Towards QoE-aware Video Streaming using SDN

Hyunwoo Nam*, Kyung-Hwa Kim†, Jong Yul Kim† and Henning Schulzrinne†
*Department of Electrical Engineering, Columbia University, New York, NY
†Department of Computer Science, Columbia University, New York, NY

***Abstract*—Today's over the top (OTT) video service providers take advantage of content distribution networks (CDNs) and adaptive bitrate (ABR) streaming where a video player adjusts resolutions based on end-to-end network conditions. Although the mechanisms are useful to improve user-perceived video quality, they do not resolve the root causes of congestion problems. To pinpoint a bottleneck and improve video quality-of-experience (QoE), we leverage a software-defined networking (SDN) platform from OTT video service provider's point of view. Our proposed SDN application is designed to monitor network conditions of streaming flow in real time and dynamically change routing paths using multi-protocol label switching (MPLS) traffic engineering (TE) to provide reliable video watching experience. We use an off-the-shelf SDN platform to show the feasibility of our approaches.**

Keywords*—*Video Streaming, Mobile Wireless, HTTP Progressive Video, Software-Defined Networking, Over The Top Applications, Multi-Protocol Label Switching Traffic Engineering

I. Introduction

Video service providers use CDNs to speed up the delivery of their contents to clients. In a CDN, a geographically close delivery node is typically assigned to a client. However, the network conditions can be unstable although the delivery node is located near the client [1]. Furthermore, once a client is connected to a CDN operated by a third party, there is no method for service providers to track user-perceived video quality. To improve video QoE, they rely on client-side rate selection algorithms such as ABR streaming. Although it can be useful to automatically adjust resolutions based on the network condition, this client-side mechanism is not helpful in discovering the bottleneck that causes the video quality problem.

To mitigate these issues, we propose to implement an SDN-based video streaming architecture. In this paper, we try to solve the problem from the perspective of video service providers where they have full control over their servers, networks and video players but have limited control over access networks of clients. In order to improve video QoE over SDN, we propose to *a*) measure various video QoE metrics at video players running on clients' devices; *b*) assign the best available delivery node based on provisioned network conditions; and *c*) dynamically change routing paths between wide area network (WAN) routers using MPLS-TE.

We implement our SDN solutions using Junos Space SDK [2] that is designed to monitor and control networking devices of Juniper Networks. In our testbed, we have created a light-weight plugin in an HTML5 video player to monitor various QoE factors (e.g., buffering status and video resolution selected by a client) to analyze user-perceived experience when a video is playing. When buffering events occur, our SDN application allows service providers to send queries to the SDN controller to analyze network conditions (e.g., TCP throughput and packet loss rate) on the streaming flow and change the routing path in the network. Our WAN traffic monitoring system is designed to communicate with the SDN controller using RESTful APIs to visualize the network information in real time. A demo can be found on our web page[1].

The remainder of the paper is organized as follows. In the second section, we look at related work. In Section III, we address problems on existing OTT video delivery systems. Our proposed SDN platform is described in Section IV. We explain our implementation in Section V and evaluate our solution in Section VI. Finally, we summarize our conclusions in Section VII.

II. Related Work

Several researchers have investigated an application-aware SDN platform and WAN routing control using SDN. Zafar et al. [3] focus on a mobile application detection framework. They use a machine learning based traffic classification technique and a crowd-sourcing approach to identify the application types in SDN. Ali et al. [4] have introduced MPLS-TE and MPLS VPNs with OpenFlow. They have shown a demo where MPLS control plane features are simply implemented on an SDN platform. Saurav et al. [5] have demonstrated application-aware aggregation and traffic engineering in a packet circuit network. Using a NOX SDN controller [6], they dynamically controlled packet flows based on different application types. Michael et al. [7] have shown SDN-based application-aware networking for YouTube video streaming. They have conducted a performance test of several path selection mechanisms such as round-robin, bandwidth-based, deep packet inspection-based and application-aware in an SDN-enabled network. Aricent group [8] has introduced application-aware routing with SDN as a business model. They address that SDN-based routing control allows service providers to lower CAPEX / OPEX and improve the overall end-user experience.

Our approach differs from the prior work in two aspects: *a*) Noticeably, we focus more on a video streaming application in SDN. Unlike other approaches where general application-aware SDN platform has been introduced, we have designed our solution from the perspective of video service provider. *b*) Instead of using emulators (e.g., Mininet [9]), we have implemented our solutions using a commercial off-the-shelf SDN platform such as Junos Space [2] to show the feasibility of our approaches.

[1] http://dyswis.cs.columbia.edu/sdn

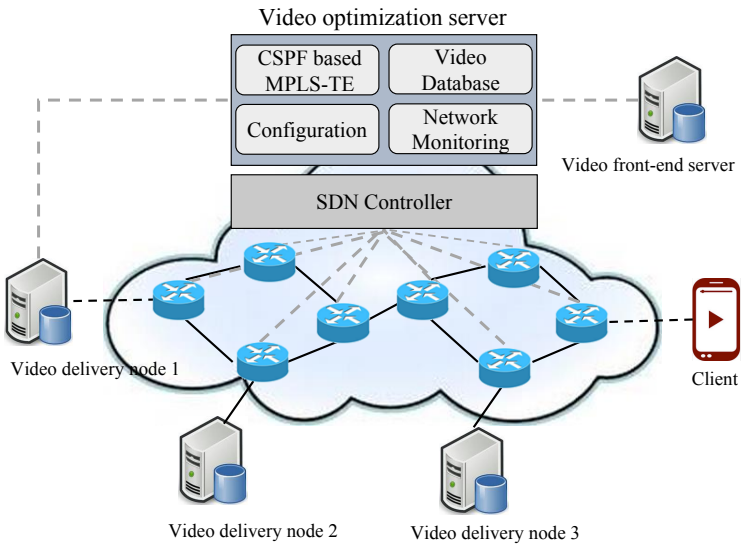


Figure 1: Video QoE-aware streaming platform using SDN

III. Problems on Existing OTT Video Delivery System

OTT video delivery can be challenging because the clients, the service providers and various Internet service providers (ISPs) involved do not have a global view of the end-to-end network condition. In this case, a video service provider does not have access to both the transit ISPs and the last mile network that actually reaches the client. Once a client is connected to a delivery node in a CDN operated by a third party, there is no method for a video service provider to track the network condition on the video streaming flow in real time. Also, the delivery node is rarely switched to another node during playback. If the network condition is unstable, the client can suffer from buffering until the end of play time. Therefore, even if users pay for HD videos, they can end up watching low bitrate videos due to the Internet-side and / or the CDN-side network problems.

In order to mitigate these problems, today's OTT video service providers take advantage of client-side rate selection algorithms where a video player automatically adjusts resolutions depending on the network conditions. However, it does not resolve the root cause of the congestion. For instance, the link on the routing path from the assigned delivery node to the client may cause the bottleneck or the delivery node may experience malfunction, in which case changing the resolution is not the best way to improve video QoE. When a client requests a video, a geographically close delivery node is assigned to the client. Although a cache server located near a client typically provides a fast delivery, it is possible that the network conditions can be unstable at the moment. In such case, other delivery nodes that are located distant from the client may provide more reliable streaming experience.

IV. QoE-aware Video Streaming using SDN

We leverage SDN to assist video service providers in selecting the best delivery nodes when clients request videos. In addition, we propose a Constrained Shortest Path First (CSPF) path selection algorithm over MPLS in order to find the best routing path for each streaming flow. Figure 1 shows our simplified SDN-based video streaming architecture that consists of a video optimization server, a video front-end server, multiple delivery nodes and a client. Taking into account scalability and performance issues, we may deploy multiple SDN controllers in the network. Via an SDN controller, our video optimization server (as an SDN application) monitors network conditions and updates routing tables of WAN routers in the network.

As an example, the overall procedures are as follows:

1) A client sends a video request to a video front-end server.
2) The video front-end server sends our video optimization server a list of available delivery nodes that can stream the requested video at the moment.
3) Our SDN application takes account of the network conditions of each connected link on the paths to the client. The proposed measurements for video streaming include available bandwidths, packet loss rates and jitter. It chooses the best available delivery node and stores the connection information such as IP addresses of the client and the selected delivery node, an assigned MPLS label and selected video resolution in the video database.
4) Once the connection is established, the video player running on the client device periodically reports video QoE metrics to the video delivery node (Section IV-A).
5) When buffering events occur, our video optimization server pinpoints a bottleneck (Section IV-B). It is designed to find the best available routing path based on the CSPF algorithm over MPLS (Section IV-C).
6) We dynamically change the delivery node if all available paths from the assigned node experience congestions. In this case, the first assigned delivery node sends an HTTP redirection message to the video player, and have the client connected to another available delivery node that can provide the content with higher networking performance. The address of newly assigned delivery node can be obtained directly from the video optimization server.

TABLE I: Required TCP throughput for CSPF-based path selection algorithms

Selected resolution	Required TCP throughput
1080p	5 Mb/s
720p	2.5 Mb/s
480p	1 Mb/s
360p	725 Kb/s
240p	325 Kb/s

Once the client is connected to the new delivery node, the video player sends a new HTTP GET message that requests next video segments to continue to watch the video from the disconnected point.

A. Application-level Video QoE metrics

Video service providers typically do not have any access to last mile networks (e.g., local ISPs) of clients. We propose to measure end-to-end network conditions between a video player and a delivery node. In our proposed architecture, the delivery node is designed to periodically receive various QoE measurements directly from the video player to analyze user-perceived video quality.

Existing QoS metrics such as packet loss rate, goodput, delay, jitter and throughput are used to indicate the impact on the video quality from the network operator's point of view, but do not present the user-perceived video quality. Moreover, it is difficult to use traditional Peak Signal to Noise Ratio (PSNR) where received frames and referenced frames of an original video are compared to measure video QoE. Such frame-to-frame comparison mechanism is inaccurate for mobile video streaming because frame loss frequently occurs over a lossy wireless channel [10]. Due to the above reasons, we suggest commonly-used quality metrics for Internet video streaming [11] [12], where video start-up latency, buffering rate and playout buffer status are mainly considered to measure the video QoE.

B. Pinpointing a bottleneck using SDN

Our video optimization server is capable of catching buffering events based on the feedback directly from a delivery node which obtains the QoE metrics from a video player. It is straightforward to find the bottleneck link in an SDN-enabled network. When the buffering events occur, the SDN application first obtains the flow information from the video database (Figure 1) such as source / destination IP addresses, routing paths (selected MPLS labels) and requested video resolution. Then, it sends queries to collect the current data rate of the video streaming flow on each connected link on the path (e.g., obtaining network statistics of an individual flow using OpenFlow [13]). There is the recommended downloading bitrate which represents the amount of bitrate required to play the selected resolution without any viewing interference. For example, YouTube requires 2.5 Mb/s for 720p and 725 Kb/s for 360p. We define a link as a bottleneck if it provides lower data rate than the required bitrate of current streaming flow, which may cause buffering experience at client-side.

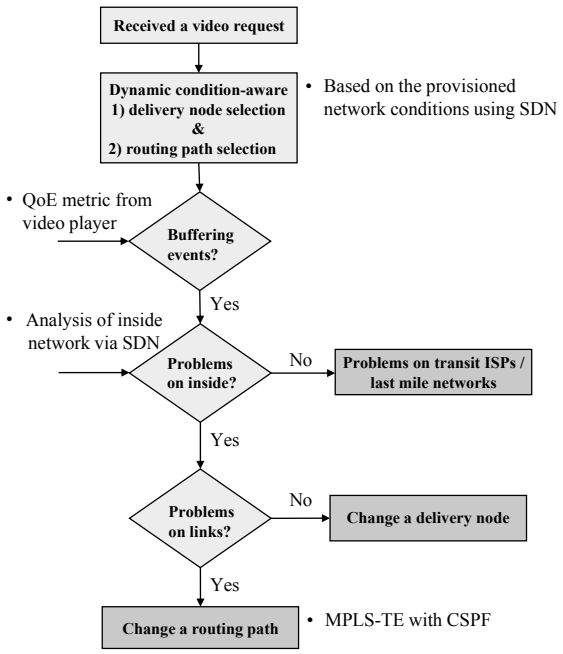


Figure 2: A simplified flowchart of a decision tree

C. Dynamic network condition-aware path optimization with SDN

We use MPLS-TE over SDN to control video streaming flow [4], [5]. An MPLS enables ISPs to provide QoS in layer 3 networks. In an MPLS network, routers conduct packet-forwarding decisions based only on the labels assigned on data packets instead of inspecting an IP address of each packet. Different MPLS labels are assigned to corresponding Labeled Switch Paths (LSPs). Typically, those labels are attached with IP packets and removed from the packets at Label Switch Routers (LSRs) and label swapping can be performed on the intermediate routers. From a QoS standpoint, MPLS-TE allows network operators to efficiently manage different kinds of data streams based on service plans and speed up network traffic flow. According to recent studies [4], [5], MPLS-TE architecture can be more flexible and simple over the SDN platform by separating the control plane from the data plane. In this paper, we apply it for the video streaming use-case and build the prototype to show the feasibility of developing MPLS-TE over SDN.

We implement a CSPF algorithm over MPLS-TE in order to select the best available routing path from a delivery node to a client. It runs shortest path algorithm after selecting links that meet a given set of constraints. In our case, we take account of three constraints (TCP bandwidth, packet loss rate and jitter) that are typically considered important for video streaming. When a client experiences buffer freeze, for example, our SDN application collects network conditions on connected links and run a CSPF algorithm taking into account the required bandwidth in Table I to find the best available LSP. We consider packet loss rate ($< 5\%$) for buffered video streaming and put more weight on packet jitter (< 20 ms) for live streaming.

Basic CSPF algorithms where a set of video streaming

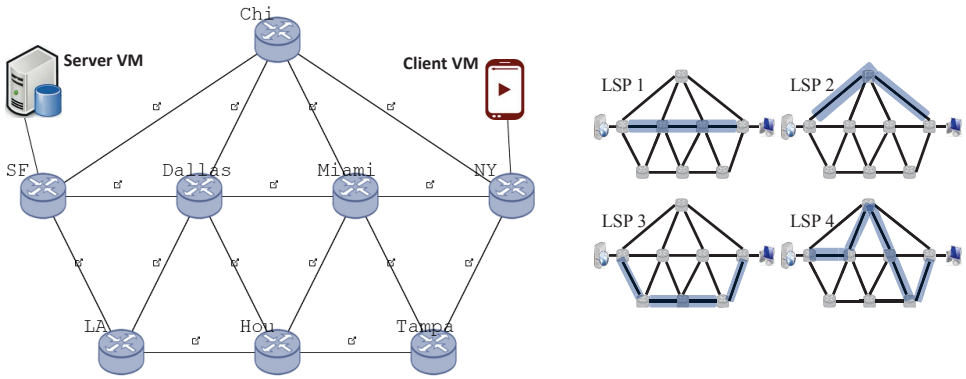


Figure 3: Implementing a testbed using Junos Space and WAN routers of Juniper Networks

requirements are considered to select the best LSP may encounter load-balancing problems on WAN links. For instance, if multiple clients request to change routing paths from the same time and place, the current CSPF algorithms may lead all the clients to take the same LSP. If the selected link is running at 80% - 90% utilization and sudden spikes of network traffic arise (e.g., during busy hours), the interface may become overwhelmed and start to drop packets, which will eventually degrade video QoE. Taking into account the load-balancing on WAN links, our CSPF algorithm has the following rules:

1) Prune WAN links that do not satisfy the required bandwidth, packet loss rate and jitter.
2) If multiple LSPs that meet the requirements are available, our CSPF-based load-balancing rule selects the LSP with the lowest link utilization.
3) If several LSPs have the same link utilization, our SDN application selects the LSP with the smallest number of hops.

In summary, Figure 2 shows our simplified flowchart of a decision tree in our proposed SDN-based video streaming architecture.

V. Implementation

As a proof of concept, we have implemented our SDN-based video streaming architecture using Junos Space that is a comprehensive network management solution developed by Juniper Networks [2]. It provides a centralized management plane to control switching, routing and security networking devices. Our SDN application is designed to improve network utilization and user-perceived video quality under dynamic network conditions. In order to achieve this, we have implemented server and client side applications over SDN.

- **Server-side application**: This is an application that determines the best paths and updates MPLS labels in real time based on our CSPF-based algorithm. It communicates with the SDN controller using RESTful APIs. It also provides GUI in order to visualize network topology and networking statistics.
- **Client-side application**: This is a light-weight plugin embedded in an HTML5 video player. It is designed to identify video resolution selected by a client and periodically report user-perceived experience to a connected delivery node. The QoE metrics include the player state (e.g., playing, paused and finished) and the status of video playout buffer while downloading a video. Those information is periodically delivered to the connected delivery node over HTTP POST messages.

Figure 3 shows our testbed network. The specific testbed setups are:

- A network control machine using the Junos Space SDN platform is connected to eight Juniper edge routers;
- The routers use an MPLS protocol to deliver video packets. Each router references the short label attached to decide a routing path of traffic flow, instead of performing an IP lookup; and
- The video packets are delivered from the server located in SF to the client located in NY via one of the four predefined LSPs.

VI. Evaluation

Due to the difficulties of creating real WAN traffic and in order to test our routing algorithms extensively in various scenarios, we have created a simulation tool that reflects the same network topology in our testbed. In our simulated network, video packets are delivered from virtual video servers to virtual clients via the links that are connected among virtual WAN routers. The video player running on the client-side has been designed to adjust resolutions based on downloading TCP throughput of streaming flow in the network.

We assume that links with 100 Mb/s bandwidth capacity are running between 80% and 90% utilization during busy hours. In order to simulate real-environmental network conditions, we take account of recent mobile streaming statistics [14] indicating that most clients watch low or medium resolutions, and about 1% of total mobile subscribers watch high definition (e.g., 720p and 1080p) videos. Based on the information, we inject background traffic flows on each link that follows Poisson distribution where 200 clients on average request a video per minute from each router and 99% of total streaming flows generate 0.5 Mb/s on average and 1% of total flows consume 2.5 Mb/s bandwidth on average.

In order to show the feasibility of our approach, we experimented two following scenarios:

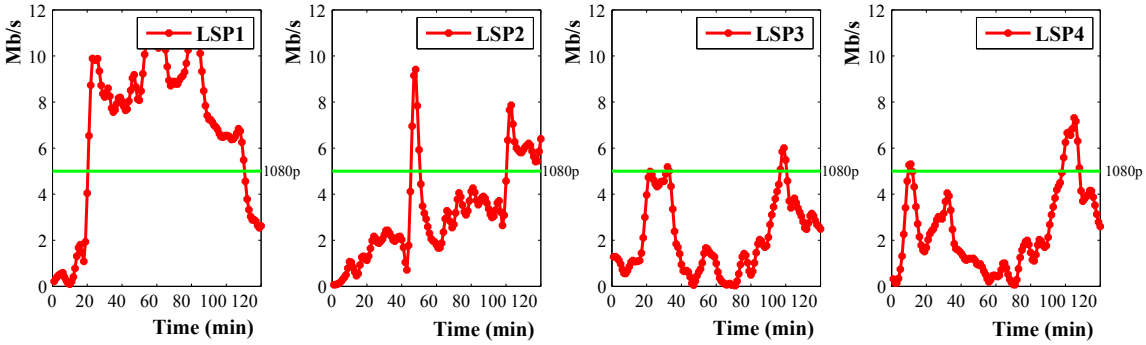


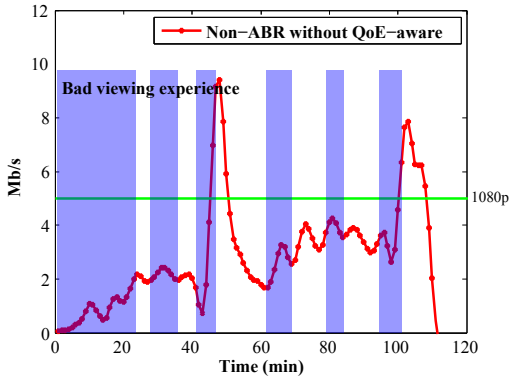
Figure 4: Available bandwidth capacity on LSPs in Scenario 1

Scenario 1) Non-ABR without QoE-aware vs. Non-ABR with QoE-aware: In non-ABR scenario, a video player does not switch resolutions during a download. Without QoE-aware, a client continues to watch a movie with 1080p via LSP 2 that has minimum number of hops among LSPs. With our QoE-aware streaming, a video player downloads the same movie via dynamically changing LSPs based on our CSPF algorithm.
Scenario 2) ABR without QoE-aware vs. ABR with QoE-aware: In ABR scenario, a video player automatically adjusts resolutions based on network conditions. Without QoE-aware, a video player switches resolutions but does not change LSPs. With our QoE-aware streaming, a video player downloads the 1080p video via dynamically changing LSPs. It only degrades a resolution if there are no available LSPs that meet the required TCP throughput (e.g., 5 Mb/s for 1080p).

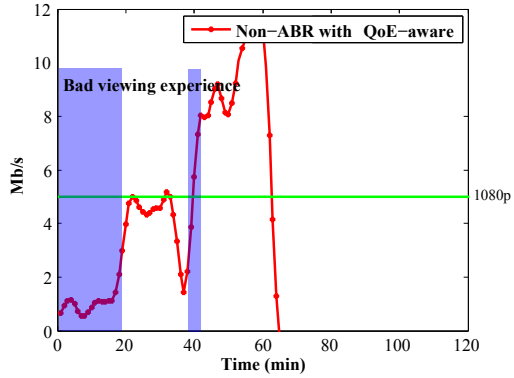
We note that all streaming flows from the SF node to the NY node in both scenarios first take LSP 2 with minimum number of hops, as shown in Figure 3. Figure 4 shows the available bandwidth on each LSP in Scenario 1. For the first 20 minutes, there are no good LSP that has available bandwidth more than 5 Mb/s. Figure 5 shows the experimental results of Scenario 1. X-axis represents the elapsed time and Y-axis indicates the downloading datarate at client-side. We measured the data until the video player completely downloaded the video content. We counted the accumulated received bytes every minute and compared it with the required bitrate of the selected resolution. We put a square box if the video player experienced bad networking conditions (downloading data rate $<$ required bitrate) for at least five seconds during the sampling period. In such unstable network conditions, there is a high possibility of experiencing buffering events at the client-side.

In a non-ABR without QoE-aware mode, the video player had bad viewing experience for 52 minutes (the total length of square box) via LSP 2. The video player with our QoE-aware mechanism over SDN experienced the unstable networking conditions only for 21 minutes in total. At the beginning, the SDN controller switched the path from LSP 2 to LSP 3 since it was the best one among others, and then it changed to LSP 1 at time t=39 to provide a fast delivery.

In Scenario 2, we played a 1080p movie with 100 minutes of length in both QoE-aware and non-QoE-aware modes. We measured how often the video player switched resolutions while playing the video. Figure 6 shows our experimental



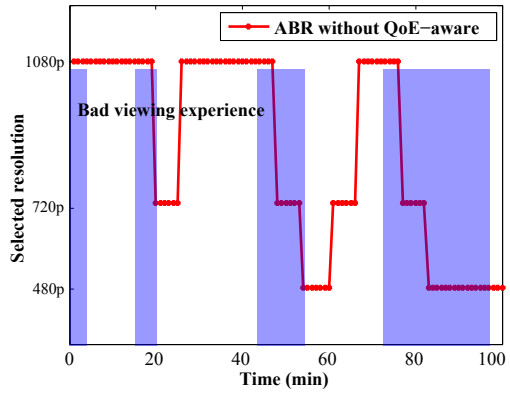
(a) A non-ABR mode without QoE-aware



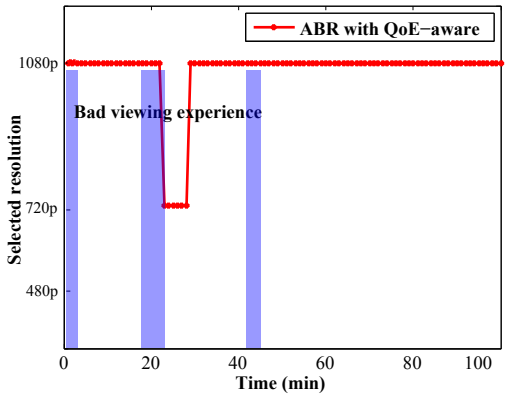
(b) A non-ABR mode with QoE-aware

Figure 5: Downloading TCP throughput and period of bad viewing experience in non-ABR modes without QoE-aware and with QoE-aware

results. Y-axis represents the video resolution selected by the video player. As we see in Figure 6a, the video player often changed resolutions and had more bad viewing experience, compared to the one in Figure 6b. In this experiment, the video player without QoE-aware played 1080p, 720p and 480p resolution for 50 minutes, 24 minutes and 26 minutes, respectively while the video player with our QoE-aware mechanism downloaded 1080p at the most of the time. We conducted the same experiment a hundred times and calculated the statistics. As shown in Table II, our QoE-aware mechanism over SDN reduces the bad viewing experience by 5.8 minutes on average,



(a) An ABR mode without QoE-aware



(b) An ABR mode with QoE-aware

Figure 6: Downloading TCP throughput and period of bad viewing experience in ABR modes without QoE-aware and with QoE-aware

TABLE II: Selected video resolution and period of bad viewing experience while watching a video with 100 minutes of length

	ABR without QoE-aware	ABR with QoE-aware
Avg. bad viewing experience period	10.36 minutes	4.56 minutes
1080p	69.08%	77.16%
720p	19.74%	16.26%
480p	11.18%	6.58%

and provides higher resolution of video streaming while the video player downloaded the content.

VII. Conclusions

In today's OTT video delivery architecture, it is difficult to track user-perceived video quality once a delivery node has been connected to a client. Without changing routing paths and delivery nodes, only switching resolutions at client-side may not resolve the bottleneck problems that degrade video QoE. For instance, it is possible that the routing paths between the assigned delivery node and the client experience congestion at the moment.

From the video service provider's point of view, we use MPLS-TE over SDN. Our goal is to improve video QoE. To monitor watching experience of a client in real time, we propose to measure video QoE metrics (e.g., buffering status and video player state) directly at video players during a download. Based on the end-to-end feedback, our SDN controller is designed to *a*) select the best available delivery node that can stream the content with more reliable network conditions than others presently; and *b*) dynamically change routing paths among WAN routers using MPLS-TE.

In our simulation, our proposed QoE-aware mechanism shows 55.9% improvement on enhancing viewing experience especially during busy hours. It selects better routing paths to provide higher resolution video during a download.

References

[1] H. Nam, K. H. Kim, D. Calin, and H. Schulzrinne, "Towards Dynamic Network Condition-Aware Video Server Selection Algorithms over Wireless Networks," Department of Computer Science, Columbia University, Tech. Rep. cucs-001-14, Jan. 2014.

[2] Junos Space. [Online]. Available: http://www.juniper.net/us/en/products-services/network-management/

[3] Z. A. Qazi, J. Lee, T. Jin, G. Bellala, M. Arndt, and G. Noubir, "Application-awareness in SDN," in *Proceedings of the ACM SIGCOMM Conference*, Hong Kong, China, Aug. 2013.

[4] A. R. Sharafat, S. Das, G. Parulkar, and N. McKeown, "MPLS-TE and MPLS VPNS with Openflow," in *Proceedings of the ACM SIGCOMM Conference*, Toronto, Ontario, Canada, Aug. 2011.

[5] S. Das, Y. Yiakoumis, G. Parulkar, N. McKeown, P. Singh, D. Getachew, and P. Desai, "Application-aware Aggregation and Traffic Engineering in a Converged Packet-Circuit Network," in *Optical Fiber Communication Conference and Exposition (OFC/NFOEC) and the National Fiber Optic Engineers Conference*, Los Angeles, USA, Mar. 2011.

[6] About NOX. [Online]. Available: http://www.noxrepo.org/

[7] M. Jarschel, F. Wamser, T. Höhn, T. Zinner, and P. Tran-Gia, "SDN-based Application-Aware Networking on the Example of YouTube Video Streaming," in *2nd European Workshop on Software Defined Networks (EWSDN)*, Berlin, Germany, Oct. 2013.

[8] Application-aware Routing in Software-defined Networks. [Online]. Available: http://www.aricent.com/pdf/Aricent_Whitepaper_-_Application_Aware_Routing_in_SDN.pdf

[9] Mininet: An Instant Virtual Network on your Laptop. [Online]. Available: http://mininet.org/

[10] A. Chan, K. Zeng, P. Mohapatra, S.-J. Lee, and S. Banerjee, "Metrics for Evaluating Video Streaming Quality in Lossy IEEE 802.11 Wireless Networks," in *Proceedings of the 29th Conference on Information Communications*, ser. INFOCOM'10, San Diego, California, USA, Mar. 2010.

[11] A. Balachandran, V. Sekar, A. Akella, S. Seshan, I. Stoica, and H. Zhang, "A Quest for an Internet Video Quality-of-experience Metric," in *Proceedings of the 11th ACM Workshop on Hot Topics in Networks*, ser. HotNets-XI, Redmond, Washington, Oct. 2012.

[12] R. Serral-Gracià, E. Cerqueira, M. Curado, M. Yannuzzi, E. Monteiro, and X. Masip-Bruin, "An Overview of Quality of Experience Measurement Challenges for Video Applications in IP Networks," in *Proceedings of the 8th International Conference on Wired/Wireless Internet Communications (WWIC)*, Lulea, Sweden, Jun. 2010.

[13] OpenFlow Switch Specification Version 1.4.0. [Online]. Available: https://www.opennetworking.org/

[14] Citrix, "Bytemobile Mobile Analytics Report," Citrix Systems, Tech. Rep., May 2012.