LETTER Special Section on Enriched Multimedia —New Technology Trends in Creation, Utilization and Protection of Multimedia Information—

Quality Improvement for Video On-Demand Streaming over HTTP

Huyen T. T. TRAN[†], **Hung T. LE**[†], **Nam PHAM NGOC**[††], ***Nonmembers***, **Anh T. PHAM**[†], ***and*** **Truong Cong THANG**[†a)], ***Members***

SUMMARY It is crucial to provide Internet videos with the best possible content value (or quality) to users. To adapt to network fluctuations, existing solutions provide various client-based heuristics to change video versions without considering the actual quality. In this work, we present for the first time the use of a quality model in making adaptation decisions to improve the overall quality. The proposed method also estimates the buffer level in the near future to prevent the client from buffer underflows. Experiment results show that the proposed method is able to provide high and consistent video quality under strongly fluctuating bandwidths.

key words: *quality improvement, video streaming, adaptivity, HTTP*

1. Introduction

More and more video contents are being provided via the Internet today. However, modern access networks are heterogeneous with time-varying characteristics. So, it is crucial to provide Internet videos with the best possible content value (quality) to users. Recently, HTTP adaptive streaming (HAS), has become the key solution for video streaming [1], [2]. For adaptivity to throughput variations, a streaming provider should generate multiple versions of an original video, each of which is further divided into short segments. The metadata, which contains the characteristics of the video versions (such as frame rate, bitrate, etc.), is sent to the client at the beginning of a streaming session. Then, based on the metadata and the status of the network connection, an adaptation method at the client decides which version is requested for each video segment [1], [2].

To guarantee the quality of service, various adaptation methods have been proposed (e.g. [3]–[5]). In adaptive streaming, most of existing methods take advantage of the client buffer to reduce the degree/frequency of version switching and heuristically improve video quality. A typical approach is dividing the buffer into multiple ranges, each is associated with specific adaptation rules [4], [5]. An overview and extensive evaluations of typical adaptation methods have recently been provided in [2].

Besides, some quality models to evaluate the quality of adaptive streaming have been attempted recently [6]–[8]. In [6], the overall quality is predicted from the qualities of the composing frequency components which are decomposed from the quality variations in a session. In [7], the quality model is mainly based on the average and standard deviation of segment quality values, and switching frequency. In [8], we introduced a quality model based on the histograms of segment quality values and quality changes. An advantage of this model is that its inputs are the quantization parameters (QPs) of video segments, which are available in metadata or in the headers of video data. This enables a client to 1) measure the overall quality in real time and also 2) select video versions to improve the overall quality.

To the best of our knowledge, no previous adaptation methods have considered a quality model in making decisions. In this work, we propose a new method, which employs the quality model of [8] to improve the overall quality. The proposed method also considers buffer levels in the near future to prevent the client from buffer underflows. Experimental results show that the proposed method can consistently achieve high video quality while still providing buffer stability. It should be noted that the proposed method could be used with any future quality models that support real-time measurement.

The letter is organized as follows. Section 2 reviews the quality model used in our method. The proposed method is described in detail in Sect. 3. Then, experiments and conclusions are presented in Sect. 4 and Sect. 5.

2. Quality Model for Adaptive Streaming

Given a sequence of received video segments (from segment 1 to the current segment i), we denote QoE_i the overall quality of the session until segment i. This quality metric, which obviously varies in time, depends on the quality values and the quality variations of the consecutive segments.

In our quality model [8], each segment i is attributed by a quality value Q_i in MOS score, which is modeled by a function of the segment's average QP as in [9]. Then the histogram of segment quality values $\{Q_i\}$ and the histogram of quality changes $\{\Delta Q_i\}$ are used to predict the overall quality. Specifically, the quality value range is split into H quality intervals denoted by IQ_h where $h \in \{1, 2, \ldots, H\}$. The frequency of the quality values in each interval IQ_h accumulated until segment i is denoted by F_{i,IQ_h}.

As for quality changes, we use the concept of "quality gradient". The quality differences between every two con-

Manuscript received April 5, 2016.
Manuscript revised August 2, 2016.
Manuscript publicized October 7, 2016.

[†]The authors are with the University of Aizu, Aizuwakamatsu-shi, 965–8580 Japan.
[††]The author is with Hanoi University of Science and Technology, 1 Dai Co Viet, Hanoi, Vietnam.
a) E-mail: thang@u-aizu.ac.jp

DOI: 10.1587/transinf.2016MUL0005

secutive segments are used to represent the instant gradients. The instant gradient value range is split into K value intervals denoted by IG_k where $k \in \{1, 2, \ldots, K\}$. The frequency of the instant gradients in each interval IG_k until segment i is denoted by F_{i,IG_k}. Finally, the overall quality QoE_i until segment i is modeled by pooling the above statistics of the segment qualities and quality gradients as follows

$$QoE_i = \sum_{h=1}^{H} \alpha_h F_{i,IQ_h} + \sum_{k=1}^{K} \beta_k F_{i,IG_k}, \quad (1)$$

where α_h and β_k are model parameters obtained from a training dataset. In [8], this quality model was evaluated by 34 different bandwidth traces (or sessions) and its performance was shown to be better than those of [6], [7].

Note that previous quality models mainly use the average or median, the minimum, and the standard deviation of the quality values in a session. Using the histograms of segment quality values and quality gradients, our model can flexibly capture different types of quality variations. Moreover, as mentioned above, the inputs of the model are QPs of video segments, which enable the model to be used in the proposed adaptation method.

3. Proposed Adaptation Method

3.1 Overview

The general mechanism of our method is illustrated in Fig. 1. The quality model of a video is obtained in advance and included in the metadata, which is sent to the client before a session. Based on the QPs of received segments, the client can compute segment quality values and the overall quality at any time instant. The adaptation method at the client then decides the version of a next segment to maximize the overall quality while meeting buffer stability constraints.

Suppose that, on a server, a video is encoded into N versions, where version 1 (N) is the lowest (highest) quality version. Each version is divided into segments with the duration of τ seconds. Denote B_{max} the client buffer size. Similarly to other buffer-based methods [3]–[5], we use buffer thresholds B_{min}, B_{low}, B_{high} ($0 < B_{min} < B_{low} < B_{high} < B_{max}$) to divide the buffer into four different ranges, namely a high range $(B_{high}, B_{max}]$, a safe range $(B_{low}, B_{high}]$, a low range $(B_{min}, B_{low}]$, and a dangerous range $[0, B_{min}]$. Note that, in adaptive streaming, these parameters of the buffer are measured in a time unit [2]. This is because video bitrate (or segment datasize) in this context is adapted according to throughput variations. A current buffer level of 20s, for example, means that there is 20s of video data (possibly having varying bitrate) in the buffer, and thus the client can continue to play for 20s even if the connection is broken. So, it is obvious that the higher the buffer level is, the lower the chance of buffer underflows will be in the near future.

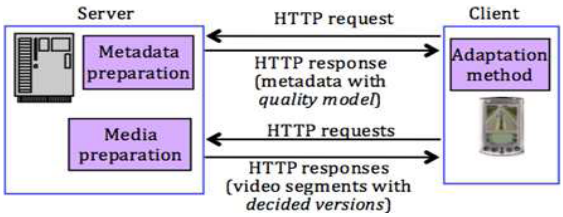

Fig. 1 General mechanism of the proposed method.

Now suppose that, after completely receiving the current segment i of version V_i ($0 \le V_i \le N$), the client measures the instant throughput T_i and the current buffer level B_i^{cur}, and then decides the version for the next segment. The estimated throughput T^e of the next segment(s) is simply set to the instant throughput, i.e. $T^e = T_i$.

Denote $R_{n,k}$ the bitrate of version k at segment n, which can be obtained from metadata [3]. After receiving the next segment $i+1$ with version V_{i+1}, the buffer has τ more seconds of media. However, the delivery of that segment would take $(\tau \times R_{i+1,V_{i+1}} / T_e)$ seconds. So, the estimated buffer level $B^e_{i+1,V_{i+1}}$ after receiving the next segment $i+1$ can be computed by

$$B^e_{i+1,V_{i+1}} = B_i^{cur} + \tau - \tau \times \frac{R_{i+1,V_{i+1}}}{T^e} \quad . \quad (2)$$

Based on the throughput trend, our method is divided into two cases: *downtrend case* ($T_i \le T_{i-1}$) and *uptrend case* ($T_i > T_{i-1}$). In any cases, if the buffer level B_i^{cur} is in the dangerous range, the lowest version will be selected (i.e. $V_{i+1} = 1$) to avoid buffer underflows.

3.2 Downtrend Case

In this case ($T_i \le T_{i-1}$), if the current buffer level is in the high or safe ranges ($B_i^{cur} > B_{low}$), the current version is simply kept (i.e. $V_{i+1} = V_i$) to maintain a high quality for the user. However, if the current buffer level is in the low range ($B_{low} \ge B_i^{cur} > B_{min}$), the client should decide whether to maintain/decrease the version to avoid buffer underflows as well as sudden quality degradations.

When using the quality model, if only one next segment $i+1$ is considered in making decision, the client will tend to select the highest quality version that can be supported by the current buffer level. As the throughput is going down, this selection will decrease the buffer level quickly, and the version of segment $i+2$ must be drastically reduced to avoid buffer underflows. For that reason, our method will consider jointly two next segments $i+1$ and $i+2$ with versions V_{i+1} and V_{i+2}. Because the current version is V_i, V_{i+1} and V_{i+2} may vary in the range $[1, V_i]$. Similarly to segment $i+1$, the estimated buffer level of segment $i+2$ is computed by

$$B^e_{i+2,V_{i+2}} = B^e_{i+1,V_{i+1}} + \tau - \tau \times \frac{R_{i+2,V_{i+2}}}{T^e} \quad . \quad (3)$$

The versions of the next segments are selected so that the overall quality QoE_{i+2} (1) is maximized and the future buffer levels $B^e_{i+1,V_{i+1}}$ and $B^e_{i+2,V_{i+2}}$ are not in danger.

For segment i+1, the buffer constraint is that $B^e_{i+1,V_{i+1}} > B_{min}$. For segment i+2, because $B^e_{i+2,V_{i+2}}$ is estimated from the estimated buffer level $B^e_{i+1,V_{i+1}}$, the estimation error could be large. Therefore, the second buffer constraint is defined by $B^e_{i+2,V_{i+2}} > B_{min} + \Delta B_{err}$, where ΔB_{err} is a safety factor to cope with estimation error. Here, ΔB_{err} is set to τ, i.e. one segment duration. So, the decision problem can be stated as follows.

Find V_{i+1}, V_{i+2} $(1 \leq V_{i+1}, V_{i+2} \leq V_i)$ *that maximizes* QoE_{i+2} *and satisfies*

$$B^e_{i+1,V_{i+1}} > B_{min} \;\&\&\; B^e_{i+2,V_{i+2}} > B_{min} + \Delta B_{err}. \tag{4}$$

As the problem space of this problem is small, the best solution can be found by a full search. The next two segments are then requested with decided versions V_{i+1} and V_{i+2}. In practice, after receiving segment i+1, the actual buffer level could be quite different from the estimated value. Currently, if the difference of the actual level B^{cur}_{i+1} and the estimated level $B^e_{i+1,V_{i+1}}$ is larger than ΔB_{err}, the decided value V_{i+2} is canceled and the versions will be re-decided at segment i+1.

Algorithm 1 The adaptation algorithm.

```
1: if (T_i ≤ T_{i−1}) then // downtrend
2:     if B_i^cur ≤ B_min then // dangerous range
3:         V_{i+1} ← 1 // switch to the lowest
4:     else if V_{i+1} was decided and |B_i^cur − B^e_{i,V_i}| ≤ ΔB_err then
5:         Keep using V_{i+1} // which is V_{i+2} in last time
6:     else if B_i^cur > B_low then // high range or safe range
7:         V_{i+1} ← V_i // keep the same version
8:     else // low range
9:         Initiate: V_{i+1} ← 1, V_{i+2} ← 1, QoE^max = 0
10:        for v_1 ← 1, 2, ..., V_i do
11:            for v_2 ← 1, 2, ..., v_1 do
12:                Compute B^e_{i+1,v_1} and B^e_{i+2,v_2} by (2) and (3)
13:                Compute the overall quality QoE_{i+2} by (1)
14:                if {(QoE_{i+2} > QoE^max) and (4) is satisfied} then
15:                    QoE^max ← QoE_{i+2}
16:                    V_{i+1} ← v_1 and V_{i+2} ← v_2
17:                end if
18:            end for
19:        end for
20:    end if
21: else // uptrend
22:    if B_i^cur ≤ B_min then // dangerous range
23:        V_{i+1} ← 1
24:    else
25:        Initiate: V_{i+1} ← V_i, QoE^max ← 0
26:        for v_1 ← V_i, V_i + 1, ..., N do
27:            Compute B^e_{i+1,v_1} by (2)
28:            Compute the overall quality QoE_{i+1} by (1)
29:            if {(QoE_{i+1} > QoE^max) and (B^e_{i+1,v_1} > B_i^cur)} then
30:                QoE^max ← QoE_{i+1}
31:                V_{i+1} ← v_1
32:            end if
33:        end for
34:    end if
35: end if
```

3.3 Uptrend Case

In this case ($T_i > T_{i-1}$), when the buffer level is not in the dangerous buffer range ($B^{cur}_i > B_{low}$), the version will be in general increased to improve the quality. Besides, an important goal of the uptrend process is to recover (or increase) the buffer level, so as preparing for future downtrend situations. So, in this case, the decision problem is stated as follows.

Find $V_{i+1}(V_i \leq V_{i+1} \leq N)$ *that maximizes* QoE_{i+1} *and satisfies*

$$B^e_{i+1,V_{i+1}} > B^{cur}_i. \tag{5}$$

That means, the version will be increased if the future buffer level is expected to be higher than the current buffer level. This is a different point from previous methods, where the version is usually increased only when the buffer level reaches a high level. Finally, the proposed method is summarized in Algorithm 1.

4. Experimental Results

In this section, the proposed method is evaluated in the context of on-demand streaming where $B_{max}$is set to 40s. For comparison, three reference methods are the segment aware rate adaptation (SARA) method [3], the trend of buffer level variation (TBLV) method [4], and the local average bitrate (LAB) method [5]. The streaming testbed is similar to that of [2], [5], which consists of a server, an IP network, and a client, where DummyNet tool is used to emulate throughput variations using bandwidth traces. RTT value is set to 40ms. The test video, which is Big Buck Bunny [8], is encoded by H.264 format into 9 versions with QPs being 24, 26, 28, 32, 36, 40, 44, 48, 52. The duration τ of all segments is 2 seconds. Two bandwidth traces of 4 minutes from a mobile network [10] are employed (Figs. 2 and 3). The quality model of the video was obtained in [8] and included in metadata. In our method, TBLV method, and SARA method, thresholds (B_{min}, B_{low}, B_{high}) are (10s, 20s, 30s). For the LAB method, only B_{min} is needed and set to 10s.

Figure 2 shows the adaptation results in terms of overall quality (MOS), version index, and buffer level of the methods with bandwidth trace #1. Note that the version and the buffer level are specific to each segment, while the overall quality is cumulative until a given segment.

As seen in Fig. 2, our method consistently provides the highest quality. The only exception is at 25s, where the TBLV method tries to maintain the current version for a long time. Meanwhile our method can estimate the future buffer level and decide to decrease the version earlier. Because of this behavior, the buffer level of the TBLV method, of which the minimum level is very low (e.g. at 180s), is less stable than that of our method. Thanks to buffer level estimation, our method can effectively take advantage of the buffer to improve the quality while the resulting buffer level is mostly above threshold B_{min}.

The SARA method always tries to increase the version as much as possible, which causes frequent switching up (and then switching down). This behavior results in low quality and very low buffer level. The LAB method provides a reasonable quality and good buffer stability. It is

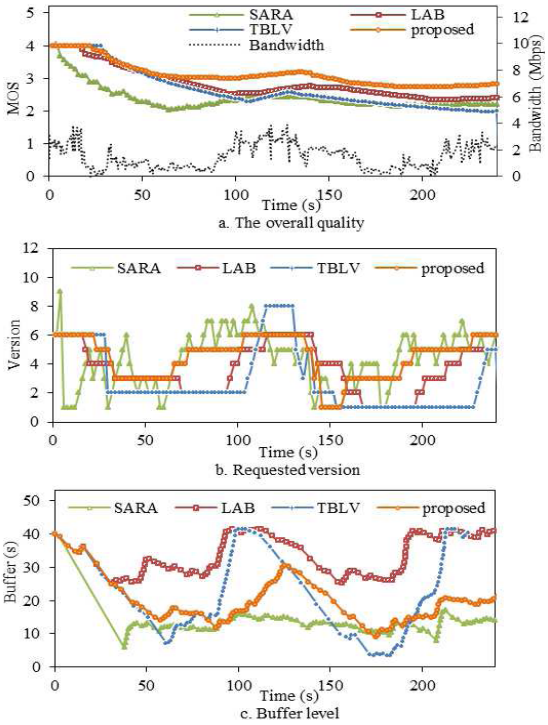

Fig. 2 Adaptation results with bandwidth trace #1.

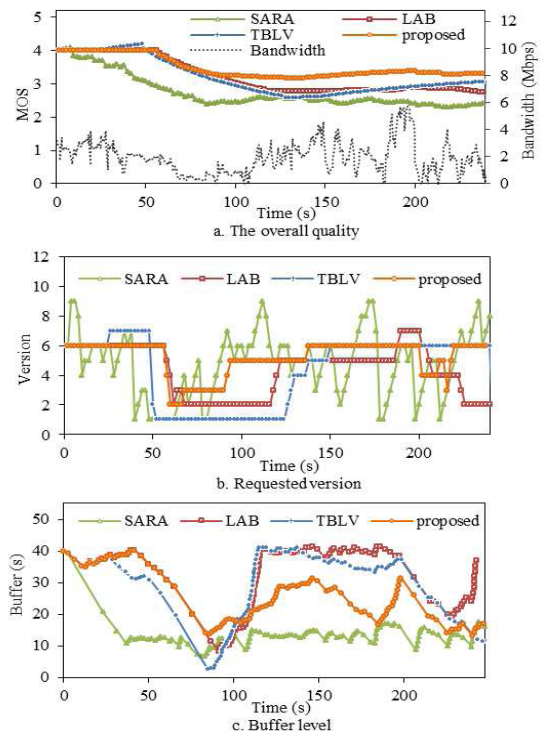

Fig. 3 Adaptation results with bandwidth trace #2.

interesting that, at the end of the session, the quality value of the TBLV method is a little lower than that of the SARA method. However, due to their behaviors, their quality values may be significantly different during the session. For example, if the session is finished at time 60s, the SARA method is much worse than the TBLV method. This is an important advantage of quality models, which provides insights into the overall quality at each time instant, not just at the end of a session.

Figure 3 shows the results with bandwidth trace #2, where similar behaviors of the adaptation methods can be seen. The SARA method has the lowest quality while our proposed method is in general the best one. Again, the exception is at time 40s, where the TBLV method tries to request version 7. However, this results in low buffer level, and then in a drastic switch to version 1. So, by using the quality model and buffer level estimation, the proposed method can provide consistently high quality and good buffer stability.

5. Conclusions

In this study, we have presented a new adaptation method, which for the first time made use of a quality model in HTTP adaptive streaming. The proposed adaptation method also estimated the buffer level in the near future to avoid buffer underflows. Through experiments with strongly fluctuating bandwidths, we showed that the proposed method provided high and consistent video quality and good buffer stability.

References

[1] T.C. Thang, Q.-D. Ho, J.W. Kang, and A.T. Pham, "Adaptive streaming of audiovisual content using MPEG DASH," IEEE Trans. Cons. Electron., vol.58, no.1, pp.78–85, Feb. 2012.
[2] T.C. Thang, H.T. Le, A.T. Pham, and Y.M. Ro "An Evaluation of Bitrate Adaption Methods for HTTP Live Streaming," IEEE J. Selected Areas in Comm., vol.32, no.4, pp.693–705, April 2014.
[3] P. Juluri, V. Tamarapalli, and D. Medhi, "SARA: Segment Aware Rate Adaptation algorithm for Dynamic Adaptive Streaming over HTTP," Proc. IEEE ICC QoE-FI Workshop, pp.1765–1770, June 2015.
[4] Y. Zhou, Y. Duan, J. Sun, and Z. Guo, "Towards Simple and Smooth Rate Adaption for VBR Video in DASH," Proc. IEEE Visual Comm. Image Processing Conf., pp.9–12, Dec. 2014.
[5] H.T. Le, H.N. Nguyen, N.P. Ngoc, A.T. Pham, and T.C. Thang, "A Novel Adaptation Method for HTTP Streaming of VBR Videos over Mobile Networks," Mobile Information Systems, vol.2016, Article ID 2920850, pp.1–11, 2016.
[6] Z. Guo, Y. Wang, and X. Zhu, "Assessing the Visual Effect of Non-periodic Temporal Variation of Quantization Stepwise in Compressed Video," Proc. IEEE ICIP, pp.3121–3125, 2015.
[7] J.D. Vriendt, D.D. Vleeschauwer, and D.C. Robinson, "QoE model for video delivered over an LTE network using HTTP adaptive streaming," Bell Labs Techn. J., vol.18, no.4, pp.45–62, March 2014.
[8] H.T.T. Tran, T. Vu, N.P. Ngoc, and T.C. Thang, "A novel quality model for HTTP Adaptive Streaming," Proc. IEEE Conf. Comm. and Electronics (ICCE), pp.423–428, July 2016.
[9] Y.-F. Ou, Y. Xue, and Y. Wang, "Q-STAR: A Perceptual Video Quality Model Considering Impact of Spatial, Temporal, and Amplitude Resolutions," IEEE Trans. Image Processing, vol.23, no.6, pp.2473–2486, June 2014.
[10] C. Müeller, S. Lederer, and C. Timmerer, "An Evaluation of Dynamic Adaptive Streaming over HTTP in Vehicular Environments," Proc. ACM SIGMM Workshop on Mobile Video, pp.37–42, 2012.