HTTP Adaptive Streaming System Maximizing Overall Video Quality over SDN-enabled Wi-Fi APs

Hyunmin Noh
Department of Computer Science and Engineering
POSTECH
Pohang, Republic of Korea
hmnoh@postech.ac.kr

Gi Seok Park
Department of Computer Science and Engineering
Dongguk University
Gyeongju, Republic of Korea
giseok@dongguk.ac.kr

Yunmin Go
School Of Computer Science and Electrical Engineering
Handong Global University
Pohang, Republic of Korea
yunmin@handong.edu

Hwangjun Song
Department of Computer Science and Engineering
POSTECH
Pohang, Republic of Korea
hwangjun@postech.ac.kr

***Abstract*—This work presents a novel HTTP adaptive streaming system to maximize overall video streaming service quality of all the clients over SDN-enabled Wi-Fi APs. To archive the goal, the proposed system employs multipath technology to overcome the limitation of a Wi-Fi AP, and SDN technology to effectively harmonize multiple Wi-Fi APs. During the streaming services, SDN controller periodically monitors the traffic information collected from the APs, and then determines video bitrate and segment transmission schedule based on this information. The proposed system is fully implemented in real Wi-Fi network environments.**

Keywords—HTTP adaptive streaming, SDN, multipath, bitrate adaptation, segment scheduling

I. INTRODUCTION

The widespread use of video streaming services such as YouTube, Hulu, and Netflix and the demand for high-quality videos have led to a tremendous increase in global mobile traffic. According to the ITU report [1], the overall mobile traffic is estimated to grow at an annual rate of approximately 55% in 2020-2030. Ericsson [2] estimated that the video traffic accounts for 66% of all mobile data traffic in 2020-2021 and is expected to increase to 77% by 2026. One of the main solutions to the increasing demand for wireless network bandwidth is Wi-Fi networks. According to the Cisco report [3], globally, there will be nearly 628 million public Wi-Fi hotspots by 2023, a fourfold increase from 169 million hotspots in 2018. However, it is still difficult to provide high-quality and seamless streaming services over wireless networks due to the limited wireless resources and time-varying network conditions.

Recently, HTTP adaptive streaming (HAS) has been widely used in video streaming services over limited and time-varying wireless networks. In 2012, the HAS standard called dynamic adaptive streaming over HTTP (DASH) [4]. So far, many research has been devoted to adaptive streaming services. Spiteri et al. [5] proposed BOLA which uses control segment bitrate by using the playback buffer occupancy-based Lyapunov optimization to enhance the video quality and avoid buffer underflow. Liu et al. [6] proposed a novel rate adaptation algorithm. It employs smoothed HTTP throughput based on the segment fetch time to provide smooth and stable adaptive HTTP streaming services. However, these studies still may have quality degradation when multiple DASH clients have a common bottleneck link. In these studies, each DASH client independently determines the next segment bitrate by considering only the local network status measured and estimated by itself. This process may cause several problems such as instability, unfairness, and underutilization of limited network resources [7]. When the download intervals of two clients do not overlap, each client overestimates the available bandwidth of the shared bottleneck link and then chooses a higher bitrate for the next request segment then the current network status. Subsequently, when the segment transmissions of the clients are overlapped, congestion occurs since the requested segment bitrates are overdetermined compared to the available bandwidth of the bottleneck link. These issues should be addressed in enterprise/private networks, where many users share a bottleneck link.

Software-defined networking (SDN) [8] is a network configuration technology in which a network is easily programmable by decoupling the control and forwarding planes. With SDN, the entire network can be managed by using a centralized controller with open interfaces (e.g. OpenFlow [9]). SDN provides application programming interfaces (APIs) for implementing network applications. Thus, new applications can be created and deployed to meet specific requirements. Therefore, combining SDN and HAS for enterprise/private networks is an effective approach [10, 11]. However, it is still difficult to provide seamless and high-quality video streaming services when tens and hundreds of mobile devices attempt to associate with one Wi-Fi Access Point (AP) due to the bandwidth limitations of a single wireless network.

In this work, we propose a HTTP adaptive streaming system to improve the overall video quality of all DASH clients via SDN-enabled Wi-Fi APs. To overcome the limited wireless bandwidth of a single Wi-Fi AP, the proposed HAS system can simultaneously receive video segments via multiple Wi-Fi network interfaces. In addition, SDN is employed to quickly respond to time-varying network conditions and control video segment transmission via multiple Wi-Fi APs with a global view of the network.

II. PROPOSED HTTP ADAPTIVE STREAMING SYSTEM

The goal of the proposed system is to maximize the overall spatial video quality of all clients without noticeable frozen video instants although congestion may occur at Wi-Fi APs. The overall architecture of the proposed system is shown in Fig. 1. The proposed system consists of a media server, SDN controller, SDN-enabled Wi-Fi APs, and DASH clients. The media server contains video segments and the extended MPD including

978-1-6654-6975-3/22/$31.00 ©2022 IEEE

PSNR information as well as segment bitrate and URL to provide rate-distortion information as shown in Fig. 2. It is assumed that the SDN controller manages the enterprise/private network (i.e. APs and DASH clients within the enterprise/private network are operated by the SDN controller). It includes two main modules: a data collector and streaming manager module. The SDN-enabled AP contains an SDN switch (i.e. Open vSwitch (OVS)). The DASH client consists of multiple wireless interfaces, a multipath agent, an HTTP client, and a media player. It may be connected to multiple adjacent Wi-Fi APs by using multiple wireless interfaces.

The overall working procedure is presented in Fig. 3. At first, the DASH client requests the extended MPD from the media server, and then the media server transmits the extended MPD to the client. When the MPD is delivered via the SDN-enabled Wi-Fi AP, the data collector on the SDN controller replicates the MPD to obtain video content characteristics (e.g. video encoding bitrates and rate-distortion model parameters). While receiving MPD and segment data, the DASH clients and SDN-enabled AP periodically report the network status information (e.g. available bandwidth between the APs and DASH clients, and received signal strength indicator (RSSI))–and current playback buffer time to the data collector on the SDN controller. By using these information, the streaming manager on the SDN controller determines the operating parameters, such as the video bitrate and the amount of data transmitted via each Wi-Fi network interface of all DASH clients. The streaming manager then provides these operating parameters to the DASH clients. After that, the multipath agent on the DASH client requests a part of the video segment to each Wi-Fi interface according to the operating parameters. The SDN controller synchronously controls the segment transmissions among the DASH clients in the time-slot unit. After receiving all parts of the segment via multiple Wi-Fi APs, the multipath agent on the DASH client aggregates the segment and stores it in a playback buffer. And then, the DASH client requests the next segment. This process is repeated until the whole streaming service is complete.

A. Problem Description

In this section, we describe problem formulation in detail. Before presenting a detailed description, some symbols are defined. First, the partial-segment bitrate allocation vector is represented by

$$\vec{r}_i = \left(r_{i,1},...,r_{i,j},...,r_{i,|\mathbf{AP}|}\right), \tag{1}$$

where $r_{i,j}$ represents the partial-segment bitrate to be delivered via AP #*j* to DASH client #*i* and $\mathbf{AP}$ is the set of available APs. In the proposed system, one segment is divided into the ratio of partial segment bitrate and total segment bitrate for each AP and DASH client and transmitted via multiple APs. The peak signal-to-noise-ratio (PSNR) is employed to measure the objective video quality at a DASH client, and the PSNR of the video provided to DASH client #*i* is simply modeled by

$$a_i \log_{10}\left(\sum\nolimits_{j\in \mathbf{AP}} r_{i,j}\right) + b_i, \tag{2}$$

where a_i and b_i are the coefficients of the rate-distortion model of video content provided to DASH client #*i* and $\sum_{j\in AP} r_{i,j}$ is overall the bitrate of an segment transferred to

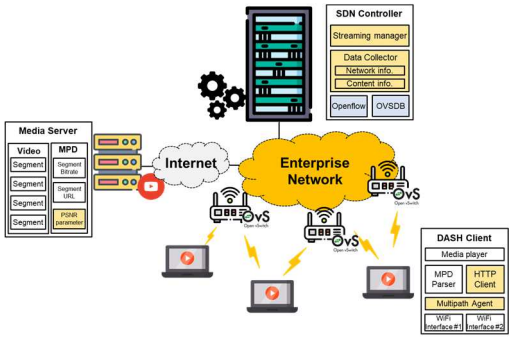


Fig. 1. Overall architecture of the proposed system.

Fig. 2. Example of an extended MPD file structure.

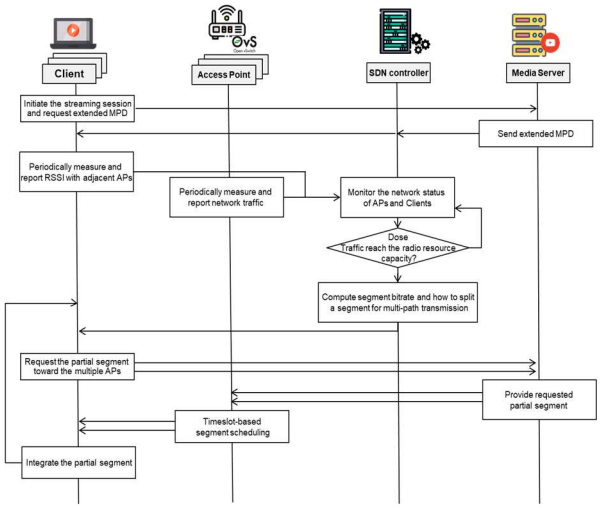


Fig. 3. Overall working procedure of the proposed system.

DASH client #*i* via multiple Wi-Fi APs. In the proposed system, we obtain these coefficients through a curve fitting process by using the trust-region algorithm [12], and the DASH client achieves them from the expanded MPD file

If DASH clients simultaneously activate multiple wireless network interfaces, then SDN controllers support better video quality for DASH clients and flexibly control network resources to handle network congestion. However, DASH clients are unwilling to activate their additional interfaces because it consumes additional energy and computing power. Thus, the incentive mechanism is quite helpful to encourage users to turn on network interfaces at the same time. Now, we define the incentives as follow.

$$\omega\left(n_i^{net}\right) = \frac{1}{1+e^{-\alpha^{net}\left(n_i^{net}-1\right)}} \quad \text{for } n_i^{net} \geq 1 \tag{3}$$

where n_i^{net} is the number of activated Wi-Fi network interfaces of DASH client #*i* and α^{net} is the coefficient of the weighting factor for network interfaces. As n_i^{net} increases, more incentives

are provided whose increasing policy is determined by the parameter α^{net}.

To calculate the segment download time, the available bandwidth over the Wi-Fi APs should be estimated at first. In the proposed system, the SDN controller periodically obtains the estimated bandwidth between DASH client #*i* and AP #*j* as follows

$$\hat{bw}_{i,j} = \begin{cases} \hat{bw}_{i,server}^{ovs_j} & \text{if client \#}i\text{ is connected to the AP \#}j \\ \min\left(\hat{bw}_{j,server}^{ovs_j}, \hat{bw}_{i,j}^{rssi}\right) & \text{otherwise,} \end{cases} \quad (4)$$

where $\hat{bw}_{i,server}^{ovs_j}$ is the estimated available bandwidth between DASH client #*i* and the media server, $\hat{bw}_{j,server}^{ovs_j}$ is the estimated bandwidth between AP #*j* and the media server, and $\hat{bw}_{i,j}^{rssi}$ is the estimated bandwidth between DASH client #*i* and AP #*j*. $\hat{bw}_{i,server}^{ovs_j}$ and $\hat{bw}_{j,server}^{ovs_j}$ are estimated by using the ARIMA [13] method based on the port-specific traffic statistics in the OVS of AP #*j*, and $\hat{bw}_{i,j}^{rssi}$ is estimated by using the RSSI [14]. If DASH client #*i* is connected to AP #*j*, then the bandwidth between the DASH client and media server via AP #*j* can be directly obtained by $\hat{bw}_{i,server}^{ovs_j}$. On the other hand, if DASH client #*i* and AP #*j* are disconnected, then $\hat{bw}_{i,j}$ is obtained by comparing $\hat{bw}_{i,j}^{rssi}$ and $\hat{bw}_{j,server}^{ovs_j}$ to take into account the end-to-end network bottleneck. Now, we can formulate the proposed system as follows.

Problem formulation: Determine $\vec{r}_i$ for $\forall i \in \mathbf{C}$ and $\forall j \in \mathbf{AP}$ with the given n_i^{net} for $\forall i \in \mathbf{C}$ to maximize the weighted sum of objective visual quality for DASH clients, i.e.

$$\sum_{i \in \mathbf{C}} \omega\left(n_i^{net}\right) \cdot \left(a_i \log\left(\sum_{j \in \mathbf{AP}} r_{i,j}\right) + b_i\right), \quad (5)$$

$$\text{subject to } \left\|\vec{r}_i\right\|_o \le n_i^{net} \text{ for } \forall i \in \mathbf{C}, \quad (6)$$

$$\sum_{j \in \mathbf{AP}} r_{i,j} \le r_i^{req} \text{ for } \forall i \in \mathbf{C}, \quad (7)$$

$$\text{and } \sum_{i \in \mathbf{C}} \frac{r_{i,j} \cdot T^{seg}}{\hat{bw}_{i,j}} \le T^{slot} + S\left(t_{avg}^{buf} - T_{th}^{buf}\right) \text{ for } \forall j \in \mathbf{AP}, \quad (8)$$

where $\left\|\vec{r}_i\right\|_o$ is the number of non-zero entries of the vector $\vec{r}_i$, r_i^{req} is the segment bitrate requested by the DASH client #*i*, $\mathbf{C}$ is the set of DASH clients, T^{seg} is the segment duration, T^{slot} is the time unit of each slot, t_{avg}^{buf} is the average media playback buffer time for all DASH clients, and $S(\cdot)$ is the penalty function to prevent the playback buffer underflow caused by sudden network fluctuations. Eq. (6) means the constraint that the number of APs delivering segment data to a DASH client should not be larger than the number of activated Wi-Fi interfaces of the DASH client, Eq. (7) is the constraint that the segment bitrate served to the DASH client should be less than or equal to the bitrate requested by the DASH client, and Eq. (8) represents that DASH clients should be download a segment within a time constraint. The time constraint is adjusted to prevent playback buffer underflow and to sustain a stable buffer condition. If the average playback buffer time of all DASH clients is larger than the threshold time T_{th}^{buf}, then $S(x)$ should be increased to enhance the quality of new-coming segments while consuming redundant playback buffering segments. Otherwise, $S(x)$ should be decreased to rapidly fill the playback buffer by reducing the segment bitrate. Thus, a third-degree polynomial $S(x)$ is selected by

$$S(x) = \begin{cases} \min\left(p_1 x^3 + p_2 x^2 + p_3 x + p_4, \frac{T^{slot}}{2}\right) & x \ge 0 \\ \max\left(p_1 x^3 + p_2 x^2 + p_3 x + p_4, -\frac{T^{slot}}{2}\right) & \text{otherwise,} \end{cases} \quad (9)$$

where p_1, p_2, p_3, and p_4 are the coefficients of $S(x)$. The range of $S(x)$ is limited to $[-T^{slot}/2, T^{slot}/2]$ to prevent sudden video quality changes that may cause blinking artifacts during video streaming services.

By the way, the above problem is too complicate to be solved in real-time due to its high computational complexity. The partial segment bitrate delivered to a DASH client via an AP depends on not only its time slot but also those of other APs. To obtain a practical and efficient near-optimal solution with reasonable computational complexity, we simplify the above formulation at a single AP.

Simplified Problem Formulation at the AP #*k*:

Determine $r_{i,k}$ with given n_i^{net} and $r_{i,j}$ for $\forall i \in \mathbf{C}$ and $\forall j \in \mathbf{AP}/\{k\}$ to maximize the weighted sum of objective visual quality of DASH clients at the AP #*k*, i.e.

$$\sum_{i \in \mathbf{C}} \omega\left(n_i^{net}\right) \cdot \left\{a_i \log\left(\sum_{j \in \mathbf{AP}} r_{i,j}\right) - a_i \log\left(\sum_{j \in \mathbf{AP}/\{k\}} r_{i,j}\right)\right\}, \quad (10)$$

$$\text{subject to } \left\|\vec{r}_i\right\|_o \le n_i^{net} \text{ for } \forall i \in \mathbf{C}, \quad (11)$$

$$\sum_{j \in \mathbf{AP}} r_{i,j} \le r_i^{req} \text{ for } \forall i \in \mathbf{C}, \quad (12)$$

$$\text{and } \sum_{i \in \mathbf{C}} \frac{r_{i,j} \cdot T^{seg}}{\hat{bw}_{i,j}} \le T^{slot} + S\left(t_{avg}^{buf} - T_{th}^{buf}\right) \text{ for } \forall j \in \mathbf{AP} \quad (13)$$

where Eq. 10 indicates the change of the overall weight PSNR when $r_{i,k}$ (for $\forall i \in \mathbf{C}$) is delivered via AP #*k*.

B. Segment Bitrate Determining Process

In this section, we describe the details of the proposed segment bitrate determining process. Actually, it is well known that Eq. (10) is concave since it is the sum of log functions [15]. Now, the Lagrange multiplier is employed to obtain the solution. The following penalty function is defined by combining the cost function Eq. (10) and the constraint Eq. (13):

$$L\left(r_{1,k}, \ldots, r_{|C\setminus k|}, \lambda\right) = \sum_{i \in \mathbf{C}} \omega\left(n_i^{net}\right) \cdot \left\{a_i \log\left(\sum_{j \in \mathbf{AP}} r_{i,j}\right) - a_i \log\left(\sum_{j \in \mathbf{AP}/\{k\}} r_{i,j}\right)\right\}$$
$$+ \lambda \cdot \left(T_{slot} + S\left(t_{avg}^{buf} - T_{th}^{buf}\right) - \sum_{i \in C} \frac{r_{i,j} \cdot T^{seg}}{\hat{bw}_{i,j}}\right), \quad (14)$$

where λ is the Lagrange multiplier. Then, gradient method can be employed to find the solution for a single AP. In the proposed system, the SDN controller recursively selects an AP and determines segment bitrate delivered via the AP until the overall weighted PSNR converges. The details of the proposed algorithm are presented below.

Step 0) Initialize $\vec{r}_i$ for $\forall i \in \mathbf{C}$ and λ.

Step 1) Set the AP index k to 1.

Step 2) Calculate the partial segment bitrate $r_{i,k}$ for $\forall i \in \mathbf{C}$ to be delivered via AP #k by taking the partial derivative of Eq. (14).

Sept 3) If $r_{i,k}$ for $\forall i \in \mathbf{C}$ do not satisfy Eq. 13. Update λ by

$$\lambda = \lambda + \Delta\lambda, \tag{15}$$

$$\Delta\lambda = \frac{L(r_{1,k},...,r_{|C|,k},\lambda)}{T^{slot}} + S\left(t_{avg}^{buf} - T_{th}^{buf}\right), \tag{16}$$

then go back to **Step 2).**

Step 4) If $\sum_{j\in\mathbf{AP}} r_{i,j} > r_i^{req}$, $r_{i,k}$ is set to r_i^{req} - $\sum_{j\in\mathbf{AP}/\{k\}} r_{i,j}$.

Step 5) Change $r_{i,k}$ so that $\sum_{j\in\mathbf{AP}} r_{i,j}$ is the to the nearest values less than or equal to the bitrate in the MPD.

Step 6) Check the number of available network interfaces. For $\forall i \in C$, if $\|\vec{r}_i\|_o \geq n_i^{net}$, then find the smallest partial segment bitrate for DASH client #i among all APs and set to 0.

Step 7) Initialize λ and increase k to select the next AP. If $k > |\mathbf{AP}|$, go to **Step 1)**, otherwise, go to **Step 2)** until the overall weighted PSNR converges.

III. Experimental Results

The proposed system is implemented by using the NS-3 [17] for a large-scale network environment test, and a real Wi- Fi network testbed is realized with Raspberry Pis and laptops in a relatively small-scale network due to the practical reasons. First, we verify the performance of the proposed system in a simulation environment with a large number DASH clients and APs. Next, we evaluate the performance and feasibility of the proposed system in a real testbed.

A. Performance Verification on NS-3-based Large-scale Environment

In this section, we demonstrate the performance of the proposed system on NS-3-based large-scale environment. A network topology consists of 19 DASH clients and three 802.11n APs. The positions and mobilities of the entities are shown in Fig. 4. The DASH clients are connected to APs located within 50 m. For the Wi-Fi channel, yet another network simulator (YANS) model and Friis propagation loss model are employed. During the simulation, 24Mbps background traffic is generated to each AP. Three full high-definition video sequences, Big Buck Bunny, Elephants Dream, and Sintel are used as test videos. Each video stream is encoded as 100, 200, 400, 600, 800, 1000, 1200, 1400, 1600, 1800, and 2000 kbps by using the H.264 with 25 frames per second. The T^{seg} and T^{slot} are set to 2 sec. The coefficients of the rate-distortion model (i.e. a_i and b_i) are achieved from the PSNR of each segment and these values are included in the extended MPD, and the total playback time is set to 300 sec. DASH clients #1, 4, 7, 10, 13, 16 and 19 request Big Buck Bunny, DASH clients #2, 5, 8, 11, 14, and 17 request Elephants Dream, and DASH clients #3, 6, 9, 12, 15, and 18 request Sintel. All DASH clients start video playback when the playback buffer time is 2.0 sec. The maximum playback buffer time is set to 10 sec and T_{th}^{buf} is set to 6 sec. p_1, p_2, p_3, and p_4 are fixed to 0.01439, 0.02866, 0.02828, and 0.005873, respectively.

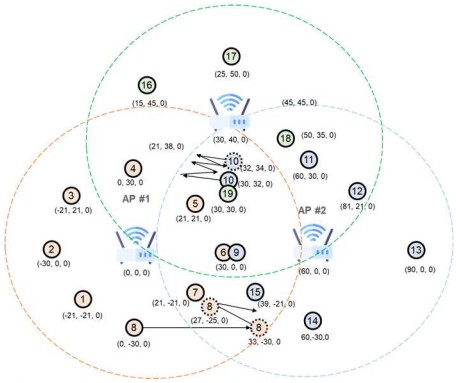


Fig. 4. Simulation network topology.

First, we investigate the proposed segment bitrate allocation algorithm. Fig. 5 represents the sum of weighted PSNR at 10 sec according to the iteration stage. At each iteration, the SDN controller recursively selects an AP and then updates the segment bitrate of DASH clients delivered via the selected AP. The time complexity of the proposed system is low because the simplified problem at a single AP can be easily solved by using the Lagrange multiplier, and thus the overall weighted PSNR quickly converges.

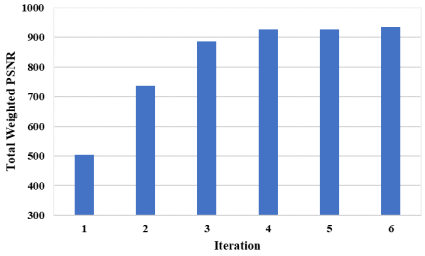
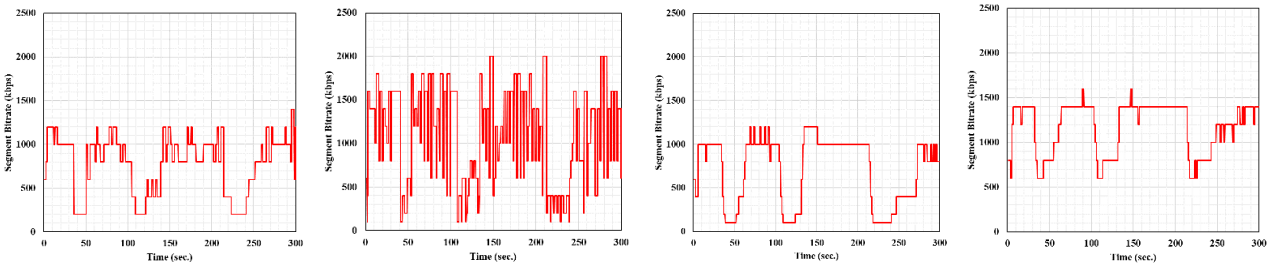


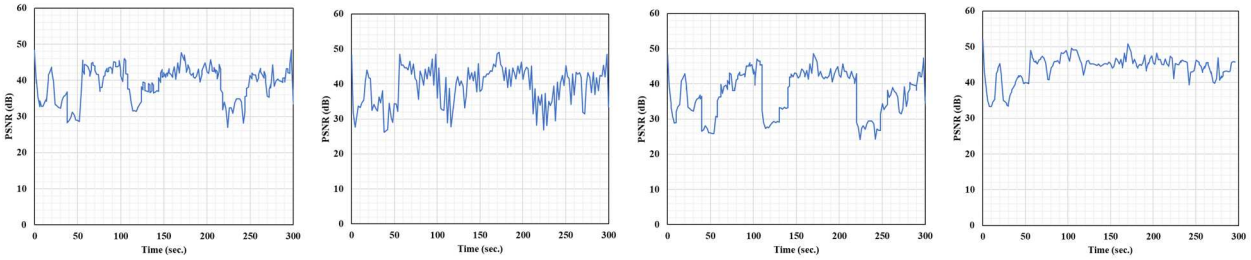
Fig. 5. Change in total weighted PSNR with the iteration.

Next, the performance of the proposed system is compared with the DASH.js [17], BOLA [5], and HAS-SDN [11] for a large-scale environment test. The experimental results are summarized in Table I. Figs. 6, 7, and 8 represent the bitrate adaptation and the corresponding PSNR and remaining playback buffer time of the DASH client #10, respectively. DASH.js performs rate control based on the available network bandwidth measured at the port of the DASH client by using the simple moving average method. Thus, it could not effectively respond to the playback buffer decrease caused by the sudden network fluctuations as shown in Figs. 6 (a) and 8 (a). In the case of BOLA, it utilizes Lyapunov optimization to fill buffer occupancy rapidly and minimize buffer underflow. As shown



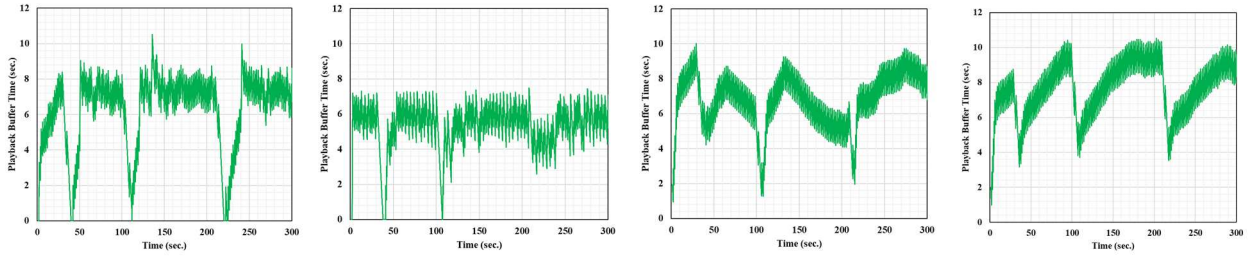
(a) (b) (c) (d)

Fig. 6. Segment bitrate comparison at DASH client #10: (a) Dash.js, (b) BOLA, (c) HAS-SDN, and (d) proposed system.



(a) (b) (c) (d)

Fig. 7. PSNR comparison at DASH client #10: (a) Dash.js, (b) BOLA, (c) HAS-SDN, and (d) proposed system.



(a) (b) (c) (d)

Fig. 8. Playback buffer time comparison at DASH client #10: (a) Dash.js, (b) BOLA, (c) HAS-SDN, and (d) proposed system.

in figs. 6 (b) and 8 (b), when the playback buffer of the DASH client #10 is low, the segment bitrate is rapidly decreased to fill the playback buffer. Thus, the buffer underflow occurs less than DASH.js, but the segment bitrate changes more often than DASH.js. In the case of HAS-SDN, it centrally controls the segment bitrate taking into account the network status and remaining playback buffer of all DASH clients sharing a single Wi-Fi network. Thus, it prevents buffer underflow while improving the overall PSNR more than DASH.js and BOLA as shown in Table I. However, as shown in Fig. 8 (c), HAS-SDN still has a quality degradation caused by the DASH client's movement due to the limitations of a single Wi-Fi network. In the case of the proposed system, it effectively performs the bitrate adaptation and segment transmission scheduling via multiple Wi-Fi APs by considering the overall network and buffer status of all DASH clients in the enterprise/private network. As shown in Fig. 7 and TABLE I, the proposed system always provides high-quality segment bitrate by using multiple interfaces while maintaining a stable playback buffer time. As shown in Fig 7 (d), even if some wireless channel conditions are degraded due to the movement of the DASH client #10, the DASH client maintains high PSNR because it can receive segment data via other Wi-Fi APs. In addition, the proposed system minimizes the changes in PSNR, segment bitrate, and remaining playback buffer of all DASH clients than other existing HAS systems by efficiently scheduling segment transmission via multiple APs by considering the rate-distortion characteristics and buffer occupancy as shown in Table I.

TABLE I. PERFORMANCE EVALUATION WITH EXISTING ALGORITHMS ON LARGE-SCALE ENVIRONMENT.

DASH Systems		DASH.js	Bola	HAS-SDN	Proposed System
PSNR (dB)	Avg.	41.57	41.58	43.46	44.3
	Std.	3.52	3.95	3.67	3.4
Bitrate (kbps)	Avg.	1141.61	1207.79	1355.15	1798.77
	Std.	258.04	429.42	256.39	168.61
Playback buffer time (sec.)	Avg.	6.56	7.39	6.89	10.44
	Std.	1.4	0.99	0.91	0.85
Avg. num. of buffer underflow		0.47	0.32	0	0

B. *Performance Verification on Real Wireless Network Testbed Environment*

In this section, we demonstrate the performance of the proposed system on a real Wi-Fi environment. The real testbed is constructed with a media server, SDN controller, APs, and laptops as shown in Fig. 9. We employ ONOS SDN controller and implement the HTTP streaming manager application on Ubuntu 18.04. The APs are implemented by using a Raspberry PI-3 with the OVS and hostapd. Five laptops are used as DASH

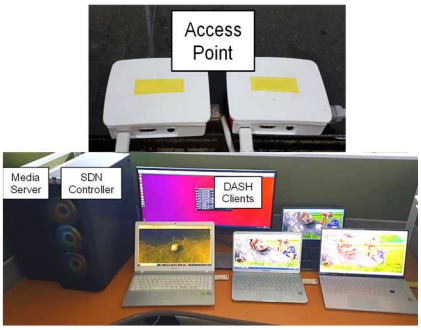


Fig. 9. Real testbed for the proposed HAS system.

clients. To support multiple connections at the DASH client, two 802.11n interfaces are configured on the laptop and are associated with different APs. DASH clients request and receive Big Buck Bunny from media server. The initial playback buffer time is set to 2 sec, the maximum buffered playback time is set to 10 sec, and time slot T^{slot} is set 2 sec. Background traffic between 35 Mbps and 45 Mbps is generated to each AP to emulate dynamic time-varying network conditions. During the experiments, in the case of DAS H.js, BOLA, and HAS-SDN, DASH clients #1, #2, #3, and #4 are connected to AP #1, and DASH client #4 is connected to AP #2. In the case of the proposed system, all DASH clients connected with two APs.

The experimental results are summarized in Table II. As shown in the table, the proposed system can provide a high-quality video streaming service as well as a stable buffered playback time compared to other existing algorithms. This is because all DASH clients of the proposed system receive segment data simultaneously via multiple Wi-Fi interfaces, and video traffic is appropriately controlled by considering the overall network and remaining playback buffer status.

IV. Conclusion

In this work, we have proposed a HTTP adaptive streaming system to maximize the overall video streaming service quality of all clients over SDN-enabled Wi-Fi APs. The proposed system employs multipath technology to overcome the limited wireless bandwidth. Moreover, SDN technology is used to effectively harmonize multiple Wi-Fi APs. To archive the goal, the SDN controller periodically collects the traffic information from the APs and DASH clients. Based on this information, the SDN controller adjusts video bitrate and schedule the segment transmission via multiple APs. The proposed system is examined by using not only simulations for large-scale Wi-Fi network environments, but also a real Wi-Fi network environment. Experimental results show that the proposed system can provide high quality video streaming services with a stable playback buffer status.

Acknowledgment

This work was partly supported by Institute for Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (No. 2021-0-00484, Core Technologies for Hybrid P2P Network-based Blockchain Services, 50%) and National Research Foundation of Korea (NRF) grant funded by the Korea government(MSIT) (No. NRF-2019R1A2B5B01070078, 50%)

TABLE II. Summary Of Performance Comparison With Existing Systems in Real-testbed environment.

System	DASH client	PSNR (dB)		Bitrate (kbps)		Playback buffer time (sec.)		Num. of buffer
		Avg.	Std.	Avg.	Std.	Avg.	Std.	underflow
DASH.js	1	34.27	7.09	566.46	645.43	4.91	1.73	5
	2	34.53	7.34	596.84	659.05	4.67	2.09	6
	3	34.94	7.14	603.25	640.52	4.59	2.22	5
	4	34.45	7.07	577.16	777.15	4.91	1.82	4
	5	43.55	2.95	1827.8	350	4.91	1.82	0
	Avg.	36.35	6.32	834.3	614.43	4.8	1.93	4.2
BOLA	1	36.38	5.38	481.05	241.38	5.83	2.84	2
	2	36.46	4.91	477.63	223.07	5.85	2.84	3
	3	36.65	4.89	492.76	231.8	5.8	2.81	5
	4	36.71	5.85	495.48	740	5.67	2.69	2
	5	43.06	3.43	1695.24	457.82	5.67	2.69	0
	Avg.	37.85	4.89	728.43	378.82	5.76	2.77	2.2
HAS-SDN	1	39.42	4.61	767.27	302.15	9.22	1.85	0
	2	39.4	4.6	764.85	303.48	9	1.86	0
	3	39.36	4.59	758.79	301.51	8.79	1.86	0
	4	39.29	4.57	749.09	300.83	8.58	1.87	0
	5	44.12	3.46	1928.48	149.26	9.39	1.85	0
	Avg.	40.32	4.36	993.7	271.45	8.99	1.86	0
Proposed	1	41.74	4.23	1221.82	433.74	7.8	1.93	0
	2	41.71	4.16	1207.27	410.11	7.45	1.93	0
	3	41.75	4.18	1223.03	426.06	7.45	1.93	0
	4	41.72	4.27	1224.39	438.28	7.28	1.92	0
	5	41.78	4.19	1229.27	431.81	7.09	1.93	0
	Avg.	41.74	4.21	1221.16	428.00	7.42	1.93	0

References

[1] ITU Report, “IMT Traffic Estimates for the Years 2020 to 2030”.

[2] Ericsson Report , “Ericsson Mobility Report data and forecast”.

[3] Cisco White Paper, “Cisco Annual Internet Report (2018–2023)”.

[4] I. MPEG, “Information technology - dynamic adaptive streaming over http (dash)”, 2019.

[5] K. Spiteri, R. Urgaonkar, and R. K. Sitaraman, “BOLA: Near-optimal bitrate adaptation for online videos,” in Proc. IEEE INFOCOMs, 2016.

[6] C. Liu, I. Bouazizi, and M. Gabbouj, “Rate adaptation for adaptive HTTP streaming,” in Proc. ACM Multimedia Systems , 2011.

[7] S. Akhshabi, L. Anantakrishnan, A. C. Begen, and C. Dovrolis, “What happens when HTTP adaptive streaming players compete for bandwidth?,” in Proc. ACM NOSSDAV, 2012.

[8] K. Kirkpatrick, “Software-Defined Networking,” Communications of the ACM, vol. 56, no. 9, pp. 16–19, 2013

[9] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, “OpenFlow: Enabling innovation in campus networks,” ACM SIGCOMM Computer Communication Review, vol. 38, no. 2, pp. 69–74, 2008.

[10] D. Bhat, A. Rizk, M. Zink, and R. Steinmetz, “Network Assisted Content Distribution for Adaptive Bitrate Video Streaming,” in Proc. MMSyS, pp. 62–75, 2017.

[11] H. Noh, H. Lee, Y. Go, H. Park, J. Lee, J. Kim, and H. Song, “Congestion-aware HTTP adaptive streaming system over SDN-enabled Wi-Fi network,” Wiley Concurrency and Computation: Practice and Experience, vol 32, issue 21, Nov. 2020.

[12] H. Helfrich and D. Zwick, “A trust region algorithm for parametric curve and surface fitting,” Journal of Computational and Applied Mathematics, vol. 73, no. 1-2, pp. 119–134, Oct. 1996.

[13] D. C. Montgomery, C. L. Jennings, and M. Kulahci, “Introduction to time series analysis and forecasting,” John Wiley & Sons, 2015.

[14] C. Bertier, M. Amorim, F. Benbadis, and V. Conan, “Modeling realistic bit rates of D2D communications between Android devices,” in Proc. ACM MSWIM, 2019.

[15] O. Johnson and C. Goldschmidt, “Preservation of log-concavity on summation,” ESAIM: Probability and Statistics, vol. 10, pp. 206–215, Oct. 2006.

[16] The network simulator 3, [Online]. Available: http://www.nsnam.org.

[17] DASH-IF dash.js player, [Online]. Available: https://github.com/DASH-Industry-Forum/dash.js/wiki.