be a shared bottleneck. Another example of such competition occurs when many users view the same live event online and an edge network link may constitute the shared bottleneck. Such competition has been previously observed to lead to network congestion [3][4], as each client operates independently and has no understanding of the traffic competition. Each client considers the traffic of the other clients as merely background traffic that leads to the unpredictable and potentially oscillating behavior of each session and ultimately to a negative Quality of Experience (QoE) for each user, particularly in a home environment.

In this paper, we propose a video rate adaptation scheme that guarantees QoE factors, including seamlessness. The proposed scheme estimates the available bandwidth according to the bitrate of each segment. To avoid the synchronization of the rate adjustment intervals due to the multi-client competition, our scheme controls the segment request time by considering the client's buffer occupancy. Also, it compensates for the inaccurate bandwidth estimation.

The remainder of this paper is organized as follows: an overview of the HTTP adaptive streaming and existing video streaming schemes are presented in Section 2. In Section 3, we describe the concepts and algorithms introduced in the rate adaptation scheme for HTTP adaptive streaming. The results of the simulation are extensively reviewed in Section 4. Finally, we present the conclusions in Section 5.

2. Background and Related Works

In this section, an overview is provided of HTTP adaptive streaming and the existing video quality adaptation schemes of HTTP adaptive streaming are reviewed.

2.1 Overview of HTTP Adaptive Streaming

Fig. 1 illustrates the concept of the HTTP adaptive streaming [5]. In HTTP adaptive streaming, the server holds multiple profiles of the same video, encoded in different bitrates and quality levels. Further, the video object is partitioned into segments, typically a few seconds long. A

client can then adapt to the underlying dynamic network conditions such as available bandwidth fluctuations and delay jitter by requesting different segments at different encoding bitrates. The adaptation logic is built into the client rather than the server to determine the bitrate for each segment request. This improves the server-side scalability and increases the system's performance. Another benefit of this approach is that it provides the ability to dynamically adjust the rate at which new segments are requested, allowing the client to control the playback buffer size.

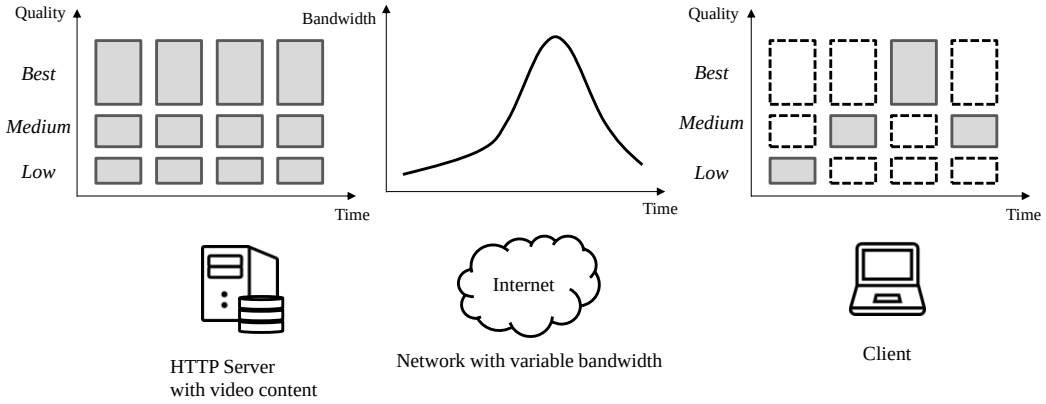

Fig. 1. Concept of the HTTP adaptive streaming

The typical adaptation algorithm is built into the client and the client behavior outlined here has been experimentally observed in recent studies [6][7]. An HTTP adaptive streaming client typically starts a streaming session in the buffering-state. The client prefers to build up its playback buffer as quickly as possible, and to do so, the client requests a new segment as soon as the previous one has been downloaded. The playback usually starts after a few seconds of content has been downloaded. After this phase, one second of video is consumed at every second [6].

Once the size of the playback buffer reaches a certain threshold, the client switches to a steady-state during which the playback buffer is maintained at the same size. For simplicity, each segment is assumed to correspond to T seconds of content, so the client requests next segment every T seconds or as soon as the previous segment has been received. At the same time, the client estimates its fair share in the underlying network path by measuring the per-segments Transmission Control Protocol (TCP) throughput, and computing a running average of these measurements to estimate the throughput of the next segment as shown in (1) [7],

$$\hat{A}(i+1) = \begin{cases} \delta \cdot \hat{A}(i) + (1-\delta) \cdot A(i) & i > 0 \\ A(i) & i = 0 \end{cases} \tag{1}$$

where $A(i)$ and $\hat{A}(i)$ denote the i_{th} segment throughput and the average throughput after downloading the i_{th} segment, respectively. δ is the weighting parameter. The client uses this average to select the bitrate for the next segment that has been requested. If the buffer size falls below the given target, the client switches back from the steady-state to the buffering-state to

quickly refill the buffer again. This behavior continues until the streaming is complete. The rate adaptation scheme for HTTP adaptive streaming needs to account for the change in the network conditions and the client buffer state, and selects the most suitable quality level in order to improve the QoE.

2.2 Rate Adaptation Schemes for HTTP Adaptive Streaming

Akshabi et al. report that mutual synchronization among clients is a relevant cause of unfairness [8], particularly when clients request video segments at different times that results in incorrect bandwidth estimations. Several HTTP adaptive streaming clients have been proposed to deal with the aforementioned problems. In [9], the measure of the segment fetch time is used to determine the requested video bitrates in an aggressive decrease and step-wise increase manner. Instead of using a TCP-like mechanism, a reliable estimation of the throughput for city commuters using the prior-knowledge of commuting routes is used to determine video bitrate [10]. To obtain the appropriate quality of media, the authors present an algorithm to estimate the bandwidth which involves bandwidth detective method and bandwidth computing method [11]. In [12], the authors present a novel approach for the throughput estimation, where the mechanism is stable against short-term fluctuations but responds quickly to large fluctuations of the networks. Jiang et al. propose the Fair, Efficient, and Stable adapTIVE algorithm (FESTIVE), a system that may lead to fairness improvement including the quality selection interval, a stateful quality level selection, and a bandwidth estimation using a harmonic mean instead of a normal estimation [13]. FESTIVE is the first adaptive video streaming algorithm specifically designed to address the fairness issues arising in a multi-client scenario. In another recent proposal [14], the authors design the Probe AND Adapt (PANDA), an algorithm that dynamically computes the segment inter-request time to address the fairness issues, and video bitrate oscillations. L. Cicco et al. propose the feedback linearization adaptive streaming controller, a client-side stream-switching controller designed to use feedback control theory to avoid the generation of on-off traffic pattern and to fairly share the bottleneck with other videos and greedy TCP flows [15]. Also, the advantages and disadvantages of different strategies to schedule HTTP requests are investigated in [16][17]. In [16], the authors provide experimental evidence that multiple HTTP-based request-response streams are a good alternative to classical TCP streaming. In [17], the authors present experimental results showing the benefits and drawbacks of various strategy with respect to achieved video quality, smoothness of playback and end-to-end delay.

In general, the works found in the literature share some common limitations. First, the previous studies have only dealt with a constant bitrate (CBR) video. In practice, most video streaming services encode videos in variable bitrate (VBR), since VBR allows for increased flexibility and can use bytes more efficiently. Second, the previous algorithms use a fixed and static buffer threshold, yet they should modify their threshold when the environment changes. Hence, a rate adaptation for HTTP video streaming is necessary to consider VBR video and improve the flexibility of the QoE in multi-client environments.

3. Rate Adaptation Scheme for HTTP Adaptive Video

A new rate adaptation scheme for HTTP adaptive video streaming in multi-client environment is proposed to guarantee the QoE of streaming services. The proposed scheme estimates the available bandwidth using the recalculated segment bitrate, and it controls segment request

intervals according to the buffer occupancy of the client. In this section, we provide an overview of the proposed scheme for HTTP adaptive video streaming as well as the detailed architecture and algorithms used in the rate adaptation scheme.

3.1 Overview of a Video Streaming Scheme

A streaming service requires rate adaptation to estimate the network bandwidth and selects the proper quality level. We propose a rate adaptation scheme in a multi-client environments and **Fig. 2** shows the architecture of the proposed scheme which operates on the client side.

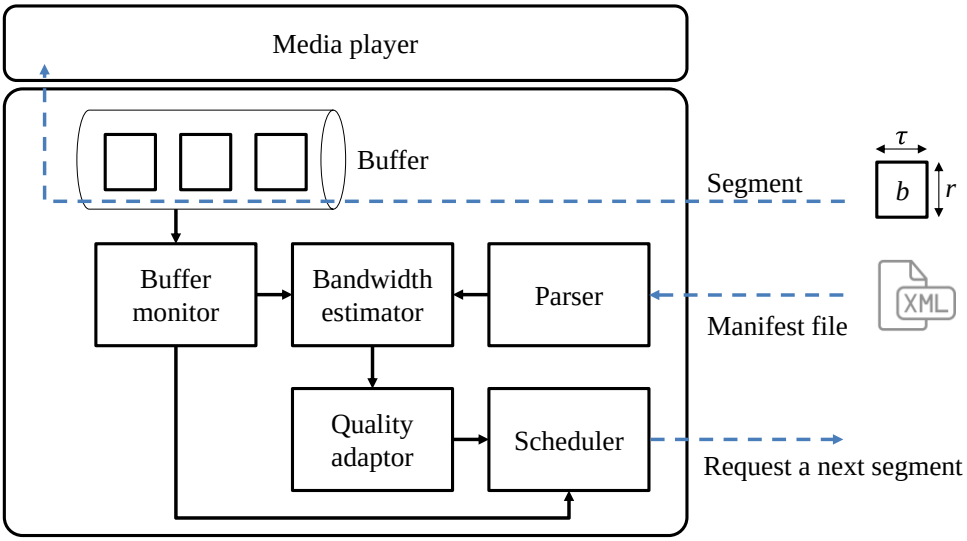

Fig. 2. Architecture of the proposed rate adaptation scheme

Our scheme consists of the *buffer monitor*, the *bandwidth estimator*, the *parser*, the *quality adaptor*, and the *scheduler*. The *buffer monitor* checks the buffer occupancy and segment download time. The *parser* extracts the segment information from the manifest file. The *bandwidth estimator* computes the segment bitrate using the buffer occupancy and segment information received from the *buffer monitor* and the *parser*. Based on the estimated bandwidth, the *quality adaptor* selects the quality level of the next segment. The *scheduler* controls the interval at which segments are requested by considering the quality level and buffer occupancy. For example, when client requests a video streaming service, it received the manifest file from the server. The *parser* extracts the information such as download address for the segments, the number of segments and the average bitrate for the given quality level from the manifest file. Then, the client continues to request the segments by considering the segment requesting interval and the quality level. We provide further details to compute the segment requesting interval and select the quality level in Sections 3.2 and 3.3. Table 1 lists the notations used in this paper. In this paper, we use '*' to distinguish the proposed scheme and the traditional scheme.

Table 1. Notations used in this paper

Notation	Explanation
$r_i(n)$	Bitrate of the quality level i for n_{th} segment
r_i	Average bitrate of the quality level i
τ	Video segment duration
t_d	Download time for video segment time
$x(n)$	TCP throughput measured at n_{th} segment
N	The number of video segments
B	Client buffer occupancy
$B_t(n)$	Target buffer threshold for n_{th} segment
T	Inter-request time

3.2 Bandwidth Estimation

We consider the video to be encoded in VBR; i.e. not all segments that are encoded at a given nominal rate are of the same size. When the video is encoded in VBR at a nominal video rate, the rate represents the average video rate and the instantaneous video rate varies around this average value. As a result, the segment size will not be uniform for a stream of a given rate. **Fig. 3** shows the segment size of VBR videos encoded at 1300 Kbps.

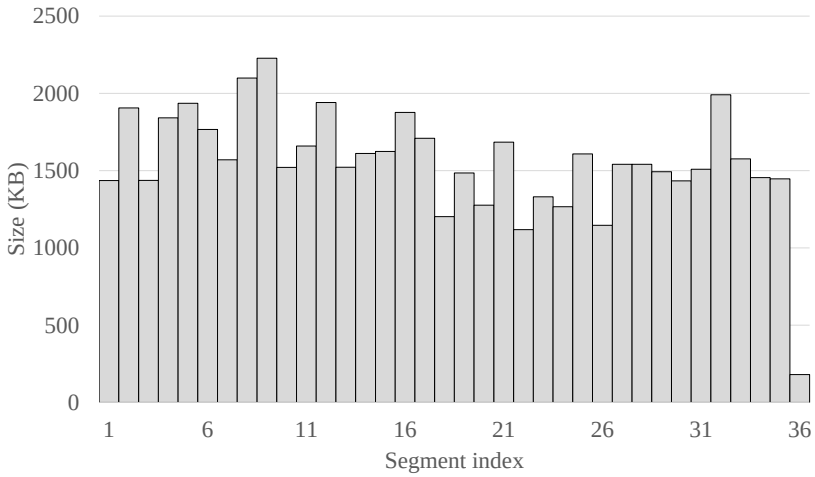

Fig. 3. Segment size of the VBR video

Fig. 3 shows the comparison of the segment size of two videos encoded with the CBR and VBR. In the CBR video, the video rate for all segments is equal: $r_i(n) = r_i(n+1)$. On the other hand, in the VBR video, the instantaneous bitrate of each segment may differ from other segments encoded at the same quality level: $r_i(n) \neq r_i(n+1)$.

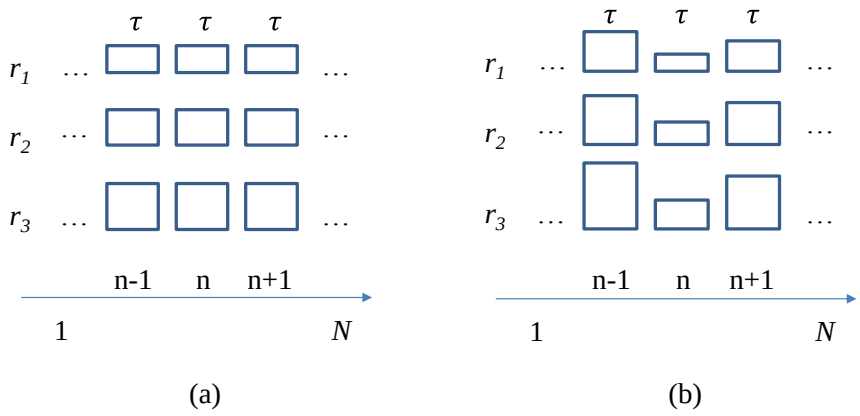

Fig. 4. Comparison of the segment size: (a) CBR video, (b) VBR video

Assuming that all segment sizes are the same, the traditional rate adaptation scheme estimates the available bandwidth using:

$$x(n) = r_i \cdot \tau / t_d(n) \tag{2}$$

where $t_d(n)$ is the download time for the n_{th} segment. However, most video streaming services encode their videos in VBR and the manifest file does not have information of the size of each segment. Considering the bitrate of each segment, our scheme estimates the next segment bitrate based on the downloaded segment as (3).

$$r_i^*(n+1) = \frac{r_i \cdot \tau \cdot N - \sum_{k=1}^{n} b(k)}{(N-n) \cdot \tau} \tag{3}$$

where $r_i^*(n+1)$ is the bitrate estimated for the $n+1_{th}$ segment and $b(k)$ is the downloaded segment size of the k_{th} segment. $r_i \cdot \tau \cdot N$ means the total segment size, summation of $b(k)$ means the already received segment size and $(N - n) \cdot \tau$ means the duration of the remaining segment. Based on the estimated segment bitrate, the bandwidth is estimated as shown in (4).

$$x(n+1) = r_i^*(n+1) \cdot \tau / t_d(n) \tag{4}$$

The *quality adapter* selects the quality level, i, according to the estimated bandwidth, as shown in (5). $x(n)$ is then mapped to the discrete bitrate of the quality level.

$$i = \arg\max_i x(n) > r_i \tag{5}$$

The quality level i is selected by the argmax function. It sets the quality level for which $x(n) > r_i$ attains the maximum value.

3.3 Segment Request Time

We are specifically interested in multi-client environments where multiple HTTP adaptive streaming clients share a bottleneck link. HTTP adaptive streaming supersedes TCP behavior and results in unpredictability in the user experience, as previously reported in [8]. Unfair bandwidth sharing may occur when one user receives video with a better quality than other users. Moreover, since each client operates independently of the others, there is no understanding of the competition for traffic by the clients since each client considers the traffic of the other clients to be background traffic. Each streaming session may be subjected to unpredictable and potentially oscillating behavior.

The performance of the adaptive algorithms in a multi-client scenario may be improved by using several methods. Fixed rate adjustment intervals contribute to inaccurate estimations in the available bandwidth and to the resulting oscillating behavior. An alternative to fixed intervals would be to randomize the intervals by using a per-session stochastic parameter. To randomize the interval, one could also consider introducing a back-off period. Following a reduction in the quality level by the client, the back-off period during which an increase in quality is not allowed could improve performance. In addition, rather than always using the same interval for the potential change in quality, a threshold could be introduced where the client operates in a relatively aggressive manner. This threshold can be the average available quality level for a specific session, or even a smoothed average value for the actual quality level that is achieved.

In this paper, the proposed scheme uses a dynamic target buffer threshold to randomize the time intervals. The HTTP adaptive streaming client continues to download the segment until the buffer occupancy exceeds the dynamic target buffer threshold, $B_t(n)$. The segment request time can then be controlled by adjusting the dynamic target buffer. The proposed scheme adjusts the dynamic target buffer using

$$B_t(n+1) = B_t(n) + (\delta / r^*(n)) \cdot \tau \tag{6}$$

where $(\delta/r^*(n)) \cdot \tau$ means the ratio of the estimation error to estimated segment size and δ denotes the bandwidth estimation error as shown in (7).

$$\delta = b - r^*(n) \cdot \tau \tag{7}$$

The bandwidth estimation error is the difference between the estimated segment size and the downloaded segment size. **Fig. 5** shows the target buffer adaptation based on the bandwidth estimation error. If the downloaded segment size is larger than the estimated segment size, the target buffer threshold is increased. A larger target buffer threshold shortens the request time, and as a consequence, the proposed scheme supports seamless playback by maintaining the buffer occupancy at the target buffer.

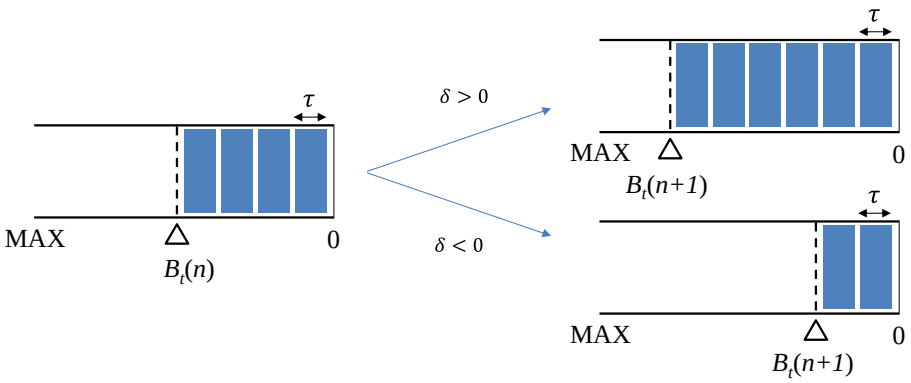

Fig. 5. Target buffer adaptation based on the bandwidth estimation error

3.4 Flow of the Rate Adaptation

The flowchart in **Fig. 6** shows the working principle of the proposed rate adaptation scheme. When the streaming client receives a segment, the estimated bandwidth is computed using (4). The client checks the buffer occupancy after every segment has been downloaded. When the buffer occupancy exceeds the dynamic target buffer, the client sends a request message for the next segment.

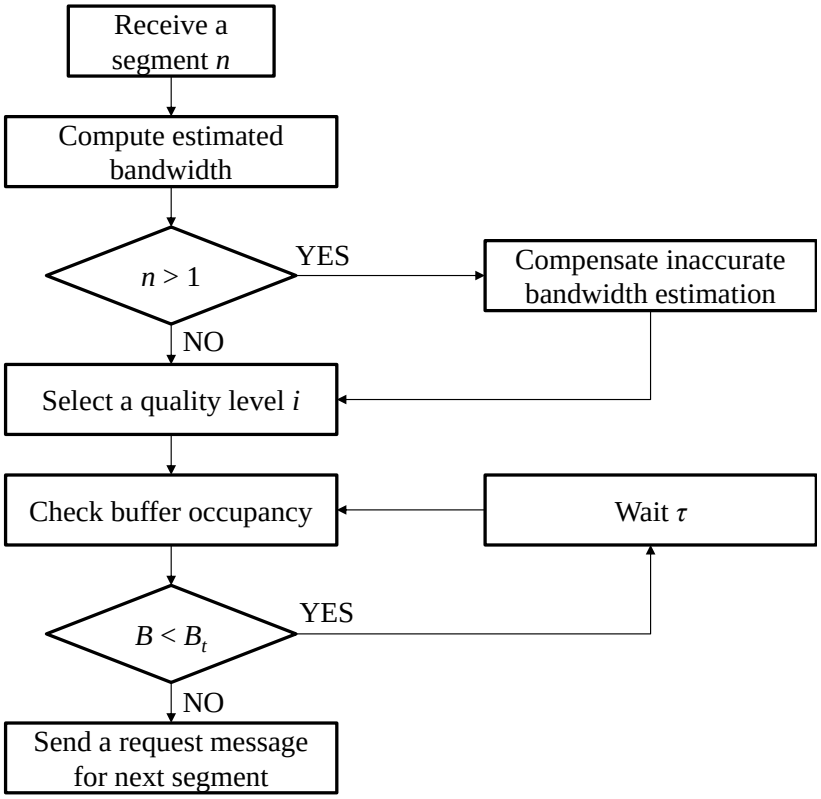

Fig. 6. Flowchart of the proposed rate adaptation scheme

4. Performance evaluation

This section presents and discusses the results of the simulation. In order to evaluate the performance, the proposed scheme is implemented in a network simulator, ns-3, with video pre-encoded at L = 10 bitrates: 459, 693, 937, 1270, 1745, 2536, 3758, 5379, 7861, and 11321 kbps. All segments have a duration of 2 seconds, and the length of the total media presentation is of 100 seconds.

Fig. 7 shows the network topology that is used in the simulations. Our experiments are configured to closely match the scenarios where a number of clients compete for bandwidth. We focus on three metrics for the evaluation: (1) Bandwidth estimation, (2) Requesting time, and (3) Re-buffering ratio. The re-buffering ratio is defined as the percentage of time that the client spends in the re-buffering state. For video streaming users, re-buffering is often the most annoying issue. Hence, it is most important to maintain a low re-buffering percentages through judicious rate adaptation.

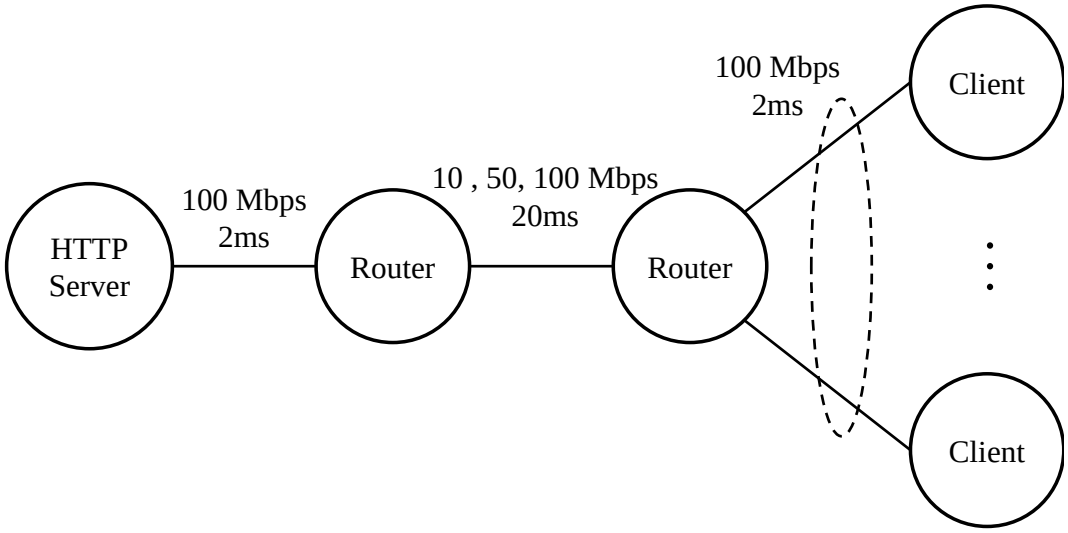

Fig. 7. Network topology

4.1 Bandwidth Estimation

The experimental evaluation is carried out by comparing the proposed algorithm with following schemes.

1) Conventional scheme:

$$x_c(n+1) = r(n) \cdot \tau / t_d(n) \tag{8}$$

2) PANDA:

$$x_P(n+1) = x_P(n) + t_d(n) \cdot \kappa \cdot (\omega - \max(0, x_P(n) - x(n) + \omega)) \tag{9}$$

3) FESTIVE:

$$x_F(n+1) = \frac{2 \cdot x_F(n) \cdot x(n)}{x_F(n) + x(n)} \tag{10}$$

To estimate the bandwidth, conventional scheme uses instanteous throuhgput, PANDA uses the ω, κ parameter, and FESTIVE uses the harmonic mean. In this evaluation, we set the ω = 0.3, κ = 0.14. **Fig. 8** and **Fig. 9** show a comparison of the estimated bandwidths. The proposed scheme outperforms the other schemes, and the large prediction errors in the tail appear because the bandwidth observed for each segment depends on the number of competing clients. The accuracy of the throughput estimation method enables a more stable buffer occupancy, thus helping reduce the initial buffering delay.

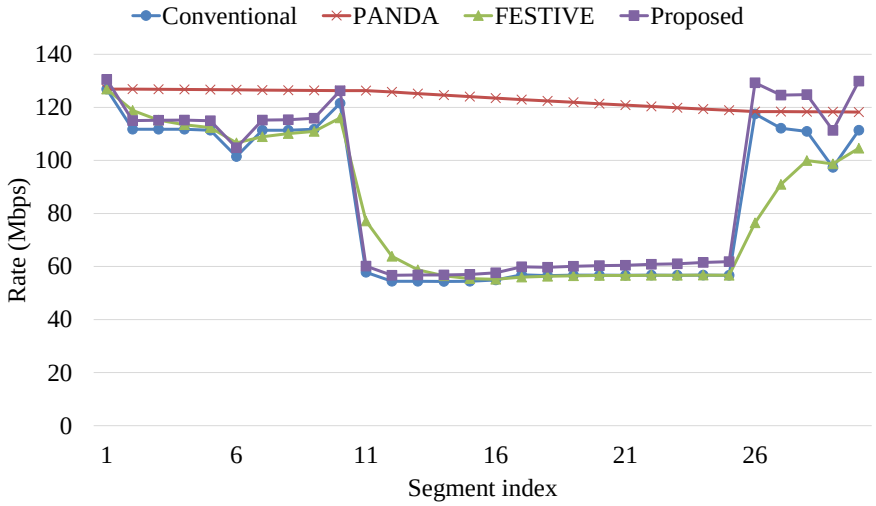

Fig. 8. Comparison of the estimated bandwidth for the CBR video

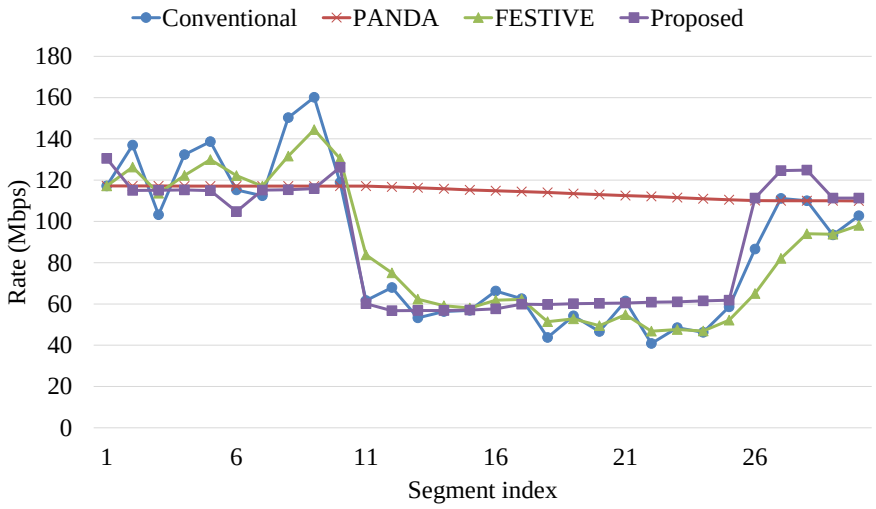

Fig. 9. Comparison of the estimated bandwidth for the VBR video

4.2 Request Time

To determine the request time, we compared the proposed algorithm with the following schemes.

1) Conventional scheme:

$$T_c = \begin{cases} 0, & \mathrm{B(n-1)} < \mathrm{B}_{\max} \\ \tau, & \text{otherwise} \end{cases} \tag{11}$$

2) PANDA:

$$T_P = \frac{r \cdot \tau}{y} + \beta \cdot (B(n-1) - B_{\min}) \tag{12}$$

3) FESTIVE:

$$T_F = \begin{cases} 0, & \text{if } \mathrm{B[n-1]} < randbuf \\ B(n-1) - randbuf, & \text{other wise} \end{cases} \tag{13}$$

Conventional scheme requests the next segment periodically. PANDA considers the buffer occupancy for determining the request time. y means the smoothed bandwidth and we set $\beta = 0.2$. FESTIVE determines the request time by comparing the buffer occupancy and uniform random variable, *randbuf*. **Fig. 10, 11**, and **12** show the average video rates for a variable number of video flows. The figures show that the proposed scheme consistently obtains an average video rate that is very close to the highest video level, regardless of the number of flows. The average video rate obtained by FESTIVE does not depend on the number of flows, but it is affected by the shared link.

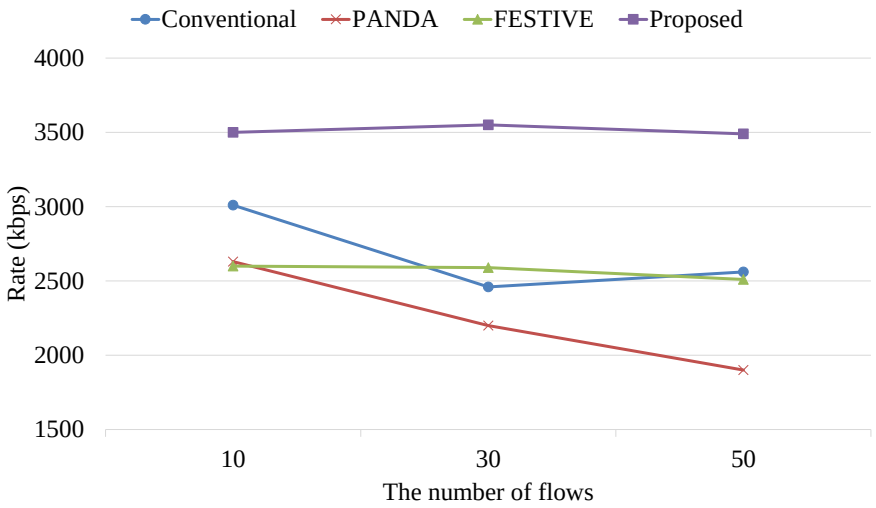

Fig. 10. Average video rate in the case of a variable number of video flows sharing a 100Mbps link

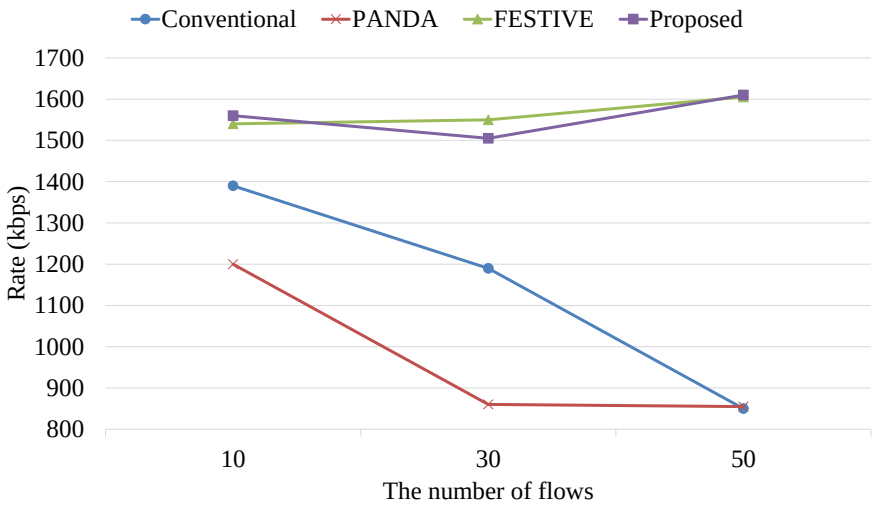

Fig. 11. Average video rate in the case of a variable number of video flows sharing a 50Mbps link

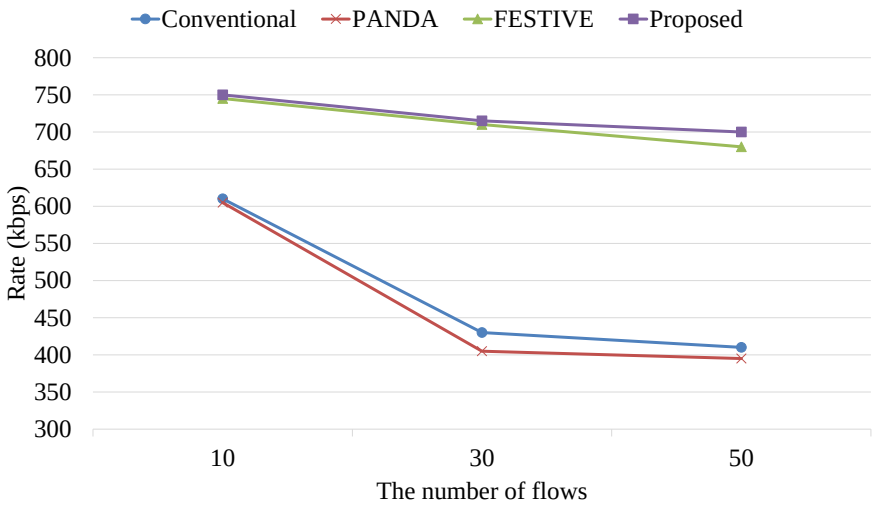

Fig. 12. Average video rate in the case of a variable number of video flows sharing a 10Mbps link

4.3 Re-buffering Ratio

Fig. 13 shows that both the proposed and the conventional schemes provide re-buffering ratios of less than 5%. FESTIVE and PANDA exhibit an increase in the re-buffering ratios when the number of flows increase. At 50 flows in particular, FESTIVE and PANDA generate 17% and 13% of the re-buffering ratios, respectively.

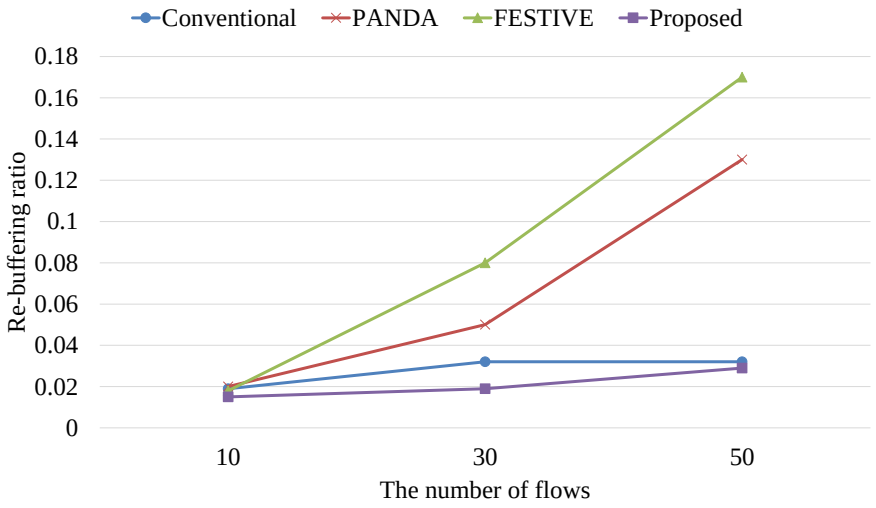

Fig. 13. Re-buffering ratio in the case of a variable number of video flows

5. Conclusion

Mismatch and competition issues pose a significant challenge for multi-client environments. These mismatch and competition issues can lead to network congestion, which adversely affects the QoE. This paper proposes a rate adaptation scheme for HTTP adaptive streaming is proposed to guarantee the QoE of video streaming services. The proposed scheme estimates the average video rate of the upcoming segments for each quality level and the video rate of the next segment is selected according to the bandwidth that is available and estimated average video rate. The proposed scheme expedites the response to variations in the bandwidth by scheduling segment request times. An evaluation of performance indicates that the proposed scheme improves the QoE. In future work, we intend to extend the proposed rate adaptation scheme for HTTP adaptive streaming in mobile networks to provide service for a greater number of use cases to satisfy client needs.

References

[1] A. C. Begen, T. Akgul, and M. Baugher, "Watching Video over the Web, Part I: Streaming Protocols," *IEEE Internet Computing*, vol. 15, no. 2, pp. 54-63, March, 2011. Article (CrossRef Link).

[2] A. C. Begen, T. Akgul, and M. Baugher, "Watching Video over the Web, Part II: Applications, Standardization, and Open Issues," *IEEE Internet Computing*, vol. 15, no. 3, pp. 59-63, March, 2011. Article (CrossRef Link).

[3] S. Akhshabi, L. Anantakrishnan, C. Dovrolis, and A. C. Begen, "What Happens When HTTP Adaptive Streaming Players Compete for Bandwidth," in *Proc. of the ACM Network and Operating System Support on Digital Audio and Video Workshop*, pp.9-14, 2012. Article (CrossRef Link).

[4] M. Xing, S. Xiang, and L. Cai "A Real-time Adaptive Algorithm for Video Streaming over Multiple Wireless Access Networks," *IEEE Journal on Selected Areas in communications*, vol.32, no.4, pp.795-805, December, 2014. Article (CrossRef Link).

[5] C. F. Lai, H. C. Chao, Y. X. Lai, and J. Wan, "Cloud-assisted Real-time Transrating for HTTP Live Streaming," *IEEE Wireless Communications*, vol.3, no.3, pp.62-70, June, 2013. Article (CrossRef Link).

[6] D. H. Lee and A. C. Begen, "Caching in HTTP Adaptive Streaming: Friend or Foe?," in *Proc. of the ACM Network and Operating System Support on Digital Audio and Video Workshop*, pp.31-36, 2014. Article (CrossRef Link).
[7] T. C. Thang, H. T. Le, H. X. Nguyen, A. T. Pham, J. W. Kang, and Y. M. Ro, "Adaptive Video Streaming over HTTP with Dynamic Resource Estimation," *Journal of Communications and Networks*, vol.15, no.6, pp.635-644, December, 2013. Article (CrossRef Link).
[8] S. Akhshabi, A. Begen, and C. Dovrolis, "An Experimental Evaluation of Rate Adaptation Algorithms in Adaptive Streaming over HTTP," in *Proc. of the ACM conference on Multimedia System*, pp.157-168, 2011. Article (CrossRef Link).
[9] C. Liu, I. Bouazizi, and M. Gabbouj, "Rate Adaptation for Adaptive HTTP Streaming," in *Proc. of the ACM Conference on Multimedia System*, pp.169-174, 2011. Article (CrossRef Link).
[10] K. Evensent, A. Petlund, H. Riiser, P. Vigmostad, D. Kaspar, C. Griwodz, and P. Halvorsen, "Mobile Video Streaming Using Location-based Network Prediction and Transparent Handover," in *Proc. of the ACM Network and Operating System Support on Digital Audio and Video Workshop*, pp.21-26, 2011. Article (CrossRef Link).
[11] Q. Lin, Y. Liu, Y. Shen, H. Shen, and L. Sang, "Bandwidth Estimation of Rate Adaptation Algorithm in DASH," in *Proc. of the IEEE Globecom Workshops*, pp. 243-247, 2014. Article (CrossRef Link).
[12] J. Jiang, V. Sekar, and H. Zhang, "Improving Fairness, Efficiency, and Stability in HTTP-based Adaptive Video Streaming with FESTIVE," in *Proc. of the ACM Conference on Emerging Networking Experiments and Technologies*, pp.97-108, 2012. Article (CrossRef Link).
[13] Z. Li, X. Zhu, J. Gahm, R. Pan, H. Hu, A. C. Begen, and D. Oran, "Probe and Adapt: Rate Adaptation for HTTP Video Streaming At Scale," *IEEE Journal on Selected Areas in Communications*, vol.32, no.4, pp.719-733, December, 2014. Article (CrossRef Link).
[14] L. D. Cicco, V. Caldaralo, V. Palmisano, and S. Mascolo, "ELASTIC: A Client-side Controller for Dynamic Adaptive Streaming over HTTP (DASH)," in *Proc. of the IEEE Packet Video Workshop*, pp.214-225, 2013. Article (CrossRef Link).
[15] R. Kuschinig, I. Kofler, and H. Hellwagner, "Evaluation of HTTP-based Request-response Streams for Internet Video Streaming," in *Proc. of the ACM Conference on Multimedia Systems*, pp.245-256, 2011. Article (CrossRef Link).
[16] T. Kupka, P. Halvorsen, and C. Griwodz, "An Evaluation of Live Adaptive HTTP Segment Streaming Request Strategies," in *Proc. of the IEEE Conference on Local Computer Networks*, pp.604-612, 2011. Article (CrossRef Link).

Dooyeol Yun received the B.S. degree from Kwangwoon University, Seoul, Korea, from the Electronics and Communications Engineering Department, in 2011. Currently he is pursuing his Ph.D. degree in the Electronics and Communications Engineering Department, Kwangwoon University. His research interests include multimedia, mobile TV, and QoE support and rate control for video communications.

Kwangsue Chung received the B.S. degree from Hanyang University, Seoul, Korea, M.S. degree from KAIST (Korea Advanced Institute of Science and Technology), Seoul, Korea, Ph.D. degree from University of Florida, Gainesville, Florida, USA, all from Electrical Engineering Department. Before joining the Kwangwoon University in 1993, he spent 10 years with ETRI (Electronics And Telecommunications Research Institute) as a research staff. He was also an adjunct professor of KAIST from 1991 to 1992 and a visiting scholar at the University of California, Irvine from 2003 to 2004. His research interests include communication protocols and networks, QoS mechanism, and video streaming. Dr. Chung is a senior member of IEEE.