QoE-Aware Video Streaming over HTTP and Software Defined Networking

Pham Hong Thinh[1,2], Nguyen Thanh Dat[1], Pham Ngoc Nam[3], Nguyen Huu Thanh[1], and Truong Thu Huong[1(✉)]

[1] Hanoi University of Science and Technology, Hanoi, Vietnam
huong.truongthu@hust.edu.vn
[2] Quy Nhon University, Binhdinh, Vietnam
[3] Vin University, Hanoi, Vietnam

Abstract. Due to the increase in video streaming traffic over the Internet, more innovative methods are in demand for improving both Quality of Experience (QoE) of users and Quality of Service (QoS) of providers. In recent years, HTTP Adaptive Streaming (HAS) has received significant attention from both industry and academia based on its impacts in the enhancement of media streaming services. However, HAS-alone cannot guarantee a seamless viewing experience, since this highly relies on the Network Operators' infrastructure and evolving network conditions. Along with the development of future Internet infrastructure, Software-Defined Networking (SDN) has been researched and newly implemented as a promising solution in improving services of different Internet layers. In order to enhance quality of video delivery, we try to combine the above two technologies, which has not been well-studied in academia. In this paper, we present a novel architecture incorporating bitrate adaptation and dynamic route allocation. At the client side, adaptation logic of VBR videos streaming is built based on the MPEG-DASH standard. On the network side, a SDN controller is implemented with several routing strategies on top of the OpenFlow protocol. Our experimental results show that the proposed solution enhances at least 38% up to 185% in term of average bitrate in comparison with some existing solutions as well as achieves smoother viewing experience than the traditional Internet.

Keywords: Dynamic routing · Adaptive streaming · SDN · DASH

1 Introduction

The last decade has witnessed a tremendous escalation of media content consumption, especially high-definition videos over the Internet. Cisco forecasts that the global Internet traffic in 2021 will equivalent to 127 times of that of the year 2005. In 2017, among the services over the Internet such as web, email, file sharing, etc., video streaming takes part in more than 74% of the global Internet traffic and will continue to rise over 81% by 2021 [1]. Those numbers figures

T. Q. Duong et al. (Eds.): INISCOM 2019, LNICST 293, pp. 40–52, 2019.
https://doi.org/10.1007/978-3-030-30149-1_4

show the necessity of developing methods in optimization of video streaming over the Internet that satisfy both efficiency for service providers and the quality for users. In that context, one technology has become the de facto standard for Internet streaming: HTTP-Based Adaptive Streaming (HAS) [2]. Using HTTP HAS is leveraging an ubiquitous and highly optimized delivery infrastructure, originally created for the web traffic, which includes, e.g., Content Delivery Networks (CDNs), caches, and proxies.

One of the enablers of the success of HAS was the open standard MPEG-DASH (Dynamic Adaptive Streaming over HTTP) [2,3]. The fundamental principle of DASH is encoding video content into multiple versions at different discrete bitrates. And clients can request segment's version actively which avoids the overhead computation at the server when numerous clients connect to the server simultaneously. Rate adaptation algorithms of HAS clients were often categorized into three categories, namely throughput based, buffer based and hybrid method throughput-buffer based.

From another aspect, Software-Defined Networking [4] is a new network architecture, that centralizes network intelligence in one network component by disassociating the forwarding process of network packets (data plane) from the routing process (control plane). In SDN, the common logical architecture in all switches, routers, and other network devices are managed by an SDN controller that allows networks policies be dynamically designed to support each single specific application.

In this study, we develop new methods to unite the advantages of dynamic adaptive streaming over HTTP and Software-Defined Networking. At the client side, the proposed adaptation algorithm based on DASH that frequently requires quality levels of video segments (a bitrate level of video segment, a high quality level means a high video quality) up or down depending on the bandwidth and client's buffer status. Within the transportation network, in order to improve the bandwidth received at client, two routing policies are proposed, called periodically routing and on-demand routing. Experimental results prove that our approach out-performs the existing state-of-the-art streaming solutions.

The remainder of this paper is organized as follows. Section 2 provides the related work on several existing adaptation method and bandwidth allocation schemes. Section 3 introduces our bitrate adaptation algorithm and SDN-based dynamic routing solutions. The experiment setup and performance evaluation of the proposed solution is also presented in this section. Conclusion and possible future extensions are presented in the last section.

2 Related Work

The DASH standard is designed to cope with highly varying delivery conditions. Over the past few years, many bitrate adaptation algorithms have been introduced in order to improve user's Quality of Experience (QoE). Their difference is mainly the required input information, ranging from network characteristics to application-layer parameters such as the playback buffer or the download speed.

In DASH, all algorithms decide the bitrate of next download segment based on the throughput variation or buffer level at the client side. These algorithms can be roughly divided into three main types such as the throughput-based group [5,6]; the buffer-based group [7,8] and the mixed type of rate adaptation algorithms, [9,10].

In throughput-based methods, bitrate is decided based on the estimated throughput without considering buffer. These schemes mainly aim at dynamically adapting a video bitrate to an available bandwidth, which usually leads to a low bandwidth utilization and cannot reach the maximum quality allowed by the available bandwidth. This is because, in such schemes, video bitrates higher than available bandwidth are never allowed to be selected to avoid playback interruptions. The key differences between these methods are the ways to estimate and use the throughput. As discussed in [5,6], authors proposed a rate adaptation algorithm based on the estimated throughput (called aggressive method) where the last segment throughput is simply used as the estimated throughput. It is currently the most responsive method to capture the dynamic changes of throughput. In the aggressive method, the bitrate is decided as the highest bitrate that is lower than the estimated throughput.

Buffer-based schemes basically employ buffer thresholds to decide the changes of bitrate. Compared to the throughput-based methods, buffer-based methods provide smoother video bitrate curves in on-demand streaming. Usually, during a certain range of buffer level, a client will try to maintain the current bitrate, resulting in a stable bitrate curve and a rather unstable buffer level. However, when bandwidth is drastically reduced, the buffer-based methods may cause sudden change of bitrate, still. This is mainly because there is a trade-off between the stability of buffer occupancy and the smoothness of video bitrate due to the time-varying bandwidth. In [7,8] Huang et al. proposed a class of buffer-based bitrate adaptation algorithm for HTTP video streaming, called BBA, that is based only on the current playback buffer occupancy as bitrate is selected heuristically without throughput estimation. The objective of BBA is to maximize the average video quality by selecting the available highest bitrate level that the network can support and avoid stalling events.

There are some ABR (Adaptive Bitrate Selection) algorithms in the literature that consider also the path bandwidth combining with the current buffer occupancy to select the most suitable video version for the next segment. In, [9], the authors proposed a rate adaptation algorithm for VBR video which based on buffer thresholds to request the next video quality level. The authors in [10] proposed a segment-aware rate adaptation (SARA) algorithm that considers the segment size variation in addition to the estimated path bandwidth and the current buffer occupancy to accurately predict the time required to download the next segment. This ensures that the best possible representation of the video is downloaded while avoiding video buer starvation.

Overall, current adaptation algorithms for HTTP Adaptive Streaming adjust the quality version of video segments to achieve the highest bandwidth utilization, smooth playback as well as avoid buffer underflow or overflow. However, the algorithms almost focus on improving the adaptation policy at the client side without considering the available resources in the networks.

There are some studies proposed in the literature for HAS over SDN. In [11], the authors proposed an adaptive HTTP video streaming framework utilizing the flow route selection capability of SDN networks. In this method the SDN controller reroutes DASH flows in each segment period after the client has completely downloaded. This leads to the controller being overloaded when great number of clients accesses at the same time or when network load increases rapidly. In [12], the authors proposed an SDN architecture to monitor network conditions of streaming ow in real time and dynamically change routing paths using multi-protocol label switching to provide reliable video watching experience. In SDNHAS [13], Bentaleb et al. relies on an SDN-based management and resource allocation architecture with the goal to estimates optimal QoE policies for groups of users and requests a bandwidth constraint slice allocation, while providing encoding recommendations to HAS players. However, these studies only present the general adaptation mechanism.

3 Problem Formulation

In this section, we propose a HTTP adaptive streaming solution included adaptation algorithm at client and network routing policies for video contents over Software-Defined Networking platform.

3.1 Network Context of HTTP Streaming over SDN

As Fig. 1. depicts, HTTP video streaming is carried out over an Openflow/SDN transportation network where Openflow switches are controlled by the SDN controller. In our design, this controller can reinforce routing policies based on

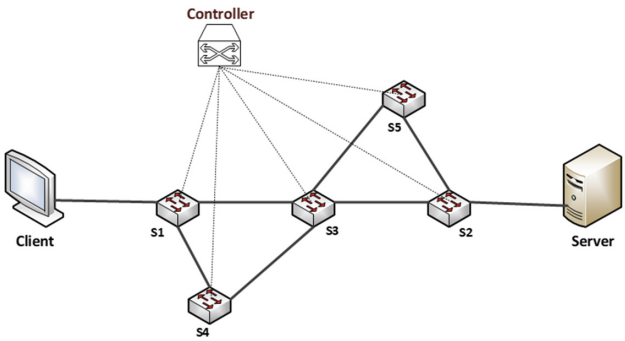

Fig. 1. HTTP streaming over SDN

network conditions, while at the client side, a bitrate adaptation algorithm can be implemented in order to acquire a seamless playback as well as a best possible video quality.

3.2 Adaptation Problems

Many adaptation algorithms have been proposed to attain a good experience in poor bandwidth conditions (highly variable throughput or low bandwidth etc.). With a proper method, a client can avoid video freezing caused by sudden severe bandwidth drop and achieve an acceptable video's quality. Before presenting the proposal, three adaptation algorithms corresponding to the three well known mechanism as mentioned before will be introduced first. Table 1 shows the symbols used in this study.

Table 1. Symbols using in the paper.

Symbol	Description
i	Segment index
j	Version index
q	The number of quality level
B_i	The buffer level at segment i
I_i	The representation's index of segment i
D_i	The download rate of segment i
R_j	The bitrate of version j
B^e_{i+1}	The estimate buffer for segment $i+1$
B^{Th}	The buffer threshold
D^{Th}	The bitrate threshold
T_i	The measured throughput at segment i
T^e_{i+1}	The estimate throughput for segment $i+1$
RTT	Round Trip Time
SD	Segment Duration

The authors in [5,6] proposed the Aggressive algorithm based on throughput without considering the buffer's condition. The next throughput T^e_{i+1} is assigned equal to the current throughput and the highest possible value of segment bitrate R_j is computed by the estimated throughput and a safety margin μ as in Eq. 1 with μ is range from 0 to 0.5.

$$I_{i+1} = \arg\max_{\mathbf{j}} \left\{ R_j \mid R_j \leq (1-\mu) \times T^e_{i+1} \right\} \tag{1}$$

BBA is a very well-known buffer-based adaptation algorithm. According to BBA [7,8], the average segment size of each corresponding bitrate is mapped to an instantaneous buffer level, in a linear manner with two fixed points for the lowest and highest bit-rate. BBA is too conservative during startup. The network can sustain a much higher video rate, but the algorithm is just not aware of it yet. There is a trade-off between buffer occupancy and video quality in BBA.

The Segment Aware Rate Adaptation - SARA [10] estimated next weighted Harmonic Mean throughput based on all measured throughputs in the past as Eq. 2, where w_i and d_i are the volume and the download rate of segment number i, respectively. SARA selects the most suitable representation for the next segment to be downloaded based on H_n the buffer occupancy at any given time, B_{curr}. The rate adaptation is done in four stages including fast start, addictive increase, aggressive switching, delayed download corresponding to B_{curr} value.

$$H_n = \frac{\sum_{i=1}^{n} w_i}{\sum_{i=1}^{n} \frac{w_i}{d_i}} \tag{2}$$

3.3 Our Proposed Adaptation Algorithm – MUNTH (iMpede sUspend and AttaiN PoTential PatH)

In our solution, next throughput is estimated based on two previous measured throughput at client as described in Eq. 3 [9], where γ is a dynamic constant range from [0,1]. This enables a client to adapt well with the highly bandwidth fluctuations especially with highly variable bandwidth which is the main cause of video freezing. The γ is usually set to be 0.5, when $\gamma = 1.0$, and the estimation is exactly the same as aggressive method.

$$T^e_{i+1} = \gamma \times T_i + (1 - \gamma) \times T_{i-1} \tag{3}$$

To avoid video freezing, we propose a new quality selection mechanism based on the estimate buffer level. Equation 4 [14] is used to estimate the next buffer level if client selects quality level j for Segment number i where SD is segment duration. Our objective is to choose a suitable quality level to keep the buffer greater than a threshold B^{Th} to prevent stalling events. The details of our method are shown in Algorithm 1.

$$B^e_{i+1} = B_i + SD - RTT - \frac{SD \times R_i}{T^e_{i+1}} \tag{4}$$

Algorithm 1. Bitrate Adaptation Algorithm - MUNTH

Input: T_i, R_n, B_i,B^{Th}, D_i, RTT, D^{Th}, SD
Output: I_{i+1}

$T^e_{i+1} \leftarrow \gamma \times T_i + (1-\gamma) \times T_{i-1}$; // Estimate throughput.
$I_{i+1} \leftarrow 0$
if $D_i \leq D^{Th}$ **then**
  Request for a new path;
else
  for $j \leftarrow q-1$ **to** 0 **do**
    $B^e_{i+1} \leftarrow B_i + SD - RTT - \frac{SD \times R_j}{T^e_{i+1}}$; // Estimate next buffer level.
    if $B^e_{i+1} \geq B^{Th}$ **then**
      $I_{i+1} = j$;
    end
  end
end

In the proposed algorithm, when the download rate D_i is smaller than $D^{Th} = 1000$ kbps, client is going to send a message to the network controller to request an optimal path which is sufficiency to transport video data. The detail policies at network controller will be discussed in the next subsection.

3.4 Routing Policies

With the OpenFlow/SDN-based architecture [15] which can control and manage all types of data flows in the control layer, the SDN platform provides ability to flexibly perform any routing rules in the network. Furthermore, the centralized scheme of SDN has the ability to entirely monitor the network topology and routing status, and timely modify the path selection according to the changes of network states.

Periodical Routing. We proposed a periodical routing mechanism to select the optimal path every T seconds based on the stability and availability of each path. Every path from a client to the server will be represented by $Repre^p$ as shown in Eq. 5.

$$Repre^p = (1 - \omega^p) \times BW^p_{inst} \tag{5}$$

$Repre^p$ is calculated from the stability w^p and the availability BW^p_{inst} which are:

BW^p_{inst}: The instant bandwidth of each path measured by the SDN controller, p represents for path number p in n available paths (indexed from 0 to n−1) from a client to the server. If a path includes multiple links, BW^p_{inst} is assigned to the bandwidth of the bottleneck link.

w^p: The stability of a path p, calculated by the Eq. 6. Where: BW_i^p is the measured bandwidth on path p at $i \times T$ seconds before. w^p is in range [0,1] and the higher w^p is, the less stable path p is.

$$\omega^p = \frac{\sqrt{\frac{\sum_1^m (BW_i^p - \overline{BW^p})^2}{m}}}{\sum_{p=0}^{n-1} \sqrt{\frac{\sum_1^m (BW_i^p - \overline{BW^p})^2}{m}}} \tag{6}$$

The Controller selects the most stable and available path which has the highest $Repre^p$ as shown in Eq. 7 where $Indx$ is the selected path index.

$$Indx = indexOf(max_{p=\overline{0..n-1}}\{Repre^p\}) \tag{7}$$

Adaptive Routing - MUNTH. We propose an active mechanism from client namely MUNTH to request for the optimal path when the current bandwidth is not satisfying their demand. A client can send a 'reroute' message to the controller when download rate is lower than 1000 kbps and the controller then selects the optimal path having highest BW_{inst}^p to serve the stream as shown in Eq. 8. the controller only has to seek a new path when poor network conditions occur otherwise the path will be kept during the stream session. This mechanism has two advantages: firstly, it reduces computation at the controller compared to the periodical routing scheme. Secondly, a client knows best about its perceived network condition as well as its ability therefore giving client an ability to control the adaptive rates is meaningful.

$$Indx = indexOf(max_{p=\overline{0..n-1}}\{BW_{inst}^p\}) \tag{8}$$

3.5 Experiment Setting

In this section we present the experiment setting as well as evaluate the performance of our proposed scheme. The setup video streaming system includes three parts: DASH client, SDN network and Video Server. The client is installed by libdash library [16] on linux to serve the DASH standards. Client's media player is implemented by Qtsampleplayer [16] with buffer size of 50 s. The emulated server stores a VBR video clip named "Elephants Dream" [17] with length of 10 min 52 s, 2 s segment, 24 fps and Full-HD resolution (1920×1080). Every segment is encoded to 12 different versions corresponding to 12 QP values (13 to 46) of the H264 standard. In this paper, safety margin μ is set equal to 0.1. We test our solution in two experiments as follows:

Experiment 1: Buffer threshold (B^{Th}) optimization

- Client runs the MUNTH adaptation algorithm with B^{Th} of 10 s, 15 s, 20 s, 25 s with the adaptive routing policy at the controller ($m = 5$)

Experiment 2: To evaluate the performance of MUNTH, we investigate the related existing schemes by setting up as follows:

- Scheme "AGG_DF": Client runs the aggressive algorithm, the controller is active with the conventional routing policy (e.g. Dijkstra algorithm).
- Scheme "SARA": Client runs SARA algorithm controller have the same configuration as scheme "AGG_DF".
- Scheme "BBA": Client runs BBA algorithm (BBA parameter), controller have the same configuration as scheme "AGG_DF".
- Scheme "AGG_RR": Client runs Aggressive algorithm, controller now uses periodical routing policy.
- And our scheme "MUNTH": Client runs MUNTH algorithm with the optimal B^{Th} selected from Experiment 1.

3.6 Performance Evaluation

***B^{Th}* Optimization for MUNTH.** The Table 2 shows the quality metric resulted from MUNTH. As we can see, with the lowest buffer thershold, with $B^{Th} = 10$ s the stalling duration gets highest at 43.1 s. It also has the highest rate of buffer level lower than 10 s as well as lowest average buffer. In contrast, experiment with $B^{Th} = 20$ s shows the best results. Stalling duration is zero, percentage of buffer $\leq$10 s is only 1.83% while keeping average buffer in a comparable level than $B^{Th} = 25$ s. However, a higher B^{Th} can harm the video quality because clients concentrate in avoiding stalling events more than video representation. As we can see, when $B^{Th} = 25$ s both average bitrate and percentage of bitrate $\geq$8000 kbps is smaller than the rest.

Table 2. MUNTH performance with different B^{Th} values

Criteria	$B^{Th} = 10$ s	$B^{Th} = 15$ s	$B^{Th} = 20$ s	$B^{Th} = 25$ s
Stalling event	6	2	0	3
Stalling duration	43.1	38.9	0	21.45
Percentage buffer $\leq$ **10 s (%)**	21.95	6.4	1.83	4.27
Average birate	10613.64	10496.54	10764.62	9575.02
Bitrate $\geq$ **8000 kbps**	46.34	44.21	46.34	40.55
Switching down version	42	33	39	44

In conclusion, the experiment results show that our MUNTH algorithm with $B^{Th} = 20$ s enables a client to occupy enough buffer to avoid freezing event while achieving a comparable video quality compared with the others. The optimal value for B^{Th} was chosen to be 20 s.

Performance Comparison. In this part, the MUNTH algorithm with optimized $B^{Th} = 20$ s will be compared with the other algorithms as listed in Experiment 2.

Figure 2 shows the occupied buffer for 5 different schemes respectively. The periodical routing mechanism enables AGG_RR to delivery segments on a stable path and a client can achieve a smoother streaming compared to AGG_DF. The result for AGG_RR shows that in only the duration from segment 50 to 100 stalling events occurs. MUNTH successfully avoids stalling events (0 times) by smooth throughput estimation and selection of suitable video rate to preserve buffer on an optimal path.

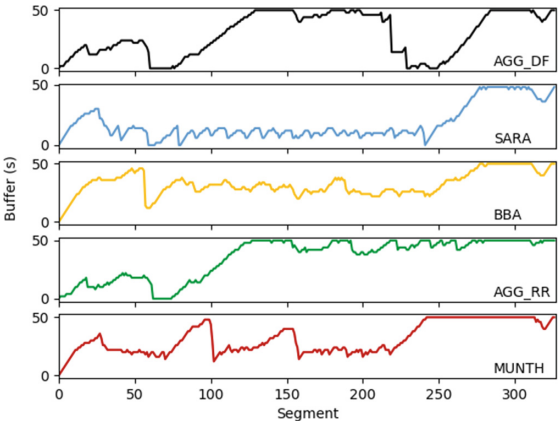

Fig. 2. Buffer level of all methods

Figure 3 shows the Cumulative Distribution Function (CDF) of bitrate during the playing time. The first thing to notice from the bitrate CDF figure is that the percentage of segments downloaded at the high bitrate of MUNTH is greater than the rest. Specifically, around 40% of MUNTH's segments has bitrate greater than 10000 kbps. The CDF value at 10000 kbps for AGG_DF and SARA is lower about 20% while the figure for BBA is lowest at 10%. The periodical routing policy of AGG_RR enable about 30% of segments are downloaded with bitrate rate $\geq$ 10000 kbps.

Table 3 shows the precise numerical metrics accumulated during our test. From the table, we can see that AGG_DF has the average quality level compared to the other methods while having the highest freeze frequency and freeze duration. SARA is the mixed throughput - buffer algorithm which can reduce stalling events compared to AGG_DF but the video's average bitrate is slightly lower. BBA successfully avoids stalling events. However, the average bitrate (in kpbs) of BBA is lowest at only 3754.68 kbps. By using the periodical rerouting policy, AGG_RR acquires a higher bitrate than AGG_DF and can avoid freezing by choosing the most stable path every T second. MUNTH shows a superior result compared to the other methods. Firstly, buffer statistics show that MUNTH can avoid stalling events while keeping a high buffer occupancy (Avg Buffer = 35.24 s). Secondly, by the adaptive routing policy, MUNTH achieves the

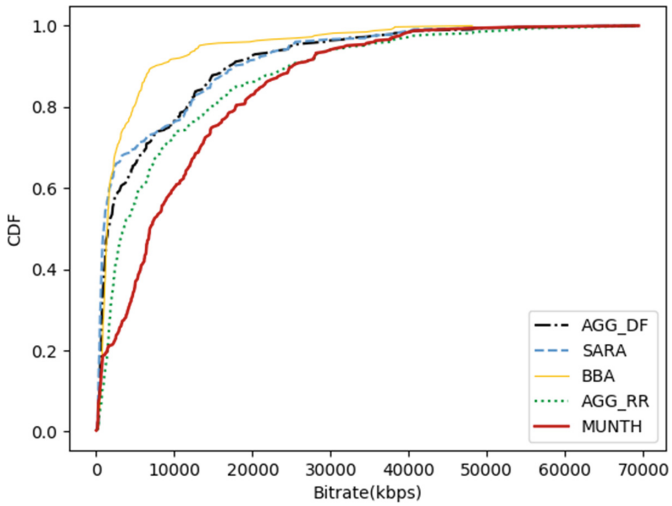

Fig. 3. Cumulative distribution function of bitrate

highest Average bitrate at 10734.18 kbps - an enhancement about 69.5%, 80.3%, 185.9%, 21.79% compared to AGG_DF, SARA, BBA and AGG_RR respectively. Another quality metric is the number of switching-down version event, users are sensitive with the changing in the video quality, especially with suddenly drop of bitrate versions. As we can see, MUNTH obtains the comparable switch-down version with BBA and SARA. Whereas achieving such better performance, MUNTH consumes much less computation at the controller compared with AGG_RR by a politic switching path mechanism.

Table 3. Results' statistics of the methods

Criteria	AGG_DF	SARA	BBA	AGG_RR	MUNTH
Stalling duration	125.11	75.32	0	81.3	0
Number of stalling events	26	8	0	13	0
Average buffer (s)	29.27	17.61	32.55	35.24	31.50
Average bitrate (kbps)	6331.33	5953.73	3754.68	8800.26	10734.18
Number of version switch-downs	79	21	25	105	39
Switch path	N/A	N/A	N/A	23	2

4 Conclusion

In this article we proposed a combined solution both from the client and network perspective to enhance users' experience while using HTTP Adaptive Streaming applications over SDN network. From the client side, our proposed adaptation

algorithm - MUNTH uses smooth throughput estimation and buffer occupancy estimation mechanism to select a segment representation which helps the client to deal with bandwidth fluctuation therefore able to request a suitable bitrate version without harming buffer level leading to stalling event. From the network side, we proposed two routing policies, periodical routing and adaptive routing – MUNTH. Two metrics named stability w^p and availability BW^p_{inst} of a path are used to select the optimal path from the client to the sever in periodical routing manner. MUNTH routing policy is a client active scheme, where the client can actively request a new path that best satisfies its requirement. The experiment results show that our proposals is superior to the predecessors.

For future work, we will solve a problem where topology is more complicated and multiple clients connect to multiple servers. Therefore, the problem is not only about rate adaptation and path selection but also about fairness, stability and resource utilization among clients.

Acknowledgement. This work was supported by the University project grant, ID [T2018-PC-065]. The grant is funded by Hanoi University of Science and Technology.

References

1. Cisco Visual Networking Index: Forecast and methodology, 2016–2021, white paper, San Jose, CA, USA 1 (2016). https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/complete-white-paper-c11-481360.html
2. Stockhammer, T.: Dynamic adaptive streaming over HTTP-: standards and design principles, pp. 133–144 (2011)
3. Sodagar, I.: The MPEG-DASH standard for multimedia streaming over the internet. IEEE Multimedia **18**(4), 62–67 (2011)
4. Open networking Foundation: Open networking Foundation. https://www.opennetworking.org/. Accessed 14 Apr 2019
5. Romero, L.R.: A dynamic adaptive HTTP streaming video service for Google android. Master's Degree Project, The Royal Institute of Technology (2011)
6. Thang, T.C., Ho, Q.D., Kang, J.W., Pham, A.T.: Adaptive streaming of audiovisual content using MPEG DASH. IEEE Trans. Consum. Electron. **58**(1), 78–85 (2012)
7. Huang, T.Y., Johari, R., McKeown, N., Trunnell, M., Watson, M.: Using the buffer to avoid rebuffers: evidence from a large video streaming service. arXiv preprint arXiv:1401.2209 (2014)
8. Huang, T.Y., Johari, R., McKeown, N., Trunnell, M., Watson, M.: A buffer-based approach to rate adaptation: evidence from a large video streaming service. ACM SIGCOMM Comput. Commun. Rev. **44**(4), 187–198 (2015)
9. Nguyen, H.N., Vu, T., Le, H.T., Ngoc, N.P., Thang, T.C.: Smooth quality adaptation method for VBR video streaming over HTTP. In: 2015 International Conference on Communications, Management and Telecommunications (ComManTel), pp. 184–188. IEEE (2015)
10. Juluri, P., Tamarapalli, V., Medhi, D.: Sara: segment aware rate adaptation algorithm for dynamic adaptive streaming over HTTP, pp. 1765–1770 (2015)

11. Cetinkaya C, K.E.: SDN for segment based flow routing of dash. In: 2014 IEEE Fourth International Conference on Consumer Electronics Berlin (ICCE-Berlin), pp. 74–77 (2014)
12. Nam, H., Kim, K.H., Kim, J.Y., Schulzrinne, H.: Towards QoE-aware video streaming using SDN. In: 2014 IEEE Global Communications Conference, pp. 1317–1322. IEEE (2014)
13. Bentaleb, A., Begen, A.C., Zimmermann, R.: SDNDASH: improving QoE of HTTP adaptive streaming using software defined networking. In: Proceedings of the 24th ACM International Conference on Multimedia, pp. 1296–1305. ACM (2016)
14. Vu, T., Le, H.T., Nguyen, D.V., Ngoc, N.P., Thang, T.C.: Future buffer based adaptation for VBR video streaming over HTTP. In: 2015 IEEE 17th International Workshop on Multimedia Signal Processing (MMSP), pp. 1–5. IEEE (2015)
15. Owens II, H., Durresi, A.: Video over software-defined networking (VSDN). Comput. Netw. **92**, 341–356 (2015)
16. Bitmovin: Libdash - bitmovin. https://github.com/bitmovin/libdash. Accessed 14 Apr 2019
17. Orange Open Movie Project Studio: Elephants dream (2009). https://orange.blender.org/. Accessed 3 Dec 2018