IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

SDN Assisted Codec, Path and Quality Selection for HTTP Adaptive Streaming

REZA SHOKRI KALAN [1], MUGE SAYIT[2]
[1]International Computer Institute, Ege University, Izmir, Turkey (e-mail: reza.shokri@hotmail.com)
[2]International Computer Institute, Ege University, Izmir, Turkey (e-mail: muge.sayit@ege.edu.tr)

Corresponding author: Reza Shokri Kalan (e-mail: reza.shokri@hotmail.com).

This work was supported by the Scientific and Technological Research Council of Turkey (TUBITAK) Electric, Electronic and Informatics Research Group (EEEAG) under grant 115E449, and partially supported by Digiturk Bein Media Group.

ABSTRACT Adaptive streaming over HTTP is the dominant video streaming technology for more than a decade. HTTP Adaptive Streaming (HAS) systems provide a framework which enables clients to adapt quality with respect to network fluctuations during streaming, hence to optimize the perceived quality on the client side. Recently, network assistance is integrated with HAS in order to improve underlying network conditions and to provide network-related information to the clients. The performance of HAS systems can be further enhanced if the characteristics of the streamed video are considered. In this paper, we propose a HAS system architecture where Software Defined Networking (SDN) technology is utilized for assisting clients to select the most appropriate video codec and bitrate under the constraint of current network conditions as well as routing the video packet over the appropriate paths. In the proposed architecture, layered video is used, where each additional layer improves the quality. The controller estimates the packet loss probability by taking video codec characteristics, the bitrates of the layers and network capacity into account. Based on these estimations, the controller selects the appropriate codec type and video quality for the clients and manage the network. Simulation results show that the performance of the video streaming architecture can be improved significantly when codec, quality and path selection are jointly considered, and combined with SDN flexibility and advantageous.

INDEX TERMS
Codec, DASH, DANE, HAS, SDN

I. INTRODUCTION

BEING one of the most popular application types used on the Internet, video streaming applications offer a wide range of usage scenarios, from live video streaming services to distribution of personal videos of users. Cisco's forecasting reports state that nearly half of total devices will be video capable in 2022 and the percentage of the video packets that will be transferred over IP will have reached to 82% by then [1]. While emerging network technologies such as 5G, Software Defined Networking (SDN) and Network Functions Virtualization (NFV) enable an infrastructure that provides high connectivity and low latency, the requirements of the future multimedia applications have been increasing on the other hand. As well as maximizing the underlying bandwidth capacity, minimizing latency is very important for the applications like Augmented Reality/Virtual Reality (AR/VR) implementations or interactive multimedia systems.

For almost a decade, HTTP Adaptive Streaming (HAS) has become a dominant technology for streaming video on the Internet. In HAS systems, more than one representation of the same content is encoded at different bitrates in order to enable smooth quality adaptation on the client side during streaming. While HTTP provides utilization of the web-caches and uses a reliable end-to-end transport infrastructure due to TCP, quality adaptation enables the selection of optimal quality under the constraint of network conditions and client side parameters. In order to provide interoperability between HAS systems developed by different vendors, Dynamic Adaptive Streaming over HTTP (DASH) standard was proposed by MPEG working group [2]. DASH has codec and format agnostic nature and can be applied with any media format [3].

In HAS systems, the general approach to produce quality alternatives, i.e. representations, is to have non-layered encoded files by using a codec such as H.264 Advanced Video

Codec (AVC). In this case, there is one encoded video file for each quality level. Another option for generating the alternative qualities of the same video file could be the usage of a layered video codec such as Scalable Video Coding (SVC) or Multiple Description Coding (MDC). With layered coding, all quality levels can be obtained from a single encoded file. The use of a layered video in a HAS system provides network bandwidth utilization and cache storage efficiency [4]. Although the use of SVC in commercial systems has not been preferred until today due to its higher overhead on the bitrate compared to AVC, recent developments in video codec standards have paved the way for the use of it. The newest video codec standard, Versatile Video Coding (VVC-H.266), was finalized in July, 2020 by ITU-T VCEG and ISO/IEC MPEG groups. It is expected that layered multi-stream and scalability will be used in the commercial system thanks to H.266, which provides up to 50% bitrate saving over High Efficiency Video Coding (HVEC) (H.265) [5].

SDN technology can be a good alternative for network operators that offer services to video streaming companies [6], [7]. Since it's possible to design application specific network solutions thanks to the decoupled data and control plane architecture of SDN, this technology can be used for increasing the performance of HAS applications. HAS clients have limited knowledge about network conditions, so they may suffer from video freezes or under utilization of available bandwidth and may not get the best possible video quality under the current conditions [3], [8], [9]. Recently proposed approaches for enhancing the performance of the video streaming applications shift toward network-assisted approaches and SDN is one of the most preferred technology in these systems [10], [11], [12].

The streaming paths used for transferring the video packets, quality adaptation techniques and video codec type jointly affect the performance of the video streaming applications. Hence, these parameters should be carefully considered when designing a video streaming system architecture. In this study, we propose a video streaming system architecture that utilizes SDN technology to determine jointly video codec type, quality and streaming paths. The advantages of different codecs according to their characteristics are considered in this study. In the proposed system, the SDN controller has some knowledge about HAS characteristics. It utilizes underlying network information and HAS-related knowledge to select the video codec type for the clients and the optimal number of layers that the clients should request. Although SDN controller can collect the network statistics in real time, providing a good level of performance is not an easy task due to the dynamic nature of HAS and network conditions, as well as different video codec characteristics. In this work, codec type, streaming paths, and optimal quality selections are made by considering the latency at the application layer. For this purpose, the packet loss rate in the link layer is estimated and codec and quality selections are made in a way that the delay of packets due to TCP retransmissions is minimized.

The SDN controller uses an optimization model to select video codec type, the number of layers and streaming paths for each layer when a client joins the system. The controller also runs an event-based heuristic algorithm, which re-determines the number of layers and streaming paths with respect to the network fluctuations during the streaming session. Clients run a rate adaptation algorithm which interprets the recommendations sent by the controller, which are the outputs of the optimization and heuristic algorithms.

The contributions of this study can be listed as follows:

- We propose an approach that considers the characteristics of the video codecs and underlying network conditions. To the best of our knowledge, there has not been any previous study on the joint selection of the optimal number of video layers and streaming paths by taking into account video codec type and network conditions.
- We define formulas for estimation packet loss ratios at the link layer. These estimates take into account video layer dependencies and the effects of the lost layer packages on other video layers.
- We utilize packet loss estimations for assigning the optimal number of video layers. For this purpose, we estimate the video packets that is expected to be delayed and select the layers so that the latency is minimized for live video streaming applications.
- We propose a new rate adaptation algorithm for HAS clients, which can interpret the SDN controller's comments and act accordingly. With presented comparative performance results, we demonstrate the performance gain that can be achieved by *(i)* HAS aware network assistance and, *(ii)* network assistance aware client implementation.

The rest of the paper is organized as follows: In Section II, background on SDN and HAS based systems with related works is given. The details of the proposed architecture, the heuristic algorithm, and the client implementation are presented in Section III. The comparative performance results are provided in Section IV. Finally, conclusions are given in Section V, which is followed by the reference list.

II. BACKGROUND AND RELATED WORKS

A. BACKGROUND

1) Software Defined Networking

Conventional network architecture which relies on the vertical design, where data and control planes are bundled together, makes efficient network configuration a difficult and complex task. SDN technology, which separates control and data planes, overcomes this complexity and gives more agility to network functions [13]. It transforms hardware and device-centric network architecture into a flexible, virtual, and programmable form that provides high agility and rapid innovation in network services. SDN is seen as a promising technology in future networks for its advantages and functionalities that can meet various types of demands and requirements of future Internet applications. The main functionalities of the SDN technology rely on the communication

between forwarding devices and the controller via an open standard interface. The OpenFlow protocol [14] is the most popular communication protocol for exchanging messages between the switches and the controller.

2) HTTP Adaptive Streaming Characteristics

In the architecture of HAS applications, a video file is encoded at various bitrates, which, in turn, encoded video files are produced at various qualities. The encoded video files are partitioned into N chunks known as *segments*, with each segment carrying t seconds of the video. Encoded video files are called *representations*. A manifest file, which is called Media Presentation Description (MPD), keeps the information about media content such as the bitrates of the representations and URLs of the segments. The clients use this information in the file to request the selected segments after downloading the file at the beginning of the streaming session.

The quality adaptation is provided by the selection of segments with different qualities over time. For the quality adaptation, the client runs a rate adaptation algorithm which determines the quality of the segments to be requested. The rate adaptation algorithms can be classified into two main classes with respect to criteria that is used for the selection of the quality: Throughput Based Adaptation (TBA) [15], [16], and Buffer Based Adaptation (BBA) [17], [18], [19]. Beside TBA and BBA, hybrid models [20], [21] also have been proposed. The recent studies in this area focus on adaptive playback speed while jointly adapting the quality [22], [23].

As well as non-layered video codec usage, there are several HAS architectures proposed in the literature that use a scalable video codec such as H.264 SVC. SVC codec is an extension of ISO/ITU advanced video coding (H.264/AVC) standard, which produces video layers by using spatial, temporal and Signal to Noise Ratio (SNR) scalability [24]. With scalable video coding, the video files encoded at different bitrates consist of a base layer and one or more enhancement layers. While the base layer has the lowest bitrate and the lowest quality, each additional enhancement layer increases the bitrate and consequently improves the quality of the encoded video. The base layer is the most important layer and should never be lost, because, in order to decode an enhancement layer, the base layer and the all previous enhancement layers are required.

Same as SVC, MDC encodes the video into layers called descriptions [25]. Similar to SVC, each description received by the client improves the video quality. However, in contrast to SVC, the description layers are self-decodable and each layer carries basic information to decode the video independently. This characteristic of MDC causes extra overhead compared to SVC since each description contains redundant information to be decoded without requiring other descriptions. Thus, MDC is more tolerant of loss compared to SVC, but SVC provides better compression.

The advantages of the layered video codecs have led some researchers to design streaming systems that use layered video over emerging network architectures using them such as SDN [26], NFV [27], 5G networks [28] and mobile edge computing [29], and to propose to use layered video in future video applications such as AR/VR. Authors in MS-Stream [30] presents an effective solution for enhancing the perceived quality for DASH clients, by using the different multiple descriptions sent by multiple sources. They showed MDC increased the performance of DASH applications. In [31], the authors presented a cost-effective DASH system utilizing MS-Stream. In AR/VR applications, when the users move their head and change the viewport, the new viewport data should arrive to the users quickly in order to prevent motion sickness. In [32], the authors propose to use layered video for DASH based multicasting system in order to minimize latency in AR/VR applications. In the next section, we give the literature review on layered video streaming over SDN.

B. HAS ARCHITECTURES BASED ON SDN AND SAND

Because HAS is a client-driven architecture, it has limited information and view about network conditions and other clients' behavior. Also, service providers do not have the control over the client's behavior, therefore they may not be able to guarantee a high level of service quality with client based adaptation. Since the performance of the multimedia systems in terms of Quality of Experience (QoE) mostly relies on the underlying network conditions and server characteristics such as availability and distance, there is a tendency to get assistance from the network to overcome the limitations of client based architectures. SDN is a good alternative to enhance the performance of video streaming applications and to provide network support to HAS systems.

There are many studies that propose video streaming services over SDN in the literature. In one group of studies, SDN is utilized for selection of suitable streaming paths to transfer video packets from the server to the clients, thanks to the flexibility of SDN on determining flow routes [33], [34], [35] [36]. Network resources and bandwidth allocation strategies which consider parameters specific to HAS systems are also studied widely in the literature [37], [38], [39], [10]. Path assignment and resource allocation approaches proposed in the literature do not consider any strategy related to the layered video characteristics.

In the studies given above, there is no communication between the controller and the video streaming clients. In another group of studies that utilize SDN technology, there are several approaches proposed, where the controller explicitly signals the clients to assist quality adaptation in order to increase the performance of video streaming applications [40], [41], [42], [43]. These studies mostly focus on the quality selection process, any SDN module or SDN based architecture aware client rate adaptation algorithm is not proposed in these set of studies. Furthermore, layered video transmission is not addressed in any of them.

MPEG group has been working on Server and Network Assisted DASH (SAND) proposal [44]. In order to provide a centralized control on quality adaptation for network and

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3114299, IEEE Access

service providers, SAND introduces an architecture that offers asynchronous network-to-client and network-to-network communication. Network elements receive QoE metrics from the clients and returns network feedback measurements, which can be used by clients in their adaptation algorithm. In SAND architecture, there are DASH-Aware Network Elements (DANEs), which are the components that have the knowledge of DASH characteristics. They can be used to optimize network resources for DASH video traffic in order to improve the user's QoE. SAND also defines control messages between DANE and clients, and between multiple DANEs. SDN technology can be utilized in SAND architecture [45]. DASH clients may directly select the quality recommended by an SDN controller [46], use bandwidth information sent by an SDN controller [47] or connect virtualized DANEs which were managed by an SDN controller [48], [27]. None of these studies related to SAND in the literature focus on video codec characteristics.

Layered video streaming over SDN investigated in several studies in the literature. OpenFlow based video streaming system for streaming SVC video between multi-server and multi-client is specified in [49]. Transferring base layer and enhancement layers over different streaming paths by using UDP are proposed in [26], [50], [51], [52], [53], and [54]. These studies focus on transferring SVC layers over selected streaming paths and determining the suitable number of layers by taking UDP characteristics into account. The authors focus on providing QoE fairness among HAS clients, where different SVC layers can be downloaded from different servers in [55]. In [56], the SDN controller suggests the appropriate SVC layer for DASH clients to request by using a machine learning approach that takes network conditions as input. Constructing a multicast tree for each SVC layer for multicasting DASH traffic over SDN is proposed in [58]. SDN technology can also be utilized to transfer MDC coded video over different paths. In [57], a video streaming architecture that addresses MDC coded video distribution over SDN is proposed. The layers of MDC are forwarded over different multicast trees constructed by the SDN controller. The authors propose to serve different Quality of Service (QoS) class users by adjusting the number of MDC layers sent to each service class [57]. In Table 1, we show the main differences of our work from the literature. In this study we determine the optimum number of layers by considering the dependencies of layers that are sent over different paths. We also assign different codecs with respect to the current network conditions. These are the unique characteristics of our work.

Most of the prior works that implement a video streaming architecture over SDN focus on selecting the streaming paths or selecting the quality for the clients. None of them focuses or considers the video codec type when determining the streaming paths. Even for the studies that are related to layered video, the layer dependencies are not considered in the selection of the paths. In our previous work [59], we proposed an optimization model that selects video codec type and the optimal number of layers for the client which newly joined the system, where the codec type and layer selections stay constant during streaming. In this work, we enhanced our previous work [59] by adding a new module to the controller that provides to determine the optimal number of video layers by considering network fluctuations. We also modified the metrics that are used for determining the optimal number of layers and codec type. The overview and design details of the proposed video streaming system are given in the next section.

III. PATH, CODEC AND QUALITY SELECTION FOR LAYERED VIDEO STREAMING

The details of the proposed system designed for layered video streaming by using two types of layered video codecs, namely SVC and MDC, are given in this section. We present the system architecture overview before giving the details of the path and codec selection approach.

A. ARCHITECTURE OVERVIEW

The network architecture used in this study is given in Figure 1. In the architecture, the system consists of DASH server, clients, and forwarding devices that performs packet functions such as dropping, modifying and forwarding packets to a specific port or controller. Besides these actions, forwarding devices send link statistics to the controller periodically. The middle layer in the figure represents *Network Operating System* (NOS) or controller. The controller consists of two groups of main function modules. While the basic functions are used for essential network services such as link and path management, the advanced functions are related to video codec type and optimal layer selection. In order to run these modules, the controller gathers network statistics from the forwarding devices and internal parameters from the clients such as buffer level and requested video representation. This information collection module can be implemented as an external module, which runs at a different network element, in a DANE to enable large-scale deployment. In this case, the controller can communicate with DANE through its northbound. Finally, the upper layer represents network service applications. Applications in this layer define the routing policies which are ultimately translated to forwarding rules and are sent via OpenFlow commands to the switches to program their behavior. The proposed framework consists of a HAS application aware SDN controller, a DASH server which encodes the video files with SVC and MDC codecs to produce different representations, and DASH clients that can interpret the commands received from the controller and have the capability of decoding SVC and MDC encoded videos. When a new client joins the network and establishes the TCP connection with the server, its request is handled by the first-hop switch. Since there is no entry for a new client in the switch's routing table, the switch forwards the client request to the controller by sending a *PACKET-IN* message to learn the forwarding rule for that client. In the controller, *Host Manager* module detects newly joined client when it

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3114299, IEEE Access

TABLE 1. Comparison of different studies.

	path selection for each layer	layer dependency	use of different codecs	quality selection assistance	layer priorities
Xue [49]	✔	✗	✗	✗	✔
Egilmez [26]	✔	✗	✗	✗	✔
Laga [50]	✔	✗	✗	✗	✔
Ergiz [51]	✔	✗	✗	✗	✔
Gangwal [52]	Partial	✗	✗	✗	✔
Yue [53]	Partial	✗	✗	✔	✔
Uzakgider [54]	✔	✗	✗	✔	✔
Tashtarian [55]	✔	✗	✗	✔	✔
Ozcan [56]	✗	✗	✗	✔	N/A
Noghani [57]	✔	N/A	✗	✗	N/A
Our work	✔	✔	✔	✔	✔

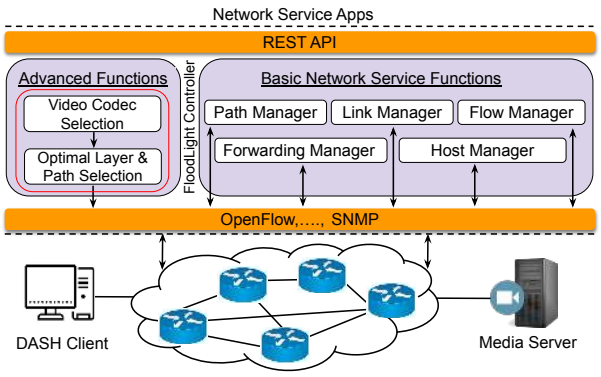

FIGURE 1. SDN controller modules and system architecture. The advanced functions, which are Video Codec Selection and Optimal Layer & Path Selection, are the modules developed in this study.

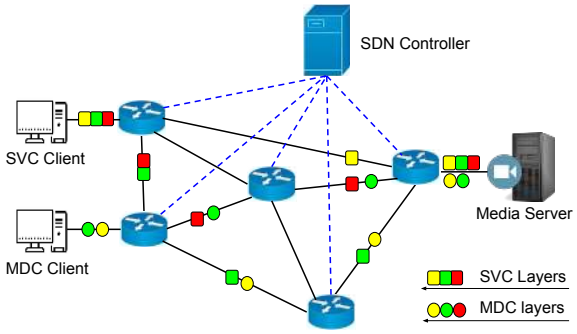

FIGURE 2. An illustration of the video layer dissemination over the links. Different layers of different codec type are sent over different paths.

receives the *PACKET-IN* message. After that, the controller runs an algorithm that determines the video codec type, the optimal number of layers, and the streaming paths of each layer. It uses the current network conditions as the input to the algorithm. Based on this information, it calculates the likelihood of the packet losses at the link layer by considering the bitrates of the video layers and available bandwidth information of the paths that the layers will be transferred over. It then determines the video codec related parameters such as video codec type and the optimal number of video layers by running an optimization model which aims to minimize packet losses at the link layer and maximizes video bitrate. Note that, we focus on the estimation of packet losses at the link layer since the packets are not lost at the application layer due to the reliable transfer mechanism of TCP. The controller signals the clients via REST API to direct them according to the output of the algorithm. At the same time it sends the flow route information to the switches via OpenFlow protocol on the southbound API. The server sends each layer of the encoded videos via connections opened over different ports. Hence, the controller can determine different streaming paths for each video layer by using the port information defined for each layer on the server side. An example scenario that shows the dissemination of the video packets belonging to different layers of different codecs is given in Figure 2.

During the streaming session, the clients select appropriate video quality by considering the observed network conditions and internal parameters. However, the SDN controller also helps the clients about the decisions on the quality. In the proposed architecture, the controller periodically measures the traffic amount on each streaming path and if the traffic pattern is changed, it runs an heuristic algorithm within its *Optimal Layer and Path Selection* module in order to determine the number of layers to be requested by the clients based on the current network conditions. The rate adaptation algorithm does not allow clients to request higher quality than the one recommended by the controller. This prevents the tendency of clients to request higher video quality than the network can transmit, and severe quality degradation is proactively eliminated. The details about the rate adaptation algorithm are given in section III-D.

B. ESTIMATION OF THE PACKET LOSS RATIO

The packet losses at the link layer affect the received video quality due to the latency introduced by TCP retransmissions. If a packet carrying enhancement layer data arrives later than its playout time, it's discarded at the application layer. Basically, these packets are lost at the application layer although they are received by the clients. Different video codec types affect packet losses at the application layer differently because of the differences in the layered structures. Table 2 shows two scenarios where the same video is encoded with

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3114299, IEEE Access

TABLE 2. Example of SVC and MDC codec behaviors

	Received Layers				# of Layers Played	
Scenario	L1	L2	L3	L4	SVC	MDC
1	+	-	+	-	1	2
2	+	+	-	+	2	3

both SVC and MDC codecs. In the table, $+$ represents the layer is received timely by the client while $-$ represents the layer is not received before its playout time, i.e. it cannot be played and is discarded by the client. As can be seen from the table, even if the same video layers are received by the client, the client plays the video with higher quality with MDC codec. It is because, unlike MDC codec, in the SVC codec, if a packet belonging to layer n is lost, other packets belonging to higher layer than n cannot be decodable even if they arrived timely. When we examine the number of layers received by both client types, MDC seems more advantageous because the clients can get higher quality. However, MDC layers have higher bitrate when compared to those of SVC layers for the same quality level, hence MDC requires higher bandwidth.

The packet loss ratio is relevant to the available bandwidth of the streaming path, the number of video layers and the bitrate of each layer that will be transferred through that path as well as video codec type as explained above. Suppose abw represents the available bandwidth of a path p and n denotes the number of video layers which are transferred through the path p. The available bandwidth of the path is the remaining bandwidth when the traffic amount of UDP and non-HAS TCP flows is subtracted from the original capacity. TCP fairness ensures that the bandwidth portion that can be used to transfer each layer roughly equals to abw/n on the path p. The controller calculates the expected packet loss ratio of the transmitted video layer packets by using the following formula:

$$plr_L = 1 - \begin{cases} 1 & if\ \frac{abw}{n} \geq bitrate_L \\ \frac{abw}{n * bitrate_L} & otherwise, \end{cases} \quad (1)$$

where plr_L and $bitrate_L$ represents the expected packet loss rate of the packets of layer L and the bitrate of video layer L, respectively.

The packet loss rate given in (1) indicates the loss probability of an arbitrary packet belonging to a TCP flow. The lost packets are re-transmitted by TPC, hence the delay of the packets are increased. We refer to packets that are arrived later than their playout time and discarded at application layer as the lost packets at this layer. When we consider the video packet losses at the application layer due to this retransmission delay, we should also consider the layer dependencies especially for SVC coded video. If an SVC layer packet is lost, then the upper enhancement layers of this lost packet can not be decoded on the client side, hence they are also treated as lost packets. Therefore, in the calculation of packet loss probability, codec type should also be considered. The packet loss probability also depends on the number of flows that shares the same streaming paths and the bitrate of the video layers because packets losses are highly related to the capacities of the paths and the traffic amount transferred over those paths. Let c_type represents the codec type and L is the layer number. And let plr_{L,c_type} is the packet loss rate in each layer, which is calculated by considering layer dependencies. plr_{L,c_type} equals to:

$$\begin{cases} plr_L & if\ c\_type = MDC \\ \sum_{k=0}^{L-1} plr_k \prod_{k=0}^{L-1}(1 - plr_k) * plr_L & if\ c\_type = SVC. \end{cases} \quad (2)$$

While (1) gives the expected packet loss ratio for a specific layer, (2) calculates estimated total packet loss ratio for all layers affected by the lost packet. In our previous work, we provided a formula for estimating of packet loss ratio by considering the layer dependencies for the video layers transferred over the same paths [59]. In this current work, we generalize this formula by also considering the layers of the same client, being sent over different paths. This is important because the losses of layers sent over different paths also affect each other. The Optimal Layer and Path Selection algorithm considers dependencies between layers that are transferred over different paths. We will give the details about this algorithm in section III.C.

C. THE SELECTION OF CODEC TYPE, LAYERS AND STREAMING PATHS

The selection of the video codec type is done by the controller at the beginning of the video streaming session. The selection of the codec type is done only at the beginning of the streaming session because changing the codec type during streaming would not be practical. In this initial stage, the controller also determines the optimal number of layers and streaming paths by considering the selected video codec type and current network conditions. The optimal number of layers and streaming paths are re-determined by the controller during the streaming session while video codec type stays the same.

1) Video Codec Selection at Session Startup

When a client joins the network and starts the video streaming application, the request sent by the client to establish a TCP connection is grabbed by the controller. Therefore, the controller detects that a new client joins the system and it triggers the *Video Codec Selection* module. The *Video Codec Selection* module is responsible for selecting the codec type and the optimal number of layers for the newly connected client by considering the current network conditions. The selection of the codec and the optimal number of layers is determined by an optimization algorithm. Before running the optimization model, the controller runs another algorithm,

Algorithm 1, for calculating the estimated packet loss ratios to be given as one of the inputs to the optimization model. As mentioned previously, the packet losses affect differently to SVC and MDC encoded video due to the characteristics of the codecs.

The *Algorithm 1* calculates the total estimated packet loss for each layer based on (2) for both video codec types. Paths are selected for the layers of both codec since the codec type is not determined yet for the newly joined client at this stage. Note that, the paths are selected virtually in order to calculate packet loss estimations. At the beginning, the algorithm initializes the path set for both codec types. The algorithm selects the path with maximum available bandwidth for each layer with respect to the codec type in the *for* loop. After each time a path is selected, the available bandwidth of the selected path is re-calculated by considering the bitrate of the video layer. The estimated packet loss is also calculated each time a path is selected for a layer. As the output, the total estimated packet loss ratios for each layer of both codec types are provided.

The *Video Codec Selection* module runs an optimization model after obtaining the estimated packet loss ratios provided by *Algorithm 1*. The optimization model is given as follows:

$$min \quad \sum_{L=0}^{L_{max}} plr_{L,c\_type}, \tag{3}$$

$$max \quad L, \tag{4}$$

s.t.

$$L \leq L_{max}. \tag{5}$$

Algorithm 1: Packet Loss Estimation Algorithm

Input:
P : the set of paths between the client and server
Bitrate of each layer for both codec types: *SVC, MDC*
$P_{\mathrm{Max-abw}}$: the path with maximum available bandwidth, is updated during the algorithm runs
Output: $plr^{total}_{l,SVC}$, $plr^{total}_{l,MDC}$, $\forall l \in K$

initialize $plr^{total}_{l,SVC}$, $plr^{total}_{l,MDC}$
foreach *Video Codec Type* **do**
  initialize P
  foreach *Layer l* **do**
    $Path_{\mathrm{p}} \leftarrow P_{\mathrm{Max-abw}}$
    $abw_{\mathrm{Path_p}} \leftarrow abw_{\mathrm{Path_p}}$-$bitrate_{l,c\_type}$
    $plr^{total}_{l,c\_type} \leftarrow plr^{total}_{l,c\_type} + plr_{l,c\_type}$
  end foreach
end foreach

The equations given in (3) and (4) represent the objectives of the model. The model aims to minimize packet loss ratio while maximizing the video quality by increasing the number of the layers. The constraint of the model is given in (5), that shows the number of layers, L, cannot be higher than maximum number of layers, L_{max}. Although it is a multi-objective optimization model, the search space is not large, as encoded video files usually contain between 3 and 7 layers. Hence, the controller can search all the solutions in the space to find the optimal solution within a very limited of time.

The *Packet Loss Estimation Algorithm* runs in order to determine the video codec type and the number of layers at the beginning of the streaming session. The selected codec type and quality are signaled to the client and the client starts requesting segments based on this initial configuration. In the next section, the *Optimal Layer and Path Selection Algorithm (OLAPS)* that determines the quality and the paths by using the packet loss ratio estimation formula is given.

2) Layer and Path Selection During Streaming

As explained in the previous section, when a client starts the video streaming application, it requests the segments of the video encoded with the video codec type recommended by the controller. The controller also determines the optimal number of layers at the beginning of the video streaming session. However, since network conditions are dynamic due to the cross traffic, fluctuations in available bandwidth may cause quality switches at the client side. Hence, the number of layers that are determined at the beginning of the session may not be optimal any longer. In order to cope with network dynamism and to determine up-to-date optimal number of video layers, the controller should re-new the path assignments and layer selections. The problem of assigning paths to a set of base layer and enhancement layers so that each client can get the highest number of layers possible is an instance of a generalized bin packing problem [60]. Therefore, the problem is NP class. Hence, we develop OLAPS, an heuristic algorithm, to solve the assignment problem.

In order to detect changes in network conditions, the controller periodically checks the changes in traffic amount on the paths and runs the OLAPS algorithm when network conditions considerably change. The controller measures the available bandwidth of each path and triggers the OLAPS module to run the algorithm if the measured value is above of a pre-defined threshold value. This threshold can be set by the network operator. The bitrate of the minimum representation could be a good alternative for this threshold since the changes in traffic amount higher than this bitrate value may cause considerable quality changes in the received video.

The purpose of the OLAPS algorithm is to determine the number of video layers that are sent to all clients in the system and the streaming paths for each layer. The OLAPS algorithm determines these items by taking into account available bandwidths of the end-to-end paths, bitrate of video layers, layer dependencies and the total number of the layers transferred through each path. In (2), the estimated packet

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3114299, IEEE Access

loss for a layer L that is transferred over a path p is given by considering the packet loss ratio of path p and codec type. However, the packets of a layer can also be lost because of lost packets of its underlying layers. Therefore, the layers which are sent over different paths should also be considered in the calculation of the estimated packet loss ratio for each layer. Figure 3 shows a scenario to explain the reason of why we should take into account the dependencies between the layers that are transferred over different paths in the packet loss calculations. In the figure, L_i, L_{i+1}, and L_j represent layer i and layer $i+1$ of the video sent to the client 1, and layer j of the video sent to the client 2, respectively. The flow of L_j may cause loss of client 1's i^{th} layer packets since these two flows share the same path. As a consequence, packet loss in layer i affects to the $(i+1)^{th}$ flow due to layer dependency. Hence, when packets of a layer are lost, they also affect the same client's upper layers and should be considered in the calculation of packet loss probability. Note that, the loss of the layer packets transferred over different paths is considered for only SVC layers due to its layer dependency.

The OLAPS algorithm, which is given in *Algorithm 2*, works in two phases. In the first phase, which is run in the first *for* loop, the algorithm allocates paths for the packets of the first layer for each client in order to ensure that each client receives the video at least with minimum quality. For SVC, the first layer is the base layer due to the layer dependency rules, while an arbitrary layer can be selected as the first layer for MDC coded video. In each iteration, the algorithm assigns a path with maximum available bandwidth for each client and updates path information.

In the second phase, the streaming paths for additional layers are assigned. But in this phase, a path for each additional layer is assigned by considering the likelihood of the packet losses. The paths for additional layers are determined in ordered, in other words, new path assignments for the new layers only start after the paths for the same number of layers are assigned for all clients in the system. For each layer l, the path with maximum available bandwidth ($P_{Max-abw}$) is assigned for that layer, the available bandwidth value (abw_{P_l}) is re-calculated and the estimated packet loss ratio, plr_{l,c_type}, is estimated by using the formula (1) with respect to the codec type. Note that, the codec types (c_{type}) are already determined by *Algorithm 1* for each client and OLAPS algorithm ensures that the paths are selected with respect to the codec type of each client. If the estimated packet loss ratio is greater than a certain threshold, which is determined differently for each layer based on their importance, the path assignment is canceled. Let p is the path that is selected for transferring the packets of layer l and thr_l represents the threshold determined for layer l. When the path p is assigned for this new layer l, the controller estimates the effect of the newly assigned layer l to all layers routed via the same path p and if estimated packet loss ratio is higher then the threshold defined for the related layer, the path assignment is canceled. Furthermore, for SVC clients whose layers are transferred over path p, the packet loss ratios are checked for also their other layers transferred over different paths due to the scenario given in Figure 3.

According to the second phase of the algorithm, if adding the new layer l for the selected path does not cause unacceptable estimated packet loss effect on other clients' layers on the same path and other layers of those clients on the other paths, then the selected path is assigned for the layer l. Hence, in the second phase of the algorithm, a path for each additional layer is assigned only if assigning a path for a new layer does not affect the layers by considering the flows on the same paths and related flows in different paths.

The controller runs the OLAPS algorithm periodically when the network conditions change, i.e. a new flow arrives or a flow terminates. The controller signals clients and switches with updated information according to the output of the algorithm. Typically, when a video is encoded with a layered coding, the number of layers is limited and this number can be considered as a constant. The algorithm orders the paths according to their bandwidth at the beginning. The algorithm loops for each client, layer and the flows on the path that is selected for the current client. The number of layers and the number of HAS layer flows on a path is constant. Therefore, the complexity of the proposed algorithm is $O(c + nlogn)$, where c is the number of clients and n is the number of paths.

D. RATE ADAPTATION ALGORITHM FOR SDN-ASSISTED VIDEO CLIENT

The purpose of the rate adaptation algorithm is to maximize video quality on the client's side by downloading video segments with the highest quality possible while minimizing re-buffering duration. The newly joined client downloads the first several segments from the lowest bitrate at the beginning of the streaming, which is a typical approach in such system in order to minimize startup delay. After the buffer fullness value reaches to a certain level, it starts requesting segments of the layer recommended by the SDN controller at the beginning of the streaming session.

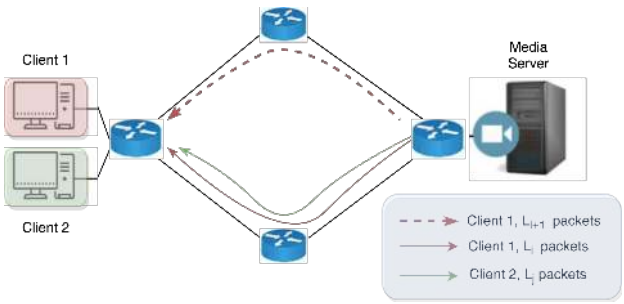

FIGURE 3. A scenario that shows different SVC layers are sent over different paths. Packet losses in L_i might cause to loss L_{i+1}, even L_{i+1} is routed over a different path.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3114299, IEEE Access

Algorithm 2: Optimal Layer and Path Selection (*OLAPS) Algorithm.*

Input: thr_{L}: threshold level determined for layer l
$P_{\mathrm{Max-abw}}$: the path with maximum available bandwidth, is updated during the algorithm runs
Output: Optimal number of video layers and the streaming paths
foreach *client* c **do**
  $P_{\mathrm{base}} \leftarrow P_{\mathrm{Max-abw}}$
  Assign selected path, P_{base} for $baselayer$, client c
  $abw_{\mathrm{P_{base}}} \leftarrow abw_{\mathrm{P_{base}}}$ - $bitrate_{baselayer,c\_type}$
end foreach
foreach *layer,* l **do**
  foreach *client,* c **do**
    $P_{\mathrm{l}} \leftarrow P_{\mathrm{Max-abw}}$
    Assign selected path, P_l for layer l, client c
    $abw_{\mathrm{P_l}} \leftarrow abw_{\mathrm{P_l}}$ - $bitrate_{l,c\_type}$
    Calculate($plr_{\mathrm{l,c\_type}}$)
    if *(*$plr_{\mathrm{l,c\_type}} <= thr_{\mathrm{L}}$*)* **then**
      foreach *layer i transferred over the same path* **do**
        if *(*$plr_{\mathrm{i,c\_type}} = 0$*)* **then**
          Continue
        else
          if *(* $0 < plr_{\mathrm{i,c\_type}} <= thr_{\mathrm{i}}$*)* **then**
            foreach $j > i$ *transferred over different paths,* $c\_type = SVC$, $\forall client \neq c$ **do**
              Calculate($plr_{\mathrm{j,SVC}}$)
              if *(*$plr_{\mathrm{j,SVC}} <= thr_{\mathrm{j}}$*)* **then**
                Continue
              else
                remove assigned path
                Break
              end if
            end foreach
          else
            remove assigned path
            Break
          end if
        end if
      end foreach
    else
      remove assigned path
      Continue
    end if
  end foreach
end foreach

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3114299, IEEE Access

As explained in the previous section, in the proposed SDN-assisted system, the controller sends recommendation messages for the number of layers that can be requested by the clients as the output of the OLAPS algorithm during the streaming session. An application that runs on the northbound of the controller is responsible for sending the recommendations to the clients. Note that, this part can be easily moved to a video streaming company's DANE server. In such architecture, the controller can send the set of recommendations to the DANE so that the DANE connects to the clients and forwards the recommendations. Hence, if the number of clients is high, the scalability of the systems is preserved and the controller does not become a bottleneck point.

Let L_{rec} represent the number of layers that is recommended by the controller. When L_{rec} value is received from the controller, one option for the clients could be to start requesting the L_{rec} layers until the controller recommends another quality layer, i.e., layer with higher or lower bitrate. However, the clients should also consider their current buffer fullness level, which is one of the crucial internal parameters that affects outage or re-buffering events.

In a typical buffer based adaptation algorithm there is a mapping between buffer occupancy and video representation, such that as buffer fullness increases, clients starts to request video from higher representations. Conversely, when the buffer fullness decreases, clients request video segments of lower representations. In this current study, the clients use such mapping algorithm and determines a quality based on buffer fullness value. They also take into account the quality recommended by the controller. The quality is selected as the layer having minimum bitrate among the recommended layer from the controller and layer determined based on buffer level.

Let L_{n} represent the number of layers that is determined by the client, which equals to the possible highest quality that can be received under the constraint of buffer occupancy. In the mapping approach used for this purpose, buffer is divided into equal levels and a quality is defined for each level. Accordingly, if the buffer fullness is at the lowest level, then the lowest quality is determined. And if the buffer fullness is at the rightest region in the buffer level, then the highest video quality is determined by the mapping function. The recommended layer by the controller (L_{rec}) may be higher or less than L_{n}. When a new recommendation is received from the controller, the client requests the segments of L_{rec} if L_{rec} is lower than L_{n}. On the contrary, the client might send a request for receiving L_{n} layer packets when the buffer fullness is below a certain threshold and L_{n} is less than L_{rec}. This approach has several benefits. Firstly, it helps to avoid re-buffering since the rate adaptation algorithm considers both internal information about buffer level and external information provided by the assistance of the controller. Secondly, the algorithm eliminates to put additional burden on the links and provides fair bandwidth allocation since L_{rec} is determined by OLAPS algorithm, which determines the equal number of layers for each client. And finally, the number of quality oscillations on the client side is reduced because the clients do not exposure unexpected buffer drains, which are the one of the main reasons of oscillations in requested video qualities.

IV. PERFORMANCE EVALUATION

A. TESTBED AND TOPOLOGY SETUP

For the performance evaluation of the proposed approach, we used Mininet emulator to setup an SDN environment and to run the tests. Mininet provides an efficient platform for constructing SDN topologies, implementing and testing SDN applications [61]. The controller modules are built on top of the Floodlight software. OpenFlow is used to provide communication between the controller and switches. We run our experiments over a real world topology, known as "*Compuserve*", whose information is taken from the *Internet Topology Zoo* [62].

During the simulations, we used four different network scenarios by using *Poisson* distribution to generate the links bandwidth with mean values of λ=8 Mbps, λ=10 Mbps, λ=12 Mbps, and λ=15 Mbps. Total available bandwidth values between source and destination based on the defined mean values (λ) are illustrated in Table 3. Video codec selection module selects one of the video codec type, SVC or MDC, for the clients. We observed the codec type selection of OLAPS algorithm in the simulations. Accordingly, on average, when the network resources are limited, i.e., λ equals to 8 or 10, the number of MDC and SVC clients are the same. For the higher values of λ, the number of SVC clients roughly equals to twice the number of MDC client. In order to analyze the performance of the system under limited bandwidth resources, which shows how well the system adapts to the current conditions, clients were placed behind bottleneck links.

Elephants Dream-II [63] is used as the streamed video. The video file consists of 327 segments, each with a duration of 2 seconds. On the client's side, the total buffer length is set to 24 seconds. The clients start to play video after buffering 8 seconds of video at the beginning of the streaming session. There are 10 video clients, which are capable of decoding both SVC and MDC video, connected to the system. The video server provides the same video with both SVC and MDC codecs. The SVC video has one base and two enhancement layers, while MDC video has three layers. Table 4 gives the bitrate distribution of the layers for both codec types. The

TABLE 3. The bandwidth capacities of the network links.

Mean Value (Mbps)	**λ=8**	**λ=10**	**λ=12**	**λ=15**
Total Bandwidth (Mbps)	28.4	32.3	41.7	58.5

TABLE 4. Bitrate Distribution of Elephants Dream (ED-II) representations.

Mean Value	**L1**	**L2**	**L3**	**Total Size**
SVC Bitrate (kbps)	2430	978	1750	5167
MDC Bitrate (kbps)	2430	2430	2430	7290

bitrates of the SVC enhancement layers present the bitrate of the related layer solely. The bitrate of an SVC enhancement layer can also be represented cumulatively by considering layer dependencies. In that case, for example, the cumulative bitrate of the L2 would equal to the sum of bitrate values of L1 and L2.

As mentioned earlier, the controller assigns the paths only if these path assignments keep the estimated packet loss ratios of the layer packets under certain threshold values. We considered two different sets of packet loss thresholds for each layer, which are listed in Table 5. While SVC has different threshold value for each layer, MDC only has one value for each threshold level. Different thresholds for each SVC layer were defined due to the layer dependency and the observations of the QoE values affected by the path selection approach based on packet loss estimation. The tests were repeated 10 times for each setting. All test results presented in the figures and tables in the next section are averaged values.

TABLE 5. Packet loss threshold levels for different layers and codec types.

Threshold	$\mathbf{Thr}_1$	$\mathbf{Thr}_2$
SVC Layer 1	0.05	0.10
SVC Layer 2	0.10	0.20
SVC Layer 3	0.20	0.30
MDC Layers	0.10	0.20

B. EVALUATION RESULTS

For the evaluation of the performance, we measure the following QoE metrics: *(i)* average received bitrate, *(ii)* re-buffering duration, *(iii)* the number of the received video segments belonging to each layer, and *(iv)* the number of quality switches. These metrics are among the most important metrics showing the perceived quality on the client's side [64]. Three additional approaches are also implemented and the performance of each approach is evaluated with the same set of configuration in order to compare the performance of the proposed architecture. The comparison approaches are Throughput Based Adaptation (TBA) [15], Buffer Based Adaptation (BBA) [17] and PANDA [20]. In the TBA algorithm, clients measure network throughput while downloading the video segments and adapt quality according to the estimated network bandwidth. In the BBA algorithm, the clients select the quality based on buffer fullness level so that they request the highest possible quality when the buffer fullness is high and request the lowest quality when the buffer is almost empty. TBA and BBA approaches are selected as to observe the performance of the clients having different approaches to select the quality. We prefer to run TBA and BBA algorithms because these works are successful implementation of throughput and buffer based approaches. Also, they were selected as comparison approaches for several studies in the literature and they can be seen as benchmark algorithms [19], [21], [22].

Different than TBA and BBA, PANDA uses a special technique to estimate bandwidth on the basis of probes [20]. As being one of the approaches having a remarkable bandwidth estimation method, PANDA is a suitable approach to observe the performance of the proposed approach in this current study, since while the proposed approach utilizes network assistance to select quality, PANDA's quality selection utilizes a bandwidth estimation method based on HAS characteristics on the client's side. Therefore, it is possible to observe whether an improvement over an approach uses HAS specific client side bandwidth estimation method can be obtained if network assistance is provided. The clients using these rate adaptation algorithms measure the bandwidth and select the quality as the algorithms proposed in [15], [17] and PANDA [20]. However, since we stream SVC video for these clients, we added a mechanism in which when an enhancement layer segment is delayed for more than 4 seconds, it is discarded. We implement this mechanism to provide a fair comparison since the proposed algorithm has the same approach and discarding the delayed packets provides to prevent long re-buffering duration. However, if the base layer packets are delayed, each type of clients experiences re-buffering events.

The selection of the paths for transferring video layers are determined by the OLAPS algorithm for the proposed approach. Therefore, the SDN controller is HAS aware and it considers the characteristics specific to the layered video in the selection of the paths. For the other approaches, the controller forwards client's requested layers over the paths with maximum available bandwidth. The performance results given in this section will also show the performance improvement provided by the proposed path selection approach, where the routing is done by considering layer dependencies, compared to the one of the best path selection approach used for layered video in the literature.

The video bitrates received by the clients according to the network capacity are given in Figure 4. In this figure and other figures and tables represented in this section, OLAPS_thr$_1$ and OLAPS_thr$_2$ refers to OLAPS algorithm results obtained with threshold values thr_1 and thr_2, respectively. The clients using TBA and BBA algorithms have

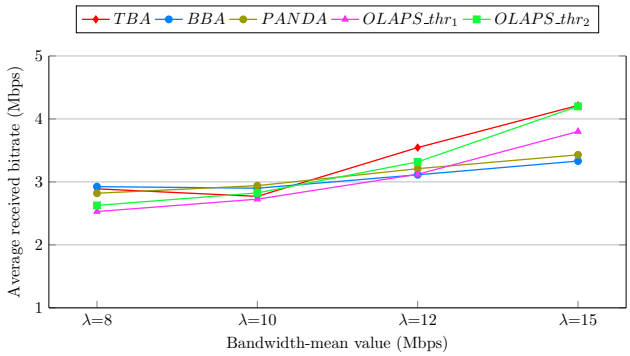

FIGURE 4. Average received bitrate with respect to network capacity. These values show the bitrate of the received video. However, a high video bitrate does not always mean a high level of QoE.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3114299, IEEE Access

higher bitrate than the clients in the OLAPS algorithm when network has limited bandwidth ($\lambda = 8$ Mbps). However, as it is going to be shown later, the clients using TBA and BBA approaches experience higher re-buffering duration and higher number of quality changes, which negatively affect the perceived quality. It is worth to mention that, seamless video streaming with minimum number of video stalls and minimum number of quality switches are so much preferable than the small quality degradation. While requesting video from the lowest bitrate results in poor video quality, requesting video from the higher layers increases the probability of re-buffering if there is not enough bandwidth. The main advantage of the OLAPS algorithm is directing clients to select a good point in the trade-off between bitrate and re-buffering risk. When comparing with the TBA, our proposed approach based on the OLAPS algorithm performs better for both threshold levels. PANDA clients also select layers providing a better trade-off point between bitrate and re-buffering than TBA and BBA approaches due to its bandwidth estimation approach. Among all approaches, OLAPS_thr$_1$ has the best performance considering this trade-off. This indicates that further improvement can be achieved with the assistance of the HAS aware network compared to even the case where the client perfectly observes the network conditions and SDN routes the video packets over the paths with maximum available bandwidths.

The graphs that show average throughput as a function of time for each bandwidth setting are given in Figure 5. The graphs in the figure show jointly the performance of the path selection approaches and the performance of bandwidth utilization obtained by each approach. It is observed that, especially if the bandwidth is limited, as in the tests where λ equals to 8 Mbps, the performance obtained with the proposed approach is better than the other approaches for both threshold values. The reason of that is, in the proposed approach, the available bandwidth is estimated with high precision due to the knowledge of the SDN controller about the HAS flows and characteristics. Since HAS clients request the segments intermittently, the traffic caused by the segments sent to the clients are not permanent. This phenomenon is known as ON and OFF periods [65]. The SDN controller without the SAND characteristics interprets the OFF periods as the increase in bandwidth as this is the case in other approaches. Hence, this leads to miscalculations in available bandwidth. On the other hand, although our approach uses the traffic measurements done by Floodlight like other studies, we also use the knowledge about HAS ON/OFF periods, the number of online flows, and the bitrate of the layers. Hence, our approach makes more successful estimation about available bandwidth, which results in better path assignments. In all cases, PANDA approach achieves higher throughput than TBA and BBA approaches. This shows that the bandwidth estimation method of PANDA algorithm is very successful since the clients were able to utilize bandwidth more than the others in the same position. When we examine Figure 5(d), we see that all approaches

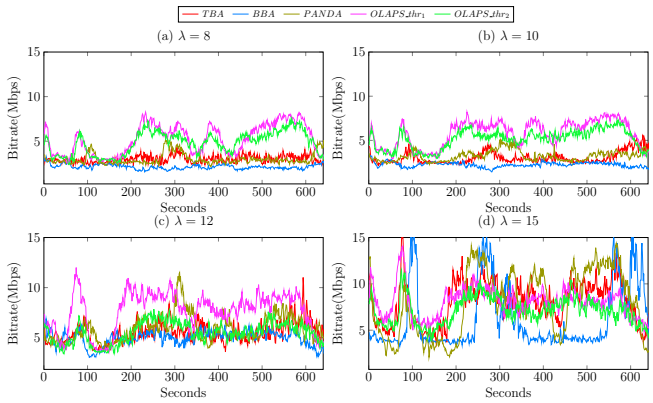

FIGURE 5. Average throughput (bitrate) as a function of time. These bitrate values show the throughput on the clients' side.

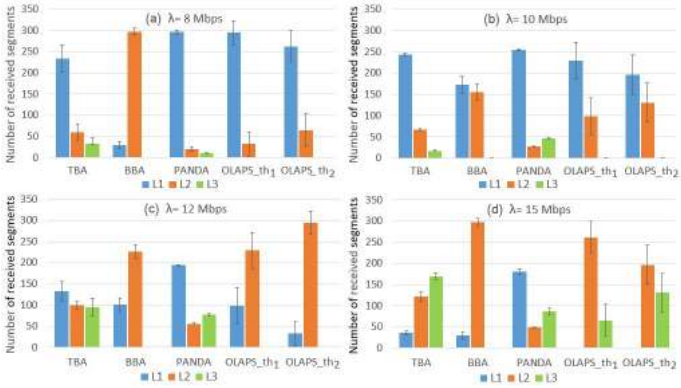

FIGURE 6. Average number of received layers per quality levels. This graph shows the quality distribution of 327 segments

.

obtain similar results. These observations lead us to conclude that, if the controller has knowledge about video streaming application characteristics, especially when the network is limited, the better routing decisions can be made.

Receiving more video segments from the higher layers provides to play the video with better quality. Figure 6 shows the received number of segments from each video layer during the simulations, where confidence interval is set to 95%. In the first scenario, it is observed that the BBA algorithm receives more segments from the first enhancement layer while the clients with other approaches receive more video segments from the base layer. At the first glance, this is seen as an indicator that the BBA algorithm outperforms other approaches while the network has less bandwidth. However, the clients experience high delay while downloading segments which, in turn, causes to unacceptable re-buffering duration. In the other scenarios where the link's capacities are increased, we observed that the OLAPS algorithm for both thresholds provides the clients to adapt quality so that the clients receive more segments from higher layers. This result shows that the clients with the OLAPS algorithm receive a minimum number of segments from the base layer among

all algorithms only when the network has enough capacity to transfer packets from higher layers. Another important observation is that, SDN controller assists client to select more appropriate codec and quality with the proposed approach. This observation can be made by especially examining the results when λ equals to 15 Mbps, where all approaches has similar network throughput (as it can be observed in Figure 5(d)). Since the bitrate amount arrived to the clients are similar, the results give more information about the performance of rate adaptation algorithm rather than performance improvement provided by the path selection approaches.

Startup delay is one of the parameters that affect the perceived quality and this time should be as minimum as possible. In Table 6, the startup delay values that were observed for each application with different network settings are provided. The results show that OLAPS_thr$_1$ managed to keep the startup delay under a certain level for each network setting. On the other hand, the higher threshold value used in OLAPS_thr$_2$ causes clients to request video from higher-quality segments and results in elongated startup delays.

When the network links are congested, the clients adapt to lower bitrates in order to avoid re-buffering. As shown in Table 7 and Table 8, when ($\lambda = 8$ Mbps), the proposed algorithm has lesser re-buffering duration than the other algorithms. But, especially the TBA and BBA algorithms experience unacceptable values for re-buffering, in both forms of duration and frequency. The main reason for that result is the client's greedy behaviour of these approach. As a result, the greedy behavior of a client in a short period of time may not ensure appropriate bitrate adaption. On the contrary, since the controller runs the OLAPS algorithm by using its SAND characteristics, it helps clients to request the highest number of video layers under the constraint of network capacity, avoiding sudden reactions based on bandwidth measurement changes caused by ON and OFF periods. Note that, when ($\lambda = 8$ Mbps), the re-buffering values observed in OLAPS is also high because the network capacity is so limited.

The dependency among SVC layers enforces the clients waiting for receiving all lower layers in order to play video with maximum quality, which can cause packets to be delayed especially when network is highly congested. Greedy behavior of the comparison algorithms trigger the clients to adapt to higher video quality which could result in congestion in competitive links. On the other hand, in the proposed approach, if the controller detects that the network has more bandwidth for only a very short time period due to the OFF periods of the clients, it restricts clients to request video from higher layers. Also, selecting optimal codec by considering SVC and MDC characteristics and network conditions, help to improve QoE. By considering those facts we can conduct that clients which are assisted by the OLAPS algorithm experience less re-buffering thanks to its approaches considering codec types and HAS characteristics.

The effects of *Algorithm 1* on the QoE can be clearer when examining the Table 7 and Table 8, for the λ values of 12 and 15 Mbps. As explained previously, *Algorithm 1* is used for calculating packet loss estimations and threshold values are determined by examining the correlation of the outputs of this algorithm and achieved QoE. The disadvantage of setting high values for packet loss thresholds is observed for these bandwidth capacities. By increasing network capacity, the re-buffering duration reduces significantly in all approaches except OLAPS_thr$_2$. Increasing packet loss threshold value gives more flexibility to the client to request video segments from the layer with the higher bitrate. Hence, the network becomes more congested and this leads clients to experience higher delay and more re-buffering events.

Table 9 shows the number of quality switches during streaming. Less number of quality switches means that the client experiences stable video quality. It is observed that the OLAPS algorithm provides stable video quality in all scenarios. The main reason is that, it prevents clients to switch higher layers when clients estimate that the network has more bandwidth in a short period of time due to OFF periods. In other words, the proposed algorithm bounds clients' greedy behaviors. In the table, $+(-)x$ shows the increase (decrease) in the video quality, where x is the difference of the video layer numbers between the successive requests of the clients. For example, +2 represents that the client receives the next segment from two quality levels higher than the lastly downloaded one. Clearly, the big jumps affect QoE more negatively. In all scenarios, it is clear that the OLAPS algorithm has the lowest value in terms of the number of increments and decrements. In addition, the BBA algorithm avoids a high number of quality switches since clients increase the video quality with increase in the buffer level. On the contrary, in the TBA algorithm, clients have greedy behavior to receive video from upper layers. Accordingly, when network is congested, clients receive base layer for a short period of time, and then may request video segments from upper layers when network capacity changes. Hence, TBA has higher number of quality switches when compared to other approaches. PANDA has better adaptation compared to the TBA and BBA algorithms.

In addition to measuring different QoE parameters, we also measure the overall QoE values. In [64], the QoE parameters that are used in the overall QoE calculation is well analyzed. We use the QoE formula given in [64] to calculate the overall QoE as follows:

$$QoE = \sum_{i=1}^{K} q(R_i) - \delta * \sum_{i=1}^{K-1} |q(R_{i+1}) - q(R_i)| - \mu * D_r - \mu * N_r - \beta * T_s. \quad (6)$$

The QoE formula calculates QoE value of the client which downloads K segments. The first term in the formula is the bitrate of the segments and the second term is the number of quality changes. D_r, N_r and T_s represents total re-buffering duration, total number of re-buffering events, and startup delay, respectively. We use the same values given in [21]

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3114299, IEEE Access

TABLE 6. Startup delay per client (seconds).

Mean Value	λ=8 Mbps	λ =10 Mbps	λ=12 Mbps	λ=15 Mbps
BBA	6.2	6.3	4.1	2.6
TBA	7.3	7.6	5.9	4.2
PANDA	6.2	6.4	4.6	4.6
OLAPS_thr$_1$	3.6	3.4	3.1	3.2
OLAPS_thr$_2$	4.3	4.2	3.6	3.1

TABLE 7. Total re-buffering duration per client (seconds).

Mean Value	λ=8 Mbps	λ =10 Mbps	λ=12 Mbps	λ=15 Mbps
BBA	157	120	66	12
TBA	143	104	49	19
PANDA	104	93	41	11
OLAPS_thr$_1$	51	35	24	9
OLAPS_thr$_2$	52	37	36	33

TABLE 8. Number of re-buffering per client.

Mean Value	λ=8 Mbps	λ=10 Mbps	λ=12 Mbps	λ=15 Mbps
BBA	5.0	3.9	2.6	0.8
TBA	4.9	2.9	2.8	0.9
PANDA	4.1	1.6	1.5	0.8
OLAPS_thr$_1$	1.6	1.4	1.1	0.6
OLAPS_thr$_2$	2.0	1.5	1.4	1.5

TABLE 9. Number of quality switches

Scenario 1(λ=8 Mbps)	TBA	BBA	PANDA	OLAPS_thr$_1$	OLAPS_thr$_2$
+1	21.4	5.5	4.7	0.1	0.3
+2	32.7	9.6	3.6	0	0
-1	52.0	0	6.8	0	0
-2	1.3	9.0	1.0	0	0
Scenario 1(λ=10 Mbps)					
+1	30.3	8.9	1.0	0.3	0.4
+2	17.0	0	8.1	0	0
-1	44.5	8.3	1.1	0	0
-2	2.3	0	7.8	0	0
Scenario 3(λ=12 Mbps)					
+1	25.5	5.9	1.9	0.7	0.9
+2	38	0	0.4	0	0
-1	60.5	5.2	2.0	0	0
-2	2.3	0	0	0	0
Scenario 4(λ=15 Mbps)					
+1	24.3	2.6	2.0	0.8	0.6
+2	50	0	0.4	0.4	0.8
-1	71	1.7	2.8	0	0
-2	2.1	0	0.16	0	0

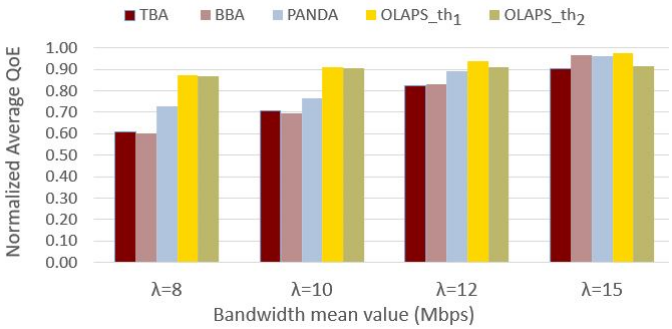

FIGURE 7. Normalized Average QoE values calculated for all approaches. The optimal QoE value is 1, depending on the given network conditions.

for the coefficients of the terms in this formula. The QoE values are calculated according to (6) for all approaches. The normalized values of the calculated QoE values are given in Figure 7. We theoretically calculate the optimal QoE value by considering the available bandwidth value that is shared by the clients and the optimal bitrate selection under this bandwidth value constraint. The optimal QoE value for given network conditions always equals to 1. The graph shows how close our proposed solution is to the optimal value.

V. CONCLUSION

In HAS applications, the clients have limited information about network conditions and other clients' behavior which also affects the client's experience. Therefore, network-based approaches, which direct clients and provide more information to the clients help them to adapt the quality optimally.

Layered video coding has some advantages due to provided storage optimization such as provided by SVC or robustness against lost layers such as provided by MDC. SVC

and MDC have different characteristics. In this paper, we proposed a video streaming system architecture where the SDN controller is aware of video codec types and HAS characteristics. By taking into account layer dependency constraints of both codecs, estimated packet loss ratios, and current network conditions, the controller selects the appropriate codec type for the clients, dynamically assigns streaming paths for each layer of the videos transferred to all clients. In addition to that, the controller recommends the client the optimal number of layers under the constraint of current network conditions. The clients utilize these recommendations within the rate adaptation algorithm and decide the video segments to be requested by also considering their own adaptation logic and buffer fullness level.

We presented HAS aware SDN controller assistance and SAND characteristics leveraging SDN technology can provide improvement in various QoE metrics, compared to other approaches where the clients are not directed by the controller. Furthermore, we showed that an HAS aware controller can estimate the available bandwidth with higher precision, compared to a regular SDN controller, although both controllers use the same information about the current bandwidth and traffic amount. Simulation results show that the proposed architecture provides an increase in received video quality up to 76% and up to 10% decrease in re-buffering duration when it is compared to another approaches where the paths with maximum available bandwidths are also assigned to the clients.

As the future work, we plan to implement an enhanced architecture of this proposal, where HAS aware web-caches cooperate for deciding which videos and which qualities should be kept. In such system, different layers of the video files can be distributed among caches within a particular proximity by considering their storage capabilities and the number of connected users to each of the caches.

ACKNOWLEDGMENTS

The authors would like to thank Dr. Cihat Cetinkaya from Mugla Sitki Kocman University for giving his valuable comments.

REFERENCES

[1] Cisco, "Global_2021_forecast_highlights." [Online]. Available: https://www.cisco.com, Accessed 2021

[2] MPEG-DASH, "Dynamic adaptive streaming over http," , https://mpeg.chiariglione.org/standards/mpeg-das, Accessed 2021. [Online]. Available: https://mpeg.chiariglione.org/standards/mpeg-dash

[3] S. Petrangeli, J. V. D. Hooft, T. Wauters, and F. D. Turck, "Quality of experience-centric management of adaptive video streaming services: Status and challenges," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 14, no. 2s, p. 31, 2018.

[4] C. Cetinkaya, Y. Ozveren, and M. Sayit, "An sdn-assisted system design for improving performance of svc-dash," *2015 Federated Conference on Computer Science and Information Systems (FedCSIS)*, pp. 819–826, Sept 2015.

[5] G. Sullivan, "Versatile video coding (vvc) – the new standard and its capabilities," *2020 Mile High Video (MHV)*, 2020.

[6] N. Bruno Astuto A, M. Marc, N. Xuan-Nam, O. Katia, and T. Thierry, "A survey of software-defined networking: Past, present, and future of programmable networks," *IEEE Communications surveys & tutorials*, vol. 16, no. 3, pp. 1617–1634, 2014.

[7] H. Kim and N. , Feamster, "Improving network management with software defined networking," *IEEE Communications Magazine*, vol. 51, no. 2, pp. 114–119, February 2013.

[8] S. Altamimi and S. Shirmohammadi, "Client-server cooperative and fair dash video streaming," *Proceedings of the 29th ACM Workshop on Network and Operating Systems Support for Digital Audio and Video*, pp. 1–6, 2019.

[9] M. Hemmati, A. Yassine, and S. Shirmohammadi, "A dec-pomdp model for congestion avoidance and fair allocation of network bandwidth in rate-adaptive video streaming," *2015 IEEE Symposium Series on Computational Intelligence*, pp. 1182–1189, 2015.

[10] A. Bentaleb, A. C. Begen, R. Zimmermann, and S. Harous, "Sdnhas: An sdn-enabled architecture to optimize qoe in http adaptive streaming," *IEEE Transactions on Multimedia*, vol. 19, no. 10, pp. 2136–2151, Oct 2017.

[11] D. Bhat, A. Rizk, M. Zink, and R. Steinmetz, "Sabr: Network-assisted content distribution for qoe-driven abr video streaming," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 14, no. 2s, Apr. 2018. [Online]. Available: https://doi.org/10.1145/3183516

[12] S. Bayhan, S. Maghsudi, and A. Zubow, "Edgedash: Exploiting network-assisted adaptive video streaming for edge caching," *arXiv preprint arXiv:2002.01553*, 2020.

[13] D. Kreutz, F. M. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, 2014.

[14] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "Openflow: Enabling innovation in campus networks," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, Mar. 2008. [Online]. Available: http://doi.acm.org/10.1145/1355734.1355746

[15] B. Rainer, S. Lederer, C. Müller, and C. Timmerer, "A seamless web integration of adaptive http streaming," in *2012 Proceedings of the 20th European Signal Processing Conference (EUSIPCO)*, 2012, pp. 1519–1523.

[16] X. Xie, X. Zhang, S. Kumar, and L. E. Li, "pistream: Physical layer informed adaptive video streaming over lte," *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, pp. 413–425, 2015.

[17] T. Huang, R. Johari, N. McKeown, M. Trunnell, and M. Watson, "A buffer-based approach to rate adaptation: Evidence from a large video streaming service," in *Proceedings of the 2014 ACM Conference on SIGCOMM*, ser. SIGCOMM '14. New York, NY, USA: ACM, 2014, pp. 187–198. [Online]. Available: http://doi.acm.org/10.1145/2619239.2626296

[18] S. K, U. R, and S. R. K, "Bola: Near-optimal bitrate adaptation for online videos," *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*, pp. 1–9, 2016.

[19] X. Wei, M. Zhou, S. Kwong, H. Yuan, S. Wang, G. Zhu, and J. Cao, "Reinforcement learning-based qoe-oriented dynamic adaptive streaming framework," *Information Sciences*, vol. 569, pp. 786–803, 2021.

[20] Z. Li, X. Zhu, J. Gahm, R. Pan, H. Hu, A. C. Begen, and D. Oran, "Probe and adapt: Rate adaptation for http video streaming at scale," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 4, pp. 719–733, 2014.

[21] W. Choi and J. Yoon, "Sate: Providing stable and agile adaptation in http-based video streaming," *IEEE Access*, vol. 7, pp. 26 830–26 841, 2019.

[22] A. Bentaleb, M. N. Akcay, M. Lim, A. C. Begen, and R. Zimmermann, "Catching the moment with lol+ in twitch-like low-latency live streaming platforms," *IEEE Transactions on Multimedia*, pp. 1–1, 2021.

[23] C. Gutterman, B. Fridman, T. Gilliland, Y. Hu, and G. Zussman, *Stallion: Video Adaptation Algorithm for Low-Latency Video Streaming*. New York, NY, USA: Association for Computing Machinery, 2020, p. 327–332.

[24] H. Schwarz, D. Marpe, and T. Wiegand, "Overview of the scalable video coding extension of the h.264/avc standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no. 9, pp. 1103–1120, Sept 2007.

[25] J. Jiang, V. Sekar, and H. Zhang, "Improving fairness, efficiency, and stability in http-based adaptive video streaming with festive," in *Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies*, ser. CoNEXT '12. New York, NY, USA: ACM, 2012, pp. 97–108. [Online]. Available: http://doi.acm.org/10.1145/2413176.2413189

[26] H. E. Egilmez, S. Civanlar, and A. M. Tekalpm, "An optimization framework for qos-enabled adaptive video streaming over openflow networks," *IEEE Transactions on Multimedia*, vol. 15, no. 3, pp. 710–715, April 2013.

[27] R. S. Kalan, M. Sayit, and S. Clayman, "Optimal cache placement and migration for improving the performance of virtualized sand," *2019 IEEE Conference on Network Softwarization (NetSoft)*, pp. 78–83, 2019.

[28] S. Kumar, N. Wang, C. Ge, and B. Evans, "Optimising layered video content delivery based on satellite and terrestrial integrated 5g networks," *2019 European Conference on Networks and Communications (EuCNC)*, pp. 161–166, 2019.

[29] D. Rui, "Dash based video caching in mec-assisted heterogeneous networks," *Multimedia tools and Applications*, 2020.

[30] J. Bruneau-Queyreix, M. Lacaud, D. Negru, J. M. Batalla, and E. Borcoci, "Ms-stream: A multiple-source adaptive streaming solution enhancing consumer's perceived quality," *2017 14th IEEE Annual Consumer Communications & Networking Conference (CCNC)*, pp. 427–434, 2017.

[31] S. Da Silva, J. Bruneau-Queyreix, M. Lacaud, and L. Negru, Dand Réveillère, "Muslin: A qoe-aware cdn resources provisioning and advertising system for cost-efficient multisource live streaming," *International Journal of Network Management*, vol. 30, no. 3, p. e2081, 2020, e2081 nem.2081.

[32] P. Jounsup and H. Jenq-Neng, "Dash and dash-vr video multicast systems," *ZTE Communications*, vol. 16, no. 3, pp. 15–22, 2020.

[33] A. A. Barakabitze, I.-H. Mkwawa, A. Hines, L. Sun, and E. Ifeachor, "Qoemultisdn: Management of multimedia services using mptcp/sr in softwarized and virtualized networks," *IEEE Access*, pp. 1–1, 2020.

[34] C. Cetinkaya, E. Karayer, M. Sayit, and C. Hellge, "Sdn for segment based flow routing of dash," *2014 IEEE Fourth International Conference on Consumer Electronics Berlin (ICCE-Berlin)*, pp. 74–77, Sept 2014.

[35] C. Cetinkaya, K. Herguner, C. Hellge, and M. Sayit, "Segment-aware dynamic routing for dash flows over software-defined networks," *International Journal of Network Management*, p. e2102, 2020.

[36] M. Sayit, C. Cetinkaya, H. U. Yildiz, and B. Tavli, "Dash-qos: A scalable network layer service differentiation architecture for dash over sdn," *Computer Networks*, vol. 154, pp. 12–25, 2019.

[37] P. Parastar, F. Pakdaman, and M. R. Hashemi, *FRAME-SDN: A Fair Resource Allocation for Multiplayer Edge-Based Cloud Gaming in SDN*. New York, NY, USA: Association for Computing Machinery, 2020, p. 21–27.

[38] A. Bentaleb, A. C. Begen, and R. Zimmermann, "Sdndash: Improving qoe of http adaptive streaming using software defined networking," in *Proceedings of the 2016 ACM on Multimedia Conference*, ser. MM '16. New York, NY, USA: ACM, 2016, pp. 1296–1305. [Online]. Available: http://doi.acm.org/10.1145/2964284.2964332

[39] S. Petrangeli, T. Wu, T. Wauters, R. Huysegems, T. Bostoen, and F. De Turck, "A machine learning-based framework for preventing video freezes in http adaptive streaming," *J. Netw. Comput. Appl.*, vol. 94, no. C, pp. 78–92, Sep. 2017. [Online]. Available: https://doi.org/10.1016/j.jnca.2017.07.009

[40] H. Nam, K. Kim, J. Y. Kim, and H. Schulzrinne, "Towards qoe-aware video streaming using sdn," *2014 IEEE Global Communications Conference*, pp. 1317–1322, 2014.

[41] S. Akhshabi, A. C. Begen, and C. Dovrolis, "An experimental evaluation of rate-adaptation algorithms in adaptive streaming over http," in *Proceedings of the Second Annual ACM Conference on Multimedia Systems*, ser. MMSys '11. New York, NY, USA: ACM, 2011, pp. 157–168. [Online]. Available: http://doi.acm.org/10.1145/1943552.1943574

[42] K. T. Bagci, K. E. Sahin, and A. M. Tekalp, "Compete or collaborate: Architectures for collaborative dash video over future networks," *IEEE Transactions on Multimedia*, vol. 19, no. 10, pp. 2152–2165, Oct 2017.

[43] I. Ozcelik and C. Ersoy, "Chunk duration–aware sdn-assisted dash," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 15, no. 3, Aug. 2019.

[44] E. Thomas, M. O. van Deventer, T. Stockhammer, A. C. Begen, and J. Famaey, "Enhancing MPEG DASH performance via server and network assistance," *SMPTE Motion Imaging Journal*, vol. 126, no. 1, pp. 22–27, 2017.

[45] R. S. Kalan, M. Sayit, and A. C. Begen, "Implementation of sand architecture using sdn," *2018 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, pp. 1–6, 2018.

[46] J. W. Kleinrouweler, S. Cabrero, and P. Cesar, "Delivering stable high-quality video: An sdn architecture with dash assisting network elements," in *Proceedings of the 7th International Conference on Multimedia Systems*, ser. MMSys '16. New York, NY, USA: ACM, 2016, pp. 4:1–4:10. [Online]. Available: http://doi.acm.org/10.1145/2910017.2910599

[47] D. Bhat, A. Rizk, M. Zink, and R. Steinmetz, "Network assisted content distribution for adaptive bitrate video streaming," in *Proceedings of the 8th ACM on Multimedia Systems Conference*, ser. MMSys'17. New York, NY, USA: ACM, 2017, pp. 62–75. [Online]. Available: http://doi.acm.org/10.1145/3083187.3083196

[48] S. Clayman, R. S. Kalan, and M. Sayit, "Virtualized cache placement in an sdn/nfv assisted sand architecture," *2018 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom)*, pp. 1–5, 2018.

[49] N. Xue, C. X, L. Gong, S. Li, D. Hu, and Z. Z, "Demonstration of openflow-controlled network orchestration for adaptive svc video manycast," *IEEE Transactions on Multimedia*, vol. 17, no. 9, pp. 1617–1629, Sept 2015.

[50] S. Laga, T. Van Cleemput, F. Van Raemdonck, F. Vanhoutte, N. Bouten, M. Claeys, and F. De Turck, "Optimizing scalable video delivery through openflow layer-based routing," *2014 IEEE Network Operations and Management Symposium (NOMS)*, pp. 1–4, May 2014.

[51] Y. Ergiz, A. M. Demirtas, and T. Girici, "Joint multipath flow and layer allocation for scalable video streaming," *Computer Networks*, vol. 191, p. 107995, 2021.

[52] A. Gangwal, M. Gupta, Gaur, V. L. M. S, and M. Conti, "Elba: Efficient layer based routing algorithm in sdn," *2016 25th International Conference on Computer Communication and Networks (ICCCN)*, pp. 1–7, Aug 2016.

[53] Y. Yue, Y. Ran, S. Chen, B. Yang, L. Sun, and J. Yang, "Joint routing and layer selecting for scalable video transmission in sdn," *2015 IEEE Globecom Workshops (GC Wkshps)*, pp. 1–6, Dec 2015.

[54] T. Uzakgider, C. Cetinkaya, and M. Sayit, "Learning-based approach for layered adaptive video streaming over sdn," *Comput. Netw.*, vol. 92, no. P2, pp. 357–368, Dec. 2015. [Online]. Available: https://doi.org/10.1016/j.comnet.2015.09.027

[55] F. Tashtarian, A. Erfanian, and A. Varasteh, "S2vc: An sdn-based framework for maximizing qoe in svc-based http adaptive streaming," *Computer Networks*, vol. 146, pp. 33–46, 2018.

[56] S. G. Ozcan and M. Sayit, "Improving the qoe of dash over sdn: A mcdm method with an intelligent approach," *2019 22nd Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN)*, pp. 100–105, 2019.

[57] K. A. Noghani and M. O. Sunay, "Streaming multicast video over software-defined networks," *2014 IEEE 11th International Conference on Mobile Ad Hoc and Sensor Systems*, pp. 551–556, Oct 2014.

[58] S. Shen, "Efficient svc multicast streaming for video conferencing with sdn control," *IEEE Transactions on Network and Service Management*, vol. 16, no. 2, pp. 403–416, 2019.

[59] R. S. Kalan, C. Cetinkaya, and M. Sayit, "Design of a layer-based video streaming system over software-defined networks," *2017 8th International Conference on the Network of the Future (NOF)*, pp. 8–13, Nov 2017.

[60] M. M. Baldi, T. G. Crainic, G. Perboli, and R. Tadei, "The generalized bin packing problem," *Transportation Research Part E: Logistics and Transportation Review*, vol. 48, no. 6, pp. 1205–1220, 2012.

[61] B. Lantz, B. Heller, and N. McKeown, "A network in a laptop: Rapid prototyping for software-defined networks," *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*, pp. 19:1–19:6, 2010.

[62] "The internet topology-zoo." [Online]. Available: http://www.topology-zoo.org, Accessed 2021

[63] "Dash SVC Dataset." [Online]. Available: http://concert.itec.aau.at/SVCDataset, Accessed 2021

[64] K. Miller, D. Bethanabhotla, G. Caire, and A. Wolisz, "A control-theoretic approach to adaptive video streaming in dense wireless networks," *IEEE Transactions on Multimedia*, vol. 17, no. 8, pp. 1309–1322, 2015.

[65] A. Bentaleb, B. Taani, A. C. Begen, C. Timmerer, and R. Zimmermann, "A survey on bitrate adaptation schemes for streaming media over http," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 562–585, 2018.

REZA SHOKRI KALAN is a CDN specialist in Digiturk beIN Media Group. He received his B.Sc degree in Computer Engineering from IAU-Shabestar-IRAN in 2003. For a decade he worked in commercial arena. He received M.Sc degree in Computer Engineering from Eastern Mediterranean University (EMU), Cyprus in 2013. and Ph.D. degree in Information Technology from Ege University- International Computer Institute in 2020. His research interests include computer networks, Software Defined Networking, wireless communication and multimedia system.

MUGE SAYIT is an Associate Professor at International Computer Institute in Ege University in Turkey. She received a M.Sc. degree in 2005 and PhD degree in 2011 in Information Technologies from the same institute and a degree in Mathematics from Ege University in 1999. Her research interests include Software Defined Networking, P2P networks,video streaming and video codecs.