DASH-QoS: A Scalable Network Layer Service Differentiation Architecture for DASH over SDN

Muge Sayit[a,*], Cihat Cetinkaya[b], Huseyin Ugur Yildiz[c], Bulent Tavli[d]

[a] *Int. Computer Institute, Ege University, 35100, Izmir, Turkey*
[b] *Dept. of Computer Engineering, Mugla Sitki Kocman University, 48000, Mugla, Turkey*
[c] *Dept. of Electrical and Electronics Engineering, TED University, 06420, Ankara, Turkey*
[d] *Dept. of Electrical and Electronics Engineering, TOBB University of Economics and Technology, 06560, Ankara, Turkey*

Abstract

As one of the key technologies residing in the future Internet concepts, Software Defined Networking (SDN) makes it possible to develop application-centric flow routing strategies. Thanks to the benefits offered by the SDN architecture, the performance of multimedia communication applications can be improved by developing application-centric flow routing strategies. Recently, MPEG-DASH standard which allows rate adaptation over HTTP has become popular. In this work, we present an architecture that provides increase in received quality of DASH clients running over an SDN domain. Within this architecture, we propose an optimization model which determines the streaming paths of the video packets of the clients. We also define different types of user classes and provide a guaranteed service and fairness among the users of the same class. Furthermore, we created a heuristic algorithm to facilitate the scalability of the solution. The simulation results show that, by considering the output of the optimization model and dynamically changing the routing paths in an ISP, the video quality offered to the DASH clients is significantly improved and a certain level of service guarantee is provided.

Keywords: HTTP streaming, service classes, OpenFlow, MIP, heuristics

*Corresponding author. Tel.: +90 (232) 311-3230
Email addresses: muge.sayit@ege.edu.tr (Muge Sayit), cihat.cetinkaya@mu.edu.tr (Cihat Cetinkaya), hugur.yildiz@tedu.edu.tr (Huseyin Ugur Yildiz), btavli@etu.edu.tr (Bulent Tavli)

1. Introduction

Video streaming applications dominate Internet bandwidth usage due to an increasing demand for such applications, as stated by Cisco. Recently, most of the video streaming applications such as Microsoft Smooth Streaming[1], Adobe's [2], and Apple's[3] applications send video packets over HTTP since HTTP traffic has reliable data transfer infrastructure, advantages of web cache usage, and firewalls [1]. Triggered by such applications, Dynamic Adaptive HTTP Streaming (DASH) standard is proposed by MPEG [1]. DASH standard enables the clients to change the bitrate of the video according to rate adaptation strategy of the client in order to maximize the received quality. For this purpose, video files stored in the DASH server have different representations with various bitrates; and each representation is divided into small segments to provide adaptation during a streaming session. In dynamic adaptive HTTP streaming systems, in addition to the encoded video data alternatives, the video server also stores a Media Presentation Description (MPD) document containing the address and timing information of these segments. DASH standard specification determines the format of MPD document and functionalities of the MPD parser and segments. After receiving the MPD file from the DASH server, DASH clients request a representation for each segment; hence they can change the bitrate of the received video in each segment period. The time of the request and the selection strategy of the representation are not defined in the standard. Thus, different rate adaptation algorithms for DASH clients deciding representation to be requested by taking estimated bandwidth [2, 3, 4], buffer status [5], or both [6], into account were proposed in the literature. Although remarkable approaches against limited bandwidth and rapid changing network conditions have been introduced; if the underlying bandwidth of the end-to-end path is not

[1] `https://msdn.microsoft.com/en-us/library/ff469518.aspx/[MS-STTR].pdf`

[2] `http://www.adobe.com/tr/products/hds-dynamic-streaming.html`

[3] `https://developer.apple.com/streaming/`

adequate and changes frequently, then decrease in QoE (Quality of Experience) parameters such as received bitrate and outages are inevitable.

Software Defined Networking (SDN) is a recently emerged paradigm introducing the separation of the data plane and the control plane of computer networks [7]. By putting the network intelligence to the control plane, this technology allows network operators to develop and implement application specific routing strategies. Routing information which are defined according to the requirements of an application are sent to the switches via an external device, the controller. Besides deciding and sending the routing information, the controller periodically communicates with the switches to monitor network capacity. It is possible to minimize the aforementioned negative effect of congested streaming paths by using the advantages that SDN technology presents. In order to overcome congested link problems causing decrease in received video quality, the streaming paths can be changed based on the commands taken from the controller. Streaming paths for conveying the packets of the applications can be determined by considering the requirements of the applications. Hence, new approaches to provide QoS can be implemented for the applications running over SDN.

In this study, we propose a service differentiated architecture developed for DASH clients residing within an SDN domain. We define different service classes for users and solve congestion-related problems by developing an approach which runs on network layer to provide an increase in the received video quality according to the users' service classes. Since our approach is solely based on network layer and it does not require any changes or modification on client's software, it can be utilized for any type of DASH software or HAS (HTTP Adaptive Streaming) based applications such as Microsoft Silverlight or Adobe HLS. In order to take traffic characteristics specific to the DASH applications into account, we consider the average bitrate of the video files stored in the server, current available bandwidth of the links between the server and the clients and competitive nature of TCP flows. Providing service differentiated classes requires providing the same service level among the users within the same class.

Furthermore, available network capacity should be shared fairly, considering service payments. Hence, a path assignment strategy providing fairness among and within the classes has to be developed for defining streaming paths according to user class for a given network topology. The contributions of this study can be listed as follows:

- We propose an architecture providing service differentiation to DASH clients. This is the first study that proposes an SDN based fair service differentiation architecture considering the characteristics of DASH applications, without requiring any modification on client's side.
- We design an optimization scheme based on Mixed Integer Programming (MIP) to determine the streaming paths between the server and the clients. The optimization model aims to maximize received quality and providing fairness among users belonging to the same service class.
- Furthermore, we design a decomposition based heuristic algorithm to provide a scalable solution which reduces the computational complexity of the MIP based optimization scheme.
- We give the performance of the proposed architecture by giving different weights to the optimization parameters under different network conditions. The network topologies are selected among the real topologies of large ISPs in the world.

The rest of the paper is organized as follows: In Section 2, we summarize the related work on video streaming over SDN. We elaborate on our architecture and the optimization framework in Section 3 and evaluate the proposed system in Section 4. In Section 5, conclusions are given. Details of the MIP model are given in the Appendix.

2. Related Work

SDN technology enables developing and applying different forwarding rules by separating the control and data planes of computer networks. In an SDN

network, a controller sends flow information to the switches. OpenFlow is the first communication protocol for transferring messages between the switches and the controller [8]. In addition to applying the rules sent from the controller, OpenFlow enabled switches to send the network topology information such as current capacity of the links and traffic flow amount to the controller. Since SDN paradigm offers a wide range of flexibility about designing application specific routing algorithms, there are various studies proposing application aware routing over SDN. Besides developing routing strategies for traffic generated by general applications [9], [10]; routing approaches specific to the packet flows of video streaming applications have been proposed recently. In [11], authors outline a QoE (Quality of Experience)-centric multimedia service framework that can be run over SDN and discuss dynamic path assignment by considering the service negotiations. An IPTV service model for providing a certain level of quality assurance is proposed in [12]. In this work, the flow statistics of RTP packets are obtained regularly and if congestion is detected, an alternative path is selected. The use of prioritized queuing is proposed for multimedia Cloud services QoS provisioning in [13]. Although service assurance is considered in [11], [12], [13] no specific path selection strategy is defined for increasing the video quality received by the clients in these studies. In [14], an optimization model for determining end-to-end streaming paths for audio, video and data flows in an SDN domain is proposed. The optimization model takes UDP related parameters such as packet loss, delay, and jitter into account as input parameters [14]. For streaming scalable coded video over SDN, packet flows belonging to different video layers can be routed according to the priority of the layers [15]. In [16], a routing algorithm based on an optimization model for SVC (Scalable Video Coding) video packets is proposed. This study is improved to be able to run over multiple SDN domains in [17]. A learning model is proposed to determine when to adapt video quality and when to re-route streaming paths of video scalability layers in [18]. In [19], the authors proposed a scalable video manycast framework for SDN domains, where multi-clients and multi-servers exist in an SDN domain. These studies propose to send the packets of base and

enhancement layers over different paths which are selected by considering UDP streaming parameters. If congestion is detected, video packets are re-routed in order to provide an increase in QoE.

The popularity of DASH applications lead researchers to stream DASH packets over various network technologies such as ICN (Information Centric Networking), P2P, and from different network entities such as battery-limited devices or sensors. There are some studies, which use different type of codecs such as SVC [20] or MVC (Multi-view Coding) [21], utilize P2P networks in order to increase the QoE achieved by DASH clients. The authors in [22] present a remarkable approach for transferring DASH packets from miniaturized devices which enables independent playing of segments. A comprehensive survey about DASH in ICN can be found in [23].

Dynamic adaptive HTTP streaming systems over SDN is proposed in several studies considering the characteristics of HTTP adaptive video streaming systems. In [24], authors propose a QoS model by introducing a traffic shaping scheme which defines packet forwarding rules according to the type of the protocol. By assigning priority to the HTTP flows, DASH clients achieve higher throughput when compared to the case that traffic shaping is not used [24]. In order to provide fairness among DASH clients, the authors proposed that the clients request bitrates according to the commands received from the controller in [25]. In [26], the video packets of the clients are placed into the prioritized queue by considering the buffer of the clients, and the size and the duration of the requested segments. In [27], an SDN architecture is proposed to maximize the QoE of the clients. In this architecture, SDN controller assists clients in their bitrate decision process by using the QoE parameters received by the clients while shared bandwidth is sliced by considering QoE fairness. Similar to [27], in [28] authors propose a bitrate adaptation assistance for DASH clients in an SDN utilizing DASH assisting network elements. In [29], a network assisted strategy is proposed in order to provide video quality fairness among DASH clients. In this study, the authors also proposed a bandwidth slicing technique to provide fairness [29]. Although QoS and/or fairness issues are considered in

these proposals, no path selection strategy for the video flows is proposed in [25, 24, 26, 27, 28, 29]. In [30], HTTP streaming by using SDN capabilities is proposed. In this study, the controller periodically obtains information such as re-buffering events and playing information from the clients. According to the status of the clients and obtained network information from the SDN, the controller decides to change the server or the streaming path in the SDN domain. QoS is not addressed in this work [30]. An optimization model providing service differentiation by determining the queue allocations among the paths between the server and the clients in an SDN network is proposed in [31], however, neither fairness among the users belonging to the same service classes nor routing by considering DASH characteristics such as representation bitrates was addressed in this study. In [32], a video service architecture, which provides QoE fairness among DASH clients by slicing link bandwidths and selecting routing paths, was proposed. Generally speaking, QoE fairness among the clients can be obtained by providing explicit information and restrictions about bitrates or bandwidth to the clients as proposed in [32]. These studies propose remarkable solutions by utilizing SDN for increasing the achieved quality, however, all of them were developed by considering client server communication and/or by requiring to use specific/modified DASH client software [30, 25, 27, 28, 32]. In our previous work, we propose to change streaming paths of DASH clients according to the bitrate of the current segment [33]. Server communicates with the controller each time it receives a request from a client, and controller determines the path for that client. The approach proposed in [33] also does not necessitate client software modification or client-controller communication, however, QoS and fairness are not considered in the previous work.

Our solution can be used with regular DASH or HAS clients because it does not require any modifications on the client rate adaptation algorithms or client-controller communication mechanisms. One may consider that adding plug-ins to the client software is not a very complex issue, however, since the specification of rate adaptation algorithms is not within the scope of the DASH standard, there are so many different types of rate adaptations used in the academic

proposals and commercial systems. On one hand, our approach is designed to be used as a universal solution which serves any kind of DASH client, which can also be used for any kind of HAS based systems. On the other hand, in the systems that depend on client-controller communication, scalability issues arise and this requires new solutions to be developed, which add further complexities to the system [27]. Furthermore, none of the aforementioned studies focus on providing QoS and guaranteed service levels through joint optimization of the received bitrate by the clients and fairness among the clients in the same service class.

3. The Proposed Service Differentiated SDN Network Architecture

The proposed architecture consists of two parts, the details of which are given in this section. The first part of the architecture is the modules defined as running in the controller. The seconds part is the optimization framework determining the streaming paths for DASH clients.

3.1. DASH-QoS Architecture

Fig. 1 shows the overall design of the proposed service differentiation architecture. In the architecture, the forwarding layer consists of DASH clients, DASH CDN server (CDN companies may deploy CDN server in access ISPs), hosts (other than DASH clients) creating cross-traffic and OpenFlow switches. In the control layer, an OpenFlow controller is responsible for the management of the network. The controller determines the streaming paths between the clients and server and sends the flow rules to the switches. The determination of the streaming paths is invoked by two events:

- when a new client joins the streaming system,
- when congestion occurs on the streaming paths between the clients and server.

In both cases, the controller runs the path assignment procedure given in the next section. After the paths are determined for the users, the controller updates

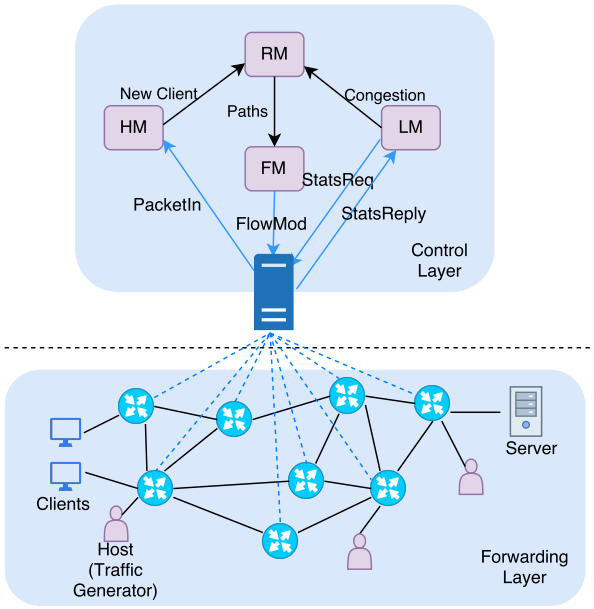


Figure 1: Design of the proposed architecture

the flow rules and sends to the switches. Hence, packets are dynamically re-routed according to the updated rules in the flow tables of the switches. Several modules which run in the controller are developed for different purposes:

- **HostManager (HM)**: It detects the new clients joining the system and maintains clients' information such as user class type, IP and MAC addresses. When a new client starts the DASH application, it establishes a TCP connection with the server. Upon receiving the first TCP message sent by the client, the first hop switch sends a *PacketIn* message to the controller since its flow-table does not have any rule for the client's request. HM module receives and extracts the *PacketIn* message and informs the RouteManager (RM) module for the assignment of the streaming path.

- **LinkManager (LM)**: It measures the available bandwidth of the links by sending *StatsReq* messages to the switches and receiving *StatsReply* periodically. If congestion occurs on a link, it informs the RouteManager module with the new values of available bandwidth of the links.

- **RouteManager (RM)**: It is invoked by the HM when a new DASH client joins the system or by the LM module when congestion occurs. It is responsible for the assignment of the streaming paths based on the optimization framework given in the next section. After assigning the streaming paths between the clients and the server, RM forwards the new streaming paths information to the FlowManager (FM).

- **FlowManager (FM)**: It keeps flow rules of the streaming paths and sends the new flow rules to the corresponding switches by *FlowMod* messages.

3.2. Mixed Integer Programming Model

In this subsection, we present the MIP framework. The algorithm determining the streaming paths between the clients and the server takes network topology and available bandwidth of the links as inputs. This algorithm is executed by the *RouteManager* module in controller and flow rules related to the selected paths are sent to the switches by *FlowManager* module.

We present a simple topology to, informally, introduce the optimization process via Fig. 2. In this topology, users' flows are from the Host (H) to the Server (S) via the four switches in the network topology. The capacity of each link (in terms of Kbps) is given in the figure and we assumed an undirected network. Link-$(H, 1)$ and link-$(4, S)$ have sufficiently have capacity that do not create any capacity bottlenecks. There are two types of users: standard users and premium users. Flows of the standard and premium users must be allocated, at least, 900 Kbps and 1200 Kbps bandwidth, individually, respectively. Furthermore, the lowest bandwidth allocated to any premium user must be higher than the highest bandwidth allocated to any standard user. The objective is to maximize the total bandwidth allocated to the flows of the users while minimizing sum of the the absolute values of the differences between the capacities allocated to the premium users. We have also a set of constraints in addition to the already stated ones. First, each flow must follow a single route (*i.e.*, flows cannot utilize multi-path routing). Second, all flows sharing a particular link

must be allocated equal amount of bandwidth whether any such flow utilize all of its allocated bandwidth or not. Third, the end-to-end bandwidth of a flow is determined by the minimum amount of the bandwidth allocated to it among all the links in its end-to-end path. Consider the case where we have three premium users (A, B, and C) and two standard users (D and E). The only feasible solution for the bandwidth and path assignment for this set of users, which does not violate any of the constraints, is as follows:

- user-A is assigned to path-$(H, 1, 4, S)$ with 4000 Kbps bandwidth,
- user-B and user-C are assigned to path-$(H, 1, 3, 4, S)$ with 2000 Kbps each,
- user-D is assigned to path-$(H, 1, 2, 3, 4, S)$ with 1000 Kbps bandwidth, and
- user-E is assigned to path-$(H, 1, 3, 2, 4, S)$ with 1000 Kbps bandwidth.

Note that the path and bandwidth assignments of premium users users are interchangeable among themselves (*e.g.*, user-B can be assigned to path-$(H, 1, 4, S)$ with 4000 Kbps bandwidth while user-A and user-C can be assigned to path-$(H, 1, 3, 4, S)$ with 2000 Kbps each) which also is true for the standard users (*e.g.*, user-E can be assigned to path-$(H, 1, 2, 3, 4, S)$ with 1000 Kbps bandwidth while user-D can be assigned to path-$(H, 1, 3, 2, 4, S)$ with 1000 Kbps bandwidth). User-A is the only user utilizing link-$(1, 4)$ which is the bottleneck in its path, therefore, User-A gets the whole bandwidth of link-$(1, 4)$ (*i.e.*, 4000 Kbps). User-B and user-C share link-$(1, 3)$ with user-E, therefore, both of them get one third of the capacity of link-$(1, 3)$ (*i.e.*, 6000/3=2000 Kbps). Likewise, user-B and user-C share link-(3.4) with user-D, thus, they get, again, one third of the capacity of link-$(3, 4)$ (*i.e.*, 6000/3=2000 Kbps). Since, allocated capacities at both of the inter-switch links in the paths of user-B and user-C (*i.e.*, link-$(1, 3)$ and link-(3.4)) are upper bounded by 2000 Kbps, both flows have capacities of 2000 Kbps. Flows of user-D and user-E share link-$(2, 3)$, therefore, the capacity on link-$(2, 3)$ is shared equally between user-D and user-E (*i.e.*, 2000/2=1000 Kbps) which is the bottleneck link in the paths of user-D and user-E. Even if the capacity of link-$(2, 3)$ were higher than 2000 Kbps (*e.g.*,

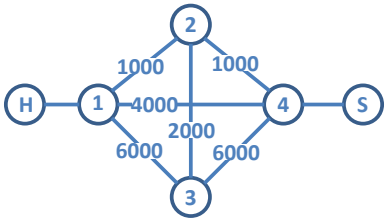


Figure 2: Network topology used for introducing the optimization process.

3000 Kbps), user-D and user-E would still get 1000 Kbps because of the capacity of link-$(1,2)$ (1000 Kbps), which is in the path of user-D, and the capacity of link-$(2,4)$ (1000 Kbps), which is in the path of user-E. Although user-D utilizes all its allocated capacities on link-$(1,2)$ and link-$(2,3)$, it can utilize only the half of its allocated capacity on link-$(3,4)$ due to the bottlenecks on link-$(1,2)$ and link-$(2,3)$.

After completing the basic introduction, we will, now, present the complete and formal description of our optimization problem. We define set $A = \{(i,j) : i \in N, j \in N - \{i\}\}$ to represent the links where set-N is the set of switches. Furthermore, set-$M = N - \{H, S\}$ represents the set of switches excluding the host (H) and the server (S) nodes. The set of all users, premium users, and standard users are denoted as U, U_P, and U_S, respectively. Capacities of links are denoted by C_{ij} and the set of link capacities are denoted as C^0. Furthermore, the capacity of the highest capacity link in the network is denoted as C_{mx}. The amount of data flow on link-(i,j) of user-k is denoted by f_{ij}^k. The variable bound on f_{ij}^k is given in Eq. (1).

$$0 \leq f_{ij}^k \leq C_{mx} \ \forall (i,j) \in A \ \forall k \in U \tag{1}$$

The objective function can be expressed as

$$\max \left[\gamma_1 \sum_{i \in U_P} s_i - \gamma_2 \sum_{i \in U_P} e_i + \gamma_3 \sum_{i \in U_S} s_i \right]. \tag{2}$$

Our objective function does not have a physical meaning, *per se*. However, this is a multi-objective optimization problem, hence, we are optimizing multiple

physical quantities. Since we are optimizing multiple objectives we combine them linearly through multiplication of weights (*i.e.*, γ_i's). The weights in the composite objective function determine the individual objectives' priorities.

The first term maximizes the data flow allocated to premium users (s_i is the source rate allocated to user-i). The second term has negative sign so it is for minimization of e_i's (the absolute value of the difference between the source rate of premium user-i and the average value of the source rates of all premium users). The third term is again for maximizing the flows allocated for standard users. The weights (*i.e.*, γ's) can be assigned for the emphasis.

For example, $\gamma_1 = 1.0$, $\gamma_2 = 0.1$, and $\gamma_3 = 0.01$ would lead to an optimization problem where the most important objective is to achieve maximum flow for premium users, the second objective is to minimize the differences between flows in the premium category, and the third objective is to maximize the flows in the standard category after achieving the first two objectives. In fact, solving a particular problem for a range of γ values will outline the operating region for the problem.

We will present an overview of the constraints of the MIP model in the rest of this Subsection. The details of the constraint equations are provided in the Appendix A.

Our first set of constraints facilitates the non-bifurcated flows within the network (*i.e.*, flow of any source is conveyed through a single path) given in Eqs. (A.1)–(A.9). The second set of constraints ensures that flows are conserved at the Host, the Server, and switches given Eqs. (A.10)–(A.14). All flows should utilize all the capacity they are allocated in their designated paths (*i.e.*, reduction of the utilized capacity arbitrarily is not allowed). To ensure such behavior, the third set of constraints puts a lower bound on the minimum bandwidth utilized by each flow which is the capacity allocated at the bottleneck link in the path of each end-to-end flow through the utilization of Eqs. (A.15)–(A.22). The fourth constraint is employed to allocate the same amount of capacity to all non-zero flows utilizing any particular link which is given in Eq. (A.23). Source rates assigned to premium users must be higher than the sources rates assigned

Table 1: Nomenclature for the optimization framework.

symbol	description	symbol	description
A	set of links	N	set of switches
M	set of switches excluding the host (H) and the server (S)	U	set of all users
U_P	set of premium users	U_S	set of standard users
f_{ij}^k	amount of data flow on link-(i,j) of user-k	C_{ij}	capacity of link-(i,j)
C^i	set of link capacities	C_{mx}	capacity of the highest capacity link
γ_i	weights of the objectives	s_i	source rate allocated to user-i
d_i	difference of user-k's data rate and the average data rate	e_i	absolute value of d_i
b_i	indicator variable utilized for transforming d_i to e_i	ϵ	minimum value for non-zero flows
a_{ij}^k	indicator variable for f_{ij}^k	x_{ij}	indicator variable for link-(i,j)
g_{ij}^k	slack flow on link-(i,j) for user-k	w_{ij}^k	indicator variable for g_{ij}^k
ζ_g	guaranteed bit rate for premium users	ζ_g	guaranteed bit rate for standard users
p_i^k	indicator variable for relay status of switch-i for user-k	r_i^k	degree of switch-i as a relay for user-k
P_l	set of paths obtained for standard users	P_m	set of paths obtained for premium users
$\overline{\mathbf{S}}$	minimum bandwidth end-to-end paths	$\overline{\mathbf{P}}$	maximum bandwidth end-to-end paths
$\hat{\mathbf{S}}$	final paths for standard users	$\hat{\mathbf{P}}$	final paths for premium users
η_{ij}^1	variable used to count the usage of link-(i,j)	u_l	virtual nodes

to standard users and sources rates of both standard and premium users must be higher than the predetermined rates for both categories. These set of rules (*i.e.*, the fifth set of constraints) are embodied within the MIP framework through the use of Eqs. (A.24)–(A.26). If not properly formulated, occurrence of phantom flows (*e.g.*, two switches sending the same amount of data to each other) cannot be avoided in our framework because such flows do not violate any of the previous constraints, hence, we made sure that any switch which is farther away from the Host (in terms of hop count) cannot send data to any other switch which is closer to the Host. Indeed our sixth set of constraints are utilized to completely avoid the phantom flows and given in Eqs. (A.27)–(A.36). Linearization of the second term of the objective function is achieved by our seventh set of constraints which are given by Eqs. (A.37)–(A.44). Nomenclature for the optimization framework is presented in Table 1.

Algorithm 1 The algorithm to determine the paths for each user.

Input: $A \leftarrow$ link set, $C^0 \leftarrow$ Initial link bandwidth set, $U_P \leftarrow$ Set of premium users, $U_S \leftarrow$ Set of standard users, $N \leftarrow$ Set of switches, $H \leftarrow$ Host node, $S \leftarrow$ Server node, $\gamma_1, \gamma_2, \gamma_3 \leftarrow$ weights, $\zeta_m \leftarrow$ minimum bandwidth, $\zeta_g \leftarrow$ guaranteed bandwidth.

Output: $s_i \ \forall i \in U_P \bigcup U_S \leftarrow$ Bandwidth of all users, $\hat{\mathbf{S}} \leftarrow$ Set of paths for standard users, $\hat{\mathbf{P}} \leftarrow$ Set of paths for premium users.

Define $C^1 = C^0$.

Determining paths for standard users :

Assume that there are no premium users ($U_P \in \varnothing$).

Assume that link-(i, j) is utilized by only one path at the beginning ($\eta_{ij}^1 = 1$) and $l = 1$ where l is the index of the path obtained.

while max-flow$(C^1) \geq \zeta_m$ **do**

  Determine a path-(P_l) with minimum bandwidth by using the MIP framework with $\gamma_1 = \gamma_2 = 0$, $\gamma_3 = -1$ including the following constraint: $f_{ij}^k + g_{ij}^k \leq \frac{C_{ij}^0}{\eta_{ij}^1} \ \forall (i, j) \in A, \forall k \in U_S$.

  Mark the links in P_l and increment η_{ij}^1 by 1.

  Update capacity values in C^1 by subtracting the flows in the links of P_l.

  Include P_l to $\overline{\mathbf{S}}$, which contains all possible paths determined so far.

  $l++$

end while

Obtaining the bandwidths for standard users by using the designated paths :

Create a new network, $H_1 = (N_1, A_1, C^S)$ such that $N_1 = \{H, u_1, \ldots u_l, S\}$, $A_1 = \{(i, j) : \forall i \in N_1, \forall j \in N_1\}$, and C^S which is the capacity matrix for all paths in $\overline{\mathbf{S}}$. In this network $u_1, \ldots u_l$ refers the virtual nodes that represent paths in $\overline{\mathbf{S}}$.

Solve the original MIP model by minimizing the maximum bandwidth of all standard users using H_1 and obtain $\hat{\mathbf{S}}$. Create the new link set, C^2, from C^0 by removing the residue bandwidth used in the standard users' path assignment steps.

Determining paths for premium users :

Assume that there are no standard users ($U_S \in \varnothing$).

Assume that $\eta_{ij}^2 = \eta_{ij}^1$ and $m = 1$ (path index).

while max-flow$(C^2) \geq \zeta_g$ **do**

  Determine a path-(P_m) with maximum bandwidth by using the MIP framework with with $\gamma_1 = 1$, $\gamma_2 = \gamma_3 = 0$ including the following constraint: $f_{ij}^k + g_{ij}^k \leq \frac{C_{ij}^0}{\eta_{ij}^2} \ \forall (i, j) \in A, \forall k \in U_P$.

  Mark the links in P_m and increment η_{ij}^2 by 1.

  Update capacity values in C^2 by using the bandwidth value of the P_m.

  Include P_m to $\overline{\mathbf{P}}$, which contains all possible paths determined.

  $m++$

end while

Obtaining the bandwidths for premium users by using the designated paths :

Create a new network, $H_2 = (N_2, A_2, C^P)$ such that $N_2 = \{H, v_1, \ldots v_m, S\}$, $A_2 = \{(i, j) : \forall i \in N_2, \forall j \in N_2\}$, and C^P which is the capacity matrix for all paths in $\overline{\mathbf{P}}$. In this network $v_1, \ldots v_m$ refers the virtual nodes that represent paths in $\overline{\mathbf{P}}$.

Solve the original MIP problem with $\gamma_1 = 1$, $\gamma_2 = 1$ or 0, and $\gamma_3 = 0$ by maximizing the total bandwidth on network H_2 and obtain $\hat{\mathbf{P}}$.

3.3. Heuristic Algorithm

Constraint optimization is the process of maximizing or minimizing an objective function while satisfying a set of constraints expressed as mathematical relationships. If the optimization variables are continuous (not constrained to be integers) and all of the mathematical relations expressing the objective and constraint functions are linear then such an optimization model is known as Linear Programming (LP) model which can be solved to optimality, efficiently, by polynomial time algorithms. If the linearity of the optimization model is preserved but some of the optimization problems are confined to the set of integers then such a model is known as an MIP model. MIP models can be solved by utilizing algorithms such as branch-and-bound and branch-and-cut. However, MIP models, in general, cannot be solved to optimality, especially, for large problem instances due to the NP-completness of the solution algorithms. To cope with the challenge, heuristic algorithms are utilized which, typically, terminate much faster, yet, they cannot guarantee the optimality of the solution obtained.

Algorithmic complexity of our MIP model is in the NP class. Consider the topology where the host and the server are connected directly by a certain number of links with different capacities. The problem of optimal assignment of standard and premium users to these links is an instance of the NP-complete problem of generalized bin packing [34]. Therefore, by the additional complexity of determining end-to-end paths with side constraints makes our problem an NP-complete problem. Hence, the computational complexity of the MIP problem is high, therefore, it does not scale well as the number of switches and users grow. In this subsection we present a heuristic algorithm which significantly enhances the scalability of our architecture. We present an analysis of the MIP model and the heuristic algorithm in Subsection 4.2 to compare the computational complexities and performances of both approaches.

The most important factor increasing the complexity of the MIP model is that all flows are considered jointly resulting in a high number of variables and constraints. In the heuristic algorithm we first determine feasible end-to-end paths, however, only one path a time is determined by using the MIP model

with some auxiliary constraints. Later, we assign users' flows on these paths by using the original MIP model, however, without the complexity of determining paths (*i.e.*, paths are determined in the previous step of the algorithm) which reduce the number of variables and constraints drastically. Nevertheless, due to the utilization of the original MIP problem in the heuristic algorithm, the heuristic algorithm is also NP-complete. Note that we still use the MIP model in the heuristic algorithm (in fact multiple times sequentially), however, the problem instances of the modified MIP model utilize very low number of variables and constraints in comparison to the original MIP model. In fact, the number of variables and the number of constraints required in the MIP model are $O(|N|^2|U|)$ and $O(|N|^2|U|^2)$, respectively. However, in the heuristic algorithm both the number of variables, $O(|N|^2)$, and the number of constraints, $O(|N|^2)$, are significantly lower for each problem instance. Moreover, the heuristic algorithm utilizes such MIP problem instances for, at most, $|U|$ times, successively. Therefore, due to the drastic reduction of the number of variables and the number of constraints, the heuristic algorithm terminates much more rapidly, as demonstrated in Subsection 4.2.

The pseudo-code of the heuristic is outlined in Algorithm 1 where we divide the problem in two main parts. In the first part we assign paths and bandwidths to the standard users. In the second part we assign paths and bandwidths to the premium users.

In the first part (lines 1–12), we determine minimum bandwidth end-to-end paths (denoted by $\overline{\mathbf{S}}$) each having a minimum bandwidth of ζ_m (lines 2–10) and assign all *standard users* to these paths (line 11–12). For this purpose, we assume that there are no *premium users* (line 2). We introduce the variables η_{ij}^1 which is used to count the usage of link-(i,j) by the obtained paths, P_l (line 3). In line 5, the objective function of the MIP framework is modified with $\gamma_1 = \gamma_2 = 0$, and $\gamma_3 = -1$ such that the total bandwidth of the standard users are minimized (*i.e.*, $\max\left[-\sum_{i \in U_S} s_i\right] = \min\left[\sum_{i \in U_S} s_i\right]$). Note that we also include the additional constraint in line 5 to make sure that if a link is utilized more than once, the capacity associated to that link is fairly shared among the standard

users (*e.g.*, if the link-$(3,2)$ is used 5 times while having the initial capacity of $C^0_{3,2} = 5000$ kbps, than each standard user can at most has a bandwidth of 1000 kbps, $s_i \leq 1000$ kbps). After solving the MIP model sequentially we obtain minimum bandwidth paths (P_l), mark the links in P_l (line 6) and update the capacity matrix, C^1, by subtracting the flows in the links of P_l (line 7). This process continues until the maximum flow between the Host and the Server in C^1 is less than the minimum bandwidth ζ_m (*i.e.*, we cannot find any more paths with at least end-to-end ζ_m capacity). After we obtain the set $\overline{\mathbf{S}}$ which contains all possible P_l sets, we assign all *standard users* to these paths. We create a virtual network, H_1, consisting of the Host (H) and Server (S) nodes as well as virtual nodes ($u_l,\ \forall l \in \overline{\mathbf{S}}$) that represent paths in $\overline{\mathbf{S}}$. In fact, each virtual node is connected to only the Host and the Server (*i.e.*, virtual nodes do not have links to each other). Link capacities of each virtual node is equal to the capacity of one path in $\overline{\mathbf{S}}$ as stated in line 11. For example, if we have three paths in $\overline{\mathbf{S}}$ with end-to-end capacities of x, y, and z then there are three virtual nodes in H_1 with link capacities of x, y, and z (*e.g.*, virtual node one has only one incoming and only one outgoing link of capacity x). When the new virtual network is constructed we solve the original MIP framework with the objective of minimizing the maximum bandwidth of standard users (line 12). By the completion of the first part of the algorithm, the paths for standard users and corresponding bandwidth values ($\hat{\mathbf{S}}$). We update the capacity values of C^0 by removing the bandwidth used in the standard users' path assignment steps and store this residue capacity as C^2.

In the second part (lines 13–23), we determine maximum bandwidth end-to-end paths each having a minimum bandwidth of ζ_g (lines 13–21) and assign all *premium users* to these paths (lines 22–23). In this case we assume that there are no standard users (line 13). The main idea is similar to the case that we obtain the minimum bandwidth paths (line 2–10). However, the only difference is the weights (γ) of the objective function. Since we aim to maximize the bandwidths of the premium users, the objective function of the MIP framework should be modified as: $\max\left[\sum_{i \in U_P} s_i\right]$. This can be achieved by choosing $\gamma_1 =$

1, $\gamma_2 = \gamma_3 = 0$ (line 16). After solving the modified MIP problem, we get the set of maximum bandwidth paths $\overline{\mathbf{P}}$ that contains all path information (P_m) obtained (line 19). By using the path information and path capacities we create another virtual network, H_2, in a similar way that we do in the first part (line 22). However, in the second part, we choose $\gamma_1 = 1$ to maximize the capacity allocated to the premium users (line 23). Minimizing the differences between the assigned capacities among the premium users is also facilitated by choosing $\gamma_2 = 1$ or $\gamma_2 = 0$. Since the capacities of standard users are already determined in the first part of the algorithm, we do not assign capacities to the standard users in the second part (*i.e.*, $\gamma_3 = 0$). Upon the completion of the second part of the algorithm paths for the premium users ($\hat{\mathbf{P}}$) are allocated.

The reduction in the total rate assigned to the premium users by the heuristic algorithm is due to the suboptimal assignment of rates to the standard users. In the first part of the heuristic algorithm, bandwidth assignment to the standard users are done in a way that minimize the total bandwidth assigned to the standard users while satisfying the minimum rate constraint of standard users. The reason for such an objective is to maximize the available bandwidth for the premium users which will be handled in the second part of the heuristic algorithm. However, while trying to minimize the bandwidth assigned to the standard users, it is possible that the first part of the heuristic, actually, can degrade the optimal bandwidth assignment for the premium users which results in suboptimal path and bandwidth assignments for the premium users in the second part of the algorithm. We will present a theorem on the performance bound and provide a constructive proof.

Theorem 1: Let S_{opt} and S_{heu} be the feasible and non-trivial solutions obtained by the MIP model and the heuristic algorithm for an arbitrary network topology, respectively. Furthermore, let the total source rates determined by S_{opt} and S_{heu} for premium users be B_{opt} and B_{heu}, respectively. Then, the optimality gap of the heuristic algorithm, in terms of the total bandwidth assigned

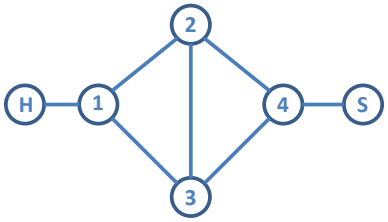


Figure 3: Network topology used for the proof of optimality gap.

to the premium users is given as

$$\frac{B_{heu}}{B_{opt}} \geq \frac{1}{|U_S|+1}. \tag{3}$$

Proof 1: The equal bandwidth allocation constraint given in Eq. A.23 dictates that if there are x users' flows are passing through the same link then each flow can receive, at most, $\frac{1}{x}$ of the link's capacity. As such, instead of receiving the full bandwidth on a link a premium user can end up getting only $\frac{1}{x}$ of the link's capacity. Furthermore, if such a link is the bottleneck link in the overall path then the capacity allocation can be as low as $\frac{1}{x}$ of the capacity that can be obtained by the optimal assignment. Hence, providing a single example of such a case will be sufficient to prove the worst-case performance of the heuristic algorithm.

Consider the network topology in Fig. 3. Assume there are $|U_S|$ standard users and 1 premium user (*i.e.*, $|U_P| = 1$). Furthermore, the link capacities are given as: $C_{12} = C_{13} = C_{24} = C_{34} = (|U_S| + 1) \times \zeta_g$ and $C_{23} = |U_S| \times \zeta_m$ where ζ_g and ζ_m are the guaranteed bit rate (for premium users) and the minimum bit rate (for standard users), respectively ($\zeta_g > \zeta_m$). One of the optimal solutions found by the MIP model assigns the standard users to the path consisting of link-$(1,2)$ and link-$(2,4)$ whereas the premium users is assigned to the path consisting of link-$(1,3)$ and link-$(3,4)$. Note that there are multiple optimal flow assignments in this scenario which will result in the same optimization objective value. Nevertheless, the premium user gets a rate of $(|U_S| + 1) \times \zeta_g$ and each standard user gets $\frac{|U_S|+1}{|U_S|} \times \zeta_g$ in the optimal solution. On the other hand, when we utilize the heuristic algorithm, to minimize the bandwidth assigned to the

standard users, all standard users are assigned to the path consisting of link-$(1,2)$, link-$(2,3)$ and link-$(3,4)$ due to the fact that $C_{23} < C_{12}$ (*i.e.*, the minimum feasible bandwidth is obtained when all standard users utilize link-$(2,3)$ which is ζ_m for each standard user). However, by making such an assignment there are only two paths left for the premium user which are consisting of either link-$(1,3)$ and link-$(3,4)$ or link-$(1,2)$ and link-$(2,4)$ both gives the same amount of bandwidth to the premium user which is $\frac{|U_S|+1}{|U_S|+1} \times \zeta_g = \zeta_g$. Therefore, the ratio of the bandwidth assigned to the premium user in the heuristic algorithm and the exact solution is $\frac{1}{|U_S|+1}$ which concludes the proof. ■

Note that the worst case optimality gap is obtained for a specially constructed topology. In fact, we have never encountered a scenario that the heuristic algorithm results in such a drastic performance deterioration. Indeed, the computational investigation of the optimality gap presented in Subsection 4.2 reveals that the average optimality gap is much lower.

4. Performance Evaluations

4.1. Simulation Setup

We use Mininet[4] emulator in order to evaluate the performance of the proposed architecture. Mininet is an efficient platform for implementing and testing systems with SDN and OpenFlow. General Algebraic Modeling System (GAMS)[5] with CPLEX solver is employed for the solutions of the optimization problems. There are 10, 20 and 30 clients connected to the DASH server, respectively. Half of the clients are subscribed as premium user whereas the remaining clients are standard users. Clients join the system with the 10 seconds of intervals. As controller software, Floodlight[6] is used .

The server sends the Big Buck Bunny video file @720p [35] to the clients. The bitrates of the representations used in the experiments are 900 Kbps, 1200

[4] `http://mininet.org/`

[5] `http://www.gams.com/`

[6] `http://www.projectfloodlight.org/floodlight/`

Kbps, 1500 Kbps and 1800 Kbps. All clients use DASH-JS software [2] to play the video. For both type of users, the clients adapt the quality based on the throughput measurements. The guaranteed bandwidth value offered to the premium user equals to 1200 Kbps, and the guaranteed bandwidth provided to standard users equals to 900 Kbps. Only soft guarantees can be given to the users since detecting and changing a congested streaming path may take several milliseconds and this causes a decrease in available bandwidth during a short period. However, the received video quality by the clients are not affected significantly by this problem as observed from the simulation results given in the next section. The simulations last for 300 seconds. The controller communicates with the switches in every 2 seconds.

We use two different type of real-world topologies, known as Dfn and Compuserve[7]. There are 11 switches and 7 paths between the server and the clients in Compuserve topology. Dfn consists of 57 switches and 1179 paths between the server and the clients. We create cross traffic on the links in both topologies during simulations. The available bandwidth value of each link in the SDN domain is independently distributed according to uniform distribution with randomly selected mean values between 1000 Kbps and 10000 Kbps. The available bandwidth of the links periodically changes due to cross traffic.

4.2. Scalability of MIP model and Heuristic Algorithm

In Table 2, we present a comparison of average data rates (in kbps) for premium and standard users obtained with the MIP model and heuristic algorithm as a function of number of users for the Compuserve topology with $\gamma_1 = 1,\ \gamma_2 = \gamma_3 = 0$. We vary the number of total users from 8 to 16 with an increment of 2 users. The MIP model and the heuristic algorithm are solved by using CPLEX in a personal computer which has 16 GB of RAM with Intel Core i7 3.2 Ghz processor. IBM CPLEX is an optimization software package. It has a modeling layer providing interfaces to a plurality of popular programming

[7] `http://www.topology-zoo.org/`

languages and software packages. It can solve linear or quadratic optimization problems with continuous or integer variables using various methods and algorithms such as the simplex method, the barrier interior point method, second-order cone programming. It takes prohibitively large amounts of time when we attempt to solve the exact optimization problem (*i.e.*, MIP) when the number of users exceeds 16, therefore, we cannot present results for higher number of users. Indeed, we cannot obtain results for the DFN topology by using the MIP model due to the high number of switches and users which increases the number of variables and constraints drastically.

The total bandwidth assigned by the heuristic algorithm is always lower than the total bandwidth assigned by the MIP model for all problem instances from 8 users to 16 users cases. For some problem instances, the total bandwidth assignment of the MIP model can be very close to that of the MIP model (*e.g.*, the total bandwidth assignment of the MIP model is only 0.32% less than the heuristic algorithm for 14 users), however, for some other problem instances the total bandwidth assignment of the MIP model can be non-negligibly lower than that of the MIP model (*e.g.*, the total bandwidth assignment of the MIP model is 8.32% less than the heuristic algorithm for 16 users).

Since the heuristic algorithm cannot assign the bandwidths of the standard users, optimally, in some scenarios heuristic algorithm's bandwidth allocation to standard users are higher than the MIP model's assignment of bandwidth to standard users which leads to the lower bandwidths for the premium users with the heuristic algorithm when compared to the MIP model. For example, the aggregate bandwidth assignments of the MIP model for premium users are 6.88%, 17.68%, 0.76%, 16.24%, and 4.67% less than the heuristic algorithm for 8, 10, 12, 14, and 16 users, respectively.

The reason for the heuristic algorithm to allocate higher bandwidth for the standard users in some instances when compared to the MIP solution is the decomposition we utilized in the heuristic algorithm. In fact, it is the decomposition that makes the heuristic algorithm faster (as analyzed in Subsection 3.3). However, the downside of the decomposition is the suboptimal assignment of

the bandwidths of the standard users which in turn leads to the suboptimal assignment of the bandwidths of the premium users . Nevertheless, suboptimal resource allocation is an expected result of a decomposition based heuristic algorithm.

Table 2: Average data rate (in kbps) for premium and standard users obtained by the MIP model and heuristic algorithm as a function of number of users for Compuserve topology

		Avg. Data Rate (kbps)				
Model	**Users**	**8**	**10**	**12**	**14**	**16**
MIP	**Premium**	14820	17580	18900	21250	29231
	Standard	4405	5902	6486	9650	9920
	Total	19425	23482	25386	30900	39151
Heuristic	**Premium**	13800	14472	18756	17800	27864
	Standard	4440	7056	5976	13000	8064
	Total	18420	21528	24732	30800	35928

In Table 3, we provide average solution times (in seconds) of the MIP model and the heuristic algorithm with respect to the number of users. Solution times for the MIP model increases with much higher pace than the solution times of the heuristic algorithm. In fact, with 16 users solution time for the MIP model is two orders or magnitude higher than the solution time of the heuristic algorithm. Nevertheless, the heuristic algorithm results in a huge gain in solution time by sacrificing a limited performance loss.

4.3. Simulation Results

As mentioned in previous section, the proposed architecture aims to provide received video quality maximization, fairness and to provide different classes of services. We compare the performance of proposed architecture with two different approaches. One of them is best-effort (shortest path) service model. In the best-effort service model, k-shortest paths are selected among the paths

Table 3: Average solution times (in seconds) for the MIP and heuristic algorithm wrt. number of users for Compuserve topology

	Avg. Solution time (s)				
Model / Users	**8**	**10**	**12**	**14**	**16**
MIP	13	101	191	538	601
Heuristic	2.09	2.50	2.71	3.22	3.30

Table 4: Quality changes in Dfn topology (heuristic algorithm)

			# of clients																	
Approach	**Result**	**User Class**	**10**						**20**						**30**					
			+1	**+2**	**+3**	**-1**	**-2**	**-3**	**+1**	**+2**	**+3**	**-1**	**-2**	**-3**	**+1**	**+2**	**+3**	**-1**	**-2**	**-3**
Proposed-max	**avg**	**pr**	9	2	0	12	1	0	10	3	1	18	1	0	11	2	1	17	1	0
		std	4	0	0	6	1	0	10	4	0	19	1	0	8	2	1	14	2	0
	stdev	**pr**	2	1	0	3	1	0	2	1	14	1	0	0	1	1	6	1	0	0
		std	2	0	0	3	1	0	1	14	9	0	0	0	1	1	6	1	0	0
Proposed-fair	**avg**	**pr**	9	2	0	11	2	0	12	3	0	18	2	0	10	2	1	16	1	0
		std	4	0	0	6	1	0	12	4	1	21	1	0	8	1	1	13	2	0
	stdev	**pr**	3	1	1	2	1	0	2	1	9	1	0	0	1	1	8	1	0	0
		std	2	0	0	3	1	0	1	9	5	0	0	0	1	1	6	1	0	0
CSP-QoS	**avg**	**pr**	15	3	1	22	1	0	14	3	1	24	1	0	12	2	1	18	1	0
		std	10	1	0	14	1	0	10	2	0	14	1	0	10	3	1	18	2	0
	stdev	**pr**	4	1	1	4	1	0	1	1	7	1	0	0	1	1	7	1	0	0
		std	3	1	0	3	1	0	1	7	4	0	0	0	2	1	12	1	0	0
k-shortest	**avg**	**pr**	4	1	0	6	1	0	5	1	0	9	1	0	6	1	0	9	1	0
		std	5	1	0	7	1	0	2	1	0	3	1	0	4	1	0	6	1	0
	stdev	**pr**	3	1	0	3	0	0	2	1	15	1	0	0	1	0	13	1	0	0
		std	4	1	0	4	1	0	1	15	8	0	0	0	1	0	9	1	0	0

between the server and the clients. Premium users are assigned to shortest paths while standard users are assigned to the remaining paths in the k-shortest path set. Other comparison approach is based on the approach given in [16], where the path selection is determined as a CSP (constrained shortest path) problem taking the bandwidth and delay variation into account. In order to obtain comparable outputs with our approach, the streaming paths are determined for premium users first. After that, available bandwidths of the paths are re-calculated and streaming paths are determined for standard users. We refer this approach as CSP-QoS in the graphs and tables given in this section.

The objective function given in Eq. 2 is a weighted (*i.e.*, γ_i's) sum of three

Table 5: Quality changes in Compuserve topology (heuristic algorithm)

			# of clients																	
Approach	Result	User Class	10						20						30					
			+1	+2	+3	-1	-2	-3	+1	+2	+3	-1	-2	-3	+1	+2	+3	-1	-2	-3
Proposed-max	avg	pr	10	2	0	15	0	0	9	2	1	16	1	0	13	2	0	19	0	0
		std	9	2	1	14	2	0	7	2	2	15	1	0	9	3	1	18	1	0
	stdev	pr	3	1	0	4	0	0	2	1	12	1	0	0	2	1	6	0	0	0
		std	4	1	1	4	1	0	1	12	7	0	0	0	2	1	5	1	0	0
Proposed-fair	avg	pr	13	4	1	24	0	0	12	3	1	21	0	0	14	3	1	24	0	0
		std	11	2	1	20	0	0	8	2	1	12	2	0	10	1	1	15	1	0
	stdev	pr	3	3	1	7	1	0	1	1	6	0	0	0	1	1	4	1	0	0
		std	5	1	1	4	1	0	1	6	3	0	0	0	1	1	7	1	0	0
CSP-QoS	avg	pr	15	3	1	25	1	0	13	3	1	21	0	0	14	2	0	20	0	0
		std	10	2	1	18	1	0	12	3	2	22	1	0	10	2	2	20	1	0
	stdev	pr	4	2	1	3	1	0	2	1	6	1	0	0	1	1	8	0	0	0
		std	3	2	1	8	1	0	1	6	4	0	0	0	2	1	7	1	0	0
k-shortest	avg	pr	11	3	1	20	1	0	8	2	1	15	1	0	11	2	1	17	1	0
		std	0	0	0	1	1	0	0	0	0	2	1	0	0	0	0	1	1	0
	stdev	pr	2	1	1	2	1	0	1	1	5	1	0	0	1	1	6	1	0	0
		std	0	0	0	1	0	0	1	5	3	0	0	0	0	0	1	0	0	0

Table 6: Percentage of outages observed in Dfn topology (%) (heuristic algorithm)

# of clients	Proposed-max	Proposed-fair	CSP-QoS	k-shortest
10	0.066 %	0.033 %	0 %	25.13 %
20	0 %	0 %	1.13 %	26 %
30	0 %	0 %	1.46 %	3.3 %

Table 7: Standard Deviation Values Observed in the Proposed Architecture (heuristic algorithm)

	Compuserve		Dfn	
# of clients	Prop.-max	Prop.-fair	Prop.-max	Prop.-fair
10	697	33	401	322
20	336	144	508	262
30	105	76	667	333

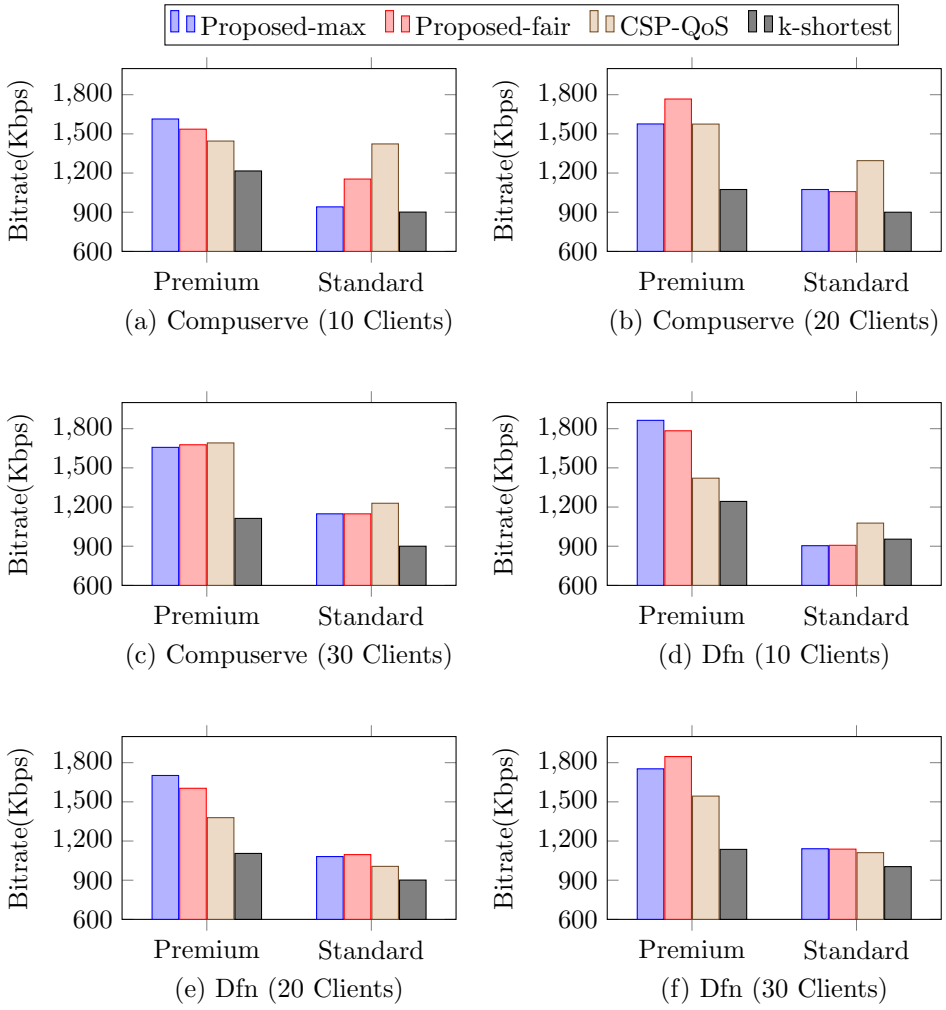
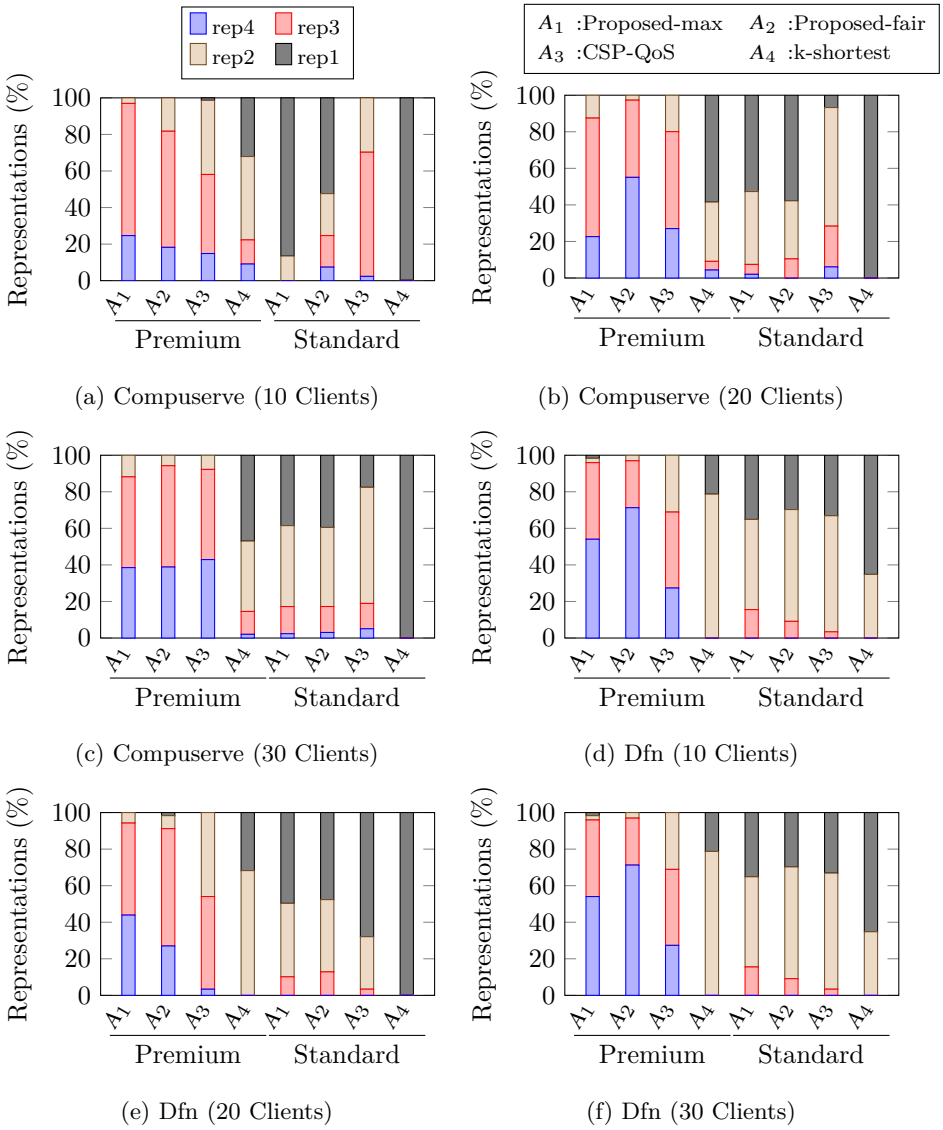


Figure 4: Average received bitrate - (a) Compuserve - 10 clients, (b) Compuserve - 20 clients, (c) Compuserve - 30 clients, (d) Dfn - 10 clients, (e) Dfn - 20 clients, (f) Dfn - 30 clients. The results are obtained by using heuristic algorithm.

individual objectives. The first objective maximizes the sum of the bandwidths allocated to premium users ($Z_1 = \sum_{i \in U_P} s_i$). The second objective is for minimizing the sum of the absolute values of the differences in bandwidth allocations



(a) Compuserve (10 Clients)

(b) Compuserve (20 Clients)

(c) Compuserve (30 Clients)

(d) Dfn (10 Clients)

(e) Dfn (20 Clients)

(f) Dfn (30 Clients)

Figure 5: Percentage of received representations - (a) Compuserve - 10 clients, (b) Compuserve - 20 clients, (c) Compuserve - 30 clients, (d) Dfn - 10 clients, (e) Dfn - 20 clients, (f) Dfn - 30 clients. The results are obtained by using heuristic algorithm.

among the premium users ($Z_2 = \sum_{i \in U_P} e_i$). The third objective maximizes the sum of the bandwidths allocated to standard users ($Z_3 = \sum_{i \in U_S} s_i$). To provide benchmarks for selecting γ_i's, we performed experiments using the Compuserve

topology with 14 users (7 premium and 7 standard users). The results we report are the averages of ten runs. The first objective function is maximized (Z_1 = 23526, Z_2 = 11433, Z_3 = 8766) when the weights are chosen as (γ_1 = 1, γ_2 = 0, γ_3 = 0). By increasing the weight of the second objective (γ_1 = 1, γ_2 = 0.5, γ_3 = 0), we obtain (Z_1 = 21869, Z_2 = 2272, Z_3 = 10141) which, clearly, shows the substantial reduction in the second objective when compared to (γ_1 = 1, γ_2 = 0, γ_3 = 0). When we double the value of γ_2 from 0.5 to 1, (γ_1 = 1, γ_2 = 1, γ_3 = 0), the reduction in Z_2, (Z_1 = 22533, Z_2 = 1514, Z_3 = 9537), is not as significant as the decrease when γ_2 increased from 0 to 0.5. Furthermore, the relative values of the weights is what matters instead of the absolute values. For example, (γ_1 = 1, γ_2 = 1, γ_3 = 0) and (γ_1 = 0.5, γ_2 = 0.5, γ_3 = 0) give exactly the same objective values. Hence, increasing γ_2 beyond 1 while keeping γ_1 = 1 does not bring significant advantage. The heuristic algorithm has only two adjustable weights (*i.e.*, γ_1 and γ_2). The weight of Z_2 cannot be changed (*i.e.*, γ_3 = 0) because of the design choice we made in the heuristic algorithm. The performance results related to the proposed approach are represented as Proposed-max for $\gamma_1 = 1, \gamma_2 = \gamma_3 = 0$ and Proposed-fair for $\gamma_1 = \gamma_2 = 1, \gamma_3 = 0$.

In the simulations, we measure the received video quality related parameters such as the received bitrate, number of received packets for each representation, the number and the duration of outages observed in the clients, the number of quality switches and fairness among the clients. All parameters used in the simulations are the same for all service differentiation approaches. The simulations are repeated over 10 times for both approaches and the results are obtained by averaging.

In Fig. 4, the average received bitrates by each client for all approaches are given with varying number of clients for both topologies. These values are obtained by averaging the received bitrates for all simulations. As observed from the figure, both types of users have achieved the offered service guarantee when the path selection is done by the proposed service differentiated architecture in both topologies. The premium users have received similar amount of bitrate

for the proposed and CSP-QoS approaches in Compuserve. The reason for such behavior is that there are only 7 paths in this topology and generally similar paths are selected for premium users with both approaches. In the simulations run over Dfn topology, the received bitrate achieved by premium users with the proposed approach is higher than that of CSP-QoS and k-shortest approaches. The performance gain in the received bitrate value reaches up to 31% and 50% when the performance of the proposed approach is compared to CSP-QoS and k-shortest path approaches, respectively. These results show that when the number of paths increases and the topology becomes larger, the proposed approach outperforms other approaches. Although the received bitrate values observed for standard users are generally similar in the proposed and CSP-QoS approaches, standard users have higher received bitrates in CSP-QoS approach for certain configurations (*e.g.*, Fig. 4a and 4b) because in our proposed approach we set $\gamma_3 = 0$. When the available bandwidth of the streaming paths determined for the standard users increases, the received bitrate values of standard users approaches to the bitrate values received by premium users. γ_3 is set to 0 to prevent this situation.

On the other hand, premium users also receive better service than that of standard users in k-shortest path based service differentiation strategy. However, unacceptable decrease in the received bitrate values of standard users are observed in the graphs with k-shortest path approach.

Frequent quality changes negatively affect the perceptual quality experienced by the clients, hence it is one of the crucial parameters related to the received video quality. In Table 4 and Table 5, average and variance of the number of quality changes observed in the clients are given for all approaches. The number of quality changes is calculated based on the number of oscillations of the selected representations. In the table, +x and -x represents the number of difference between the quality levels of current and next selected representations where quality is increased x quality level and is decreased x quality level, respectively. For example, the numbers given in the column of +1 show that how many times the clients receive the next higher representation than its current

representation. For all approaches, three oscillations are rarely observed. The decrease in the number of +1 quality level changes observed with the proposed approach is up to 40% and the number of -1 quality level changes observed with the proposed approach is up to 45% when it is compared to CSP-QoS approach. Minimum number of quality changes is observed in k-shortest path approach since the reserved bandwidth value causes clients generally receive segments of lowest quality representation and they rarely request segments of higher representations.

The percentage of received segments from each type of representation is given in Fig. 5. Especially for premium users, the perceived video quality in the proposed architecture is better than that of CSP-QoS and k-shortest path approach.

If the bandwidth is not adequate for sending the selected representation, the clients start to experience outages in video. In Table 6, the percentages of outage duration observed in a client according to the user types are given. The percentage values in the table are calculated regarding total video duration. The results show that no significant amount of duration observed in the proposed approach. The clients experience small amount of durations in CSP-QoS approach while observed duration values can be reach up to 78 seconds in the the k-shortest path approach.

Since one of the aims of the optimization framework is to provide fairness among premium users, we calculate the standard deviation of the bitrate received by these users in order to show how fairly the network resources are distributed. In Table 7, the standard deviation values are given for different γ values. When γ_2 equals 1, the difference of the standard deviation values become smaller. This shows that the optimization framework is successful in increasing the fairness (*i.e.*, all users achieve similar service quality).

5. Conclusion

In this paper, we propose a network architecture for increasing the received video quality of DASH clients and for providing service classes in an SDN do-

main. We focus on providing fairness among the clients belonging to the same service classes. Streaming paths are determined according to the optimization framework. Although the optimization framework is given for two different service classes, it can be expanded for more service classes. Furthermore, we developed a heuristic approach, which gives approximate results to the optimization framework. The proposed architecture is designed in such a way that network operators can implement it without requiring any modification on DASH client's software.

The proposed approach is compared to two service differentiation approaches. In one of the comparison approaches, streaming paths are determined by considering k-shortest paths. In the second comparison approach, the streaming paths are determined by using the solution to the constrained shortest path problem. Performance evaluations are performed for both approaches via the simulations done in real-world topologies. Simulation results reveal that the proposed architecture, in addition to providing fairness, improves the quality of the received video in terms of bitrate, segment quality, outage durations, and number of quality changes when compared to other service differentiation approaches. We observe up to 83% and 31% increase in average received bitrate with the proposed approach when compared to k-shortest path and CSP-QoS approaches, respectively. The observed average outage duration reaches up to 78 seconds for a 300 seconds of video with k-shortest path approach, while outage duration observed in the proposed approach equals to zero. On the other hand, it is observed that the proposed approach provided up to 39% decrease in the average number of quality switching when compared to CSP-QoS approach.

In our heuristic algorithm, the path and bandwidth assignment problem is divided into two phases. In the first phase, the routes for the standard users are assigned in such a way that the bandwidths of standard users are minimized while satisfying the guaranteed minimum bandwidth for standard users. In the second phase, the remaining capacity is assigned to the premium users in a way that maximize their bandwidths. However, it is possible that after the premium users obtain all the bandwidth they require, if the network has vacant bandwidth

then the remaining bandwidth can be allocated to the standard users. Hence, it is a promising future research avenue to improve/modify the heuristic to address such possible under-utilization of network resources.

Appendix A. Constraints of the MIP Model

The constraints of the optimization problem introduced in Subsection 3.2 are defined in this section. We define binary variables a_{ij}^k as flow indicators in Eq. (A.1). If a non-zero amount of data of user-k is flowing on link-(i,j) then the indicator variable is set to one (*i.e.*, $a_{ij}^k = 1$) and 0 otherwise. When $f_{ij}^k = 0$ Eq. (A.3) forces $a_{ij}^k = 0$, where ϵ is the minimum value for non-zero flows (for the sake of simplicity assume that $\epsilon = 1$ Byte for now). On the other hand, when $f_{ij}^k > 0$ Eq. (A.2) forces $a_{ij}^k = 1$.

$$a_{ij}^k \in \{0,1\} \ \forall (i,j) \in A \ \forall k \in U \tag{A.1}$$

$$f_{ij}^k \leq C_{ij} \times a_{ij}^k \ \forall k \in U \ \forall (i,j) \in A \tag{A.2}$$

$$f_{ij}^k \geq \epsilon \times a_{ij}^k \ \forall k \in U \ \forall (i,j) \in A \tag{A.3}$$

Binary variables x_{ij} given in Eq. (A.4) are utilized in Eq. (A.5) and (A.6) to determine whether link-(i,j) is used by any flows. If at least one a_{ij}^k on a particular link is equal to 1, then x_{ij} is forced to take 1, by Eq. (A.5), however if all $a_{ij}^k = 0$ then $x_{ij} \geq 0$ (by Eq. (A.5)) and $x_{ij} \leq 0$ (by Eq. (A.6)) will force $x_{ij} = 0$.

$$x_{ij} \in \{0,1\} \ \forall (i,j) \in A \tag{A.4}$$

$$x_{ij} \geq a_{ij}^k \ \forall k \in U \ \forall (i,j) \in A \tag{A.5}$$

$$x_{ij} \leq \sum_{k \in U} a_{ij}^k \ \forall (i,j) \in A \tag{A.6}$$

We need to ensure that all end-to-end paths are single path routes. Eq. (A.7) guarantees that only one link going out of the host switch (H) is used for the flow of user-k. Eq. (A.8) limits the outgoing flow of user-k at the other switches to at most one link. Since each user's data can flow on a single path s_i has the

variable bound given in Eq. (A.9).

$$\sum_{j \in N} a^k_{Hj} = 1 \ \forall k \in U \tag{A.7}$$

$$\sum_{j \in N, j \neq i} a^k_{ij} \leq 1 \ \forall k \in U, \ \forall i \in N \tag{A.8}$$

$$0 \leq s_i \leq C_{mx} \ \forall i \in U \tag{A.9}$$

Eqs. (A.10), (A.11), and (A.12), jointly guarantee the flow conservation at switches. Flow conservation constraint of host switch is given in Eq. (A.10) which states that outgoing flows of user-k at host node is equal to the source rate allocated to user-k (*i.e.*, s_k). In a similar manner, Eq. (A.11) is used to ensure that the entire flow of user-k terminates at the server (S) node. Eq. (A.12) is used to perform flow balancing at intermediate switches (M).

$$\sum_{j \in N, j \neq H} f^k_{Hj} = s_k \ \forall k \in U \tag{A.10}$$

$$s_k = \sum_{j \in N, j \neq S} f^k_{jS} \ \forall k \in U \tag{A.11}$$

$$\sum_{i \in N, i \neq j} f^k_{ij} = \sum_{l \in N, l \neq j} f^k_{jl} \ \forall k \in U \ \forall j \in M \tag{A.12}$$

To avoid any loop backs to the host and to the server, we zero all incoming flows to host node via Eq. (A.13) and from server node via Eq. (A.14), respectively.

$$\sum_{i \in N - H} f^k_{iH} = 0 \ \forall k \in U \tag{A.13}$$

$$\sum_{i \in N - S} f^k_{Si} = 0 \ \forall k \in U \tag{A.14}$$

We define the slack flow (g^k_{ij}) that represents the amount of allocated but unutilized capacity for user-k over link-(i, j). Variable bound for g^k_{ij} is the same as that of f^k_{ij} and given in Eq. (A.15). Eqs. (A.16) and (A.17) jointly ensure that the whole capacity of a particular link is allocated to real and/or slack flows if the link is utilized by at least one flow (*i.e.*, $x_{ij} = 1$). Slack flow is zero if actual

flow is also zero which is enforced by Eq. (A.18).

$$0 \leq g_{ij}^k \leq C_{mx} \ \forall (i,j) \in A \ \forall k \in U, \tag{A.15}$$

$$\sum_{k \in U} \{f_{ij}^k + g_{ij}^k\} \geq x_{ij} \times C_{ij} \ \forall (i,j) \in A \tag{A.16}$$

$$\sum_{k \in U} \{f_{ij}^k + g_{ij}^k\} \leq C_{ij} \ \forall (i,j) \in A \tag{A.17}$$

$$g_{ij}^k \leq a_{ij}^k \times C_{ij} \ \forall (i,j) \in A \ \forall k \in U \tag{A.18}$$

We cannot limit the amount of data flow in a path arbitrarily, instead, the amount of flow on a path can be limited by allocated capacity on the bottleneck link of the path. Therefore, there should be at least one link of each path with non-zero actual flow and zero slack flow. To keep the count of non-zero slack flows we introduce binary indicator variables w_{ij}^k in Eq. (A.19) which is equal to zero if $g_{ij}^k = 0$ and equal to one if $g_{ij}^k > 0$ as given in Eqs. (A.20) and (A.21). Eq. (A.22) guarantees that for each path the number of links with non-zero real flows is at least one more than the number of links with non-zero slack flows (*i.e.*, for all paths there is at least one link in the path of user-k's flow where real flow utilizes all the allocated capacity to the user).

$$w_{ij}^k \in \{0,1\} \ \forall (i,j) \in A \ \forall k \in U \tag{A.19}$$

$$g_{ij}^k \leq w_{ij}^k \times C_{ij} \ \forall k \in U \ \forall (i,j) \in A \tag{A.20}$$

$$g_{ij}^k \geq \epsilon \times w_{ij}^k \ \forall k \in U \ \forall (i,j) \in A \tag{A.21}$$

$$\sum_{(i,j) \in A} w_{ij}^k \leq \sum_{(i,j) \in A} a_{ij}^k - 1 \ \forall k \in U \tag{A.22}$$

All flows utilizing the same link are allocated the same amount of bandwidth on the link (whether they utilize all the allocated capacity or not). Eq. (A.23) guarantees that all non zero flows utilizing the same link get equal allocations for the link. To further clarify this constraint consider two users' paths (*e.g.*, user-2 and user 4) utilizing link-$(5,8)$ and none of the other users' paths utilize link-$(5,8)$. For this particular scenario, Eq. (A.23) effectively reduces to $f_{58}^2 + g_{58}^2 \leq f_{58}^4 + g_{58}^4$ and $f_{58}^4 + g_{58}^4 \leq f_{58}^2 + g_{58}^2$ (*i.e.*, $f_{58}^2 + g_{58}^2 = f_{58}^4 + g_{58}^4$).

$$f_{ij}^k + g_{ij}^k \leq f_{ij}^h + g_{ij}^h + C_{ij}(2 - a_{ij}^k - a_{ij}^h) \ \forall (i,j) \in A \ \forall k,h \in U \tag{A.23}$$

Eq. (A.24) guarantees that data flow for premium users is larger than standard users. Furthermore, Eq. (A.25) and (A.26) ensure that data flows for premium and standard users are higher than or equal to the guaranteed bit rate (ζ_g) and the minimum bit rate (ζ_m), respectively. At this point we can give the exact value of the ϵ introduced earlier which is $\epsilon = \zeta_m$.

$$s_k > s_h \ \forall k \in U_P \ \forall h \in U_S \tag{A.24}$$

$$s_k \geq \zeta_g \ \forall k \in U_P \tag{A.25}$$

$$s_h \geq \zeta_m \ \forall h \in U_S \tag{A.26}$$

While solving the optimization problem, we observe that phantom flows exist in the solution which are invalid flows that do not violate flow balancing constraints (*e.g.*, node-6, node-7, and node-8 creating a data flow loop among themselves), however, renders some other constraints ineffective. Hence, to eliminate such flows we determine the hop count of relay nodes from the host node and make sure that a node with higher hop count cannot transmit data to another node with lower hop count, thereby, eliminating phantom flows. We first introduce binary indicator variables p_i^k in Eq. (A.27) which are one if switch-i is a relay for user-k's flow and zero otherwise. Since host and server nodes are always relays for user-k's flow, p_H^k and p_S^k are set to one as stated in Eq. (A.28) and (A.29). At the other switches, if any of the incoming links transport user-k's flow to switch-i (*i.e.*, $a_{ji}^k = 1$) then switch-i is also a relay node for user-k's flow as expressed in Eq. (A.30).

$$p_i^k \in \{0, 1\} \ \forall k \in U \ \forall i \in N \tag{A.27}$$

$$p_H^k = 1 \ \forall k \in U \tag{A.28}$$

$$p_S^k = 1 \ \forall k \in U \tag{A.29}$$

$$p_i^k = \sum_{j \in N, i \neq j} a_{ji}^k \ \forall k \in U \ \forall i \in M \tag{A.30}$$

We determine the degree of switch-i as a relay for user-k's flow (*i.e.*, the hop count from the host) by using the integer variables r_i^k. Variable bound on r_i^k is given in Eq. (A.31). Note that the maximum degree of a switch can at most be the total number of switches, $|N|$, in the network. The degree of the host is always one as stated in Eq. (A.32) and the degree of the server equals to one more than the total number of relay switches on user-k's path as stated in Eq. (A.33).

$$0 \leq r_i^k \leq |N| \ \forall k \in U \ \forall i \in N \tag{A.31}$$

$$r_H^k = 1 \ \forall k \in U \tag{A.32}$$

$$r_S^k = 1 + \sum_{(i,j)\in A} a_{ij}^k \ \forall k \in U \tag{A.33}$$

We set the degrees of switches which are not relaying the data of user-k by using Eq. (A.34) which states that if switch-i is not a relay for user-k's data (*i.e.*, $p_i^k = 0$) then $r_i^k = 0$, however, if switch-i is a relay for user-k's data (*i.e.*, $p_i^k = 1$) then $r_i^k \leq |N|$ which is satisfied by all switches. If switch-i and switch-j are communicating directly for relaying the data of user-k then the link between them, link-(i, j), must be used to transport non-zero flow (*i.e.*, $a_{ji}^k = 1$) otherwise $a_{ji}^k = 0$. Eq. (A.35) and (A.36) jointly result in $r_i^k = r_j^k + a_{ji}^k$ for $a_{ji}^k = 1$ (*i.e.*, both $r_i^k \leq r_j^k + a_{ji}^k$ and $r_i^k \geq r_j^k + a_{ji}^k$ are true). If $a_{ji}^k = 0$ then Eq. (A.35) and (A.36) are always satisfied (*i.e.*, they become redundant). Nevertheless, the degree of a switch is one more than the switch it receives data from as established by Eq. (A.35) and (A.36).

$$r_i^k \leq |N| p_i^k \ \forall k \in U \ \forall i \in M \tag{A.34}$$

$$r_i^k \geq r_j^k + a_{ji}^k + |N|(a_{ji}^k - 1) \ \forall k \in U \ \forall j \in N \ \forall i \in N - j \tag{A.35}$$

$$r_i^k \leq r_j^k + a_{ji}^k + |N|(1 - a_{ji}^k) \ \forall k \in U \ \forall j \in N \ \forall i \in N - j \tag{A.36}$$

The second term in the objective function is to minimize the differences between the allocated rates to the premium users. In fact, minimizing the differences allocated to different users is a commonly adopted objective for achieving

fairness [36]. For example in [37], RMS (Root Mean Square) difference between the rates allocated to different users are minimized in the context of streaming video. Minimizing the sum of absolute values of differences of rates allocated to different users also serves in achieving fairness [38]. Furthermore, by employing such a metric we can keep our model as a linear model. For this purpose, we introduce variables d_i (*i.e.*, difference of user-k's data rate and the average data rate of all premium users) which are assigned values as expressed in Eq. (A.37). Note that $-C_{mx} \leq d_i \leq C_{mx}$ as stated in Eq. (A.38) which is a direct result of variable bound on s_k in Eq. (A.9).

$$d_i = s_i - \frac{1}{|U_P|} \sum_{k \in U_P} s_k \ \forall i \in U_P \tag{A.37}$$

$$-C_{mx} \leq d_i \leq C_{mx} \ \forall i \in U \tag{A.38}$$

Since we need to find the absolute values of d_i's (*i.e.*, $e_i = |d_i|$) which is achieved by utilizing Eq. (A.41)–(A.44). We utilize binary variables b_i given in Eq. (A.41) for obtaining the absolute values of d_i's. The variable bound on e_i's are given in Eq. (A.40). When d_i is positive, Eq. (A.41) and (A.44) always hold, however, for Eq. (A.42) to hold binary indicator variable b_i must be set to zero, hence, the effective constraints are Eq. (A.41) and (A.43) in this case (*i.e.*, $d_i \leq e_i$ and $d_i \geq e_i$) which results in $d_i = e_i$. When d_i is negative, for Eq. (A.41) to hold, $b_i = 1$ which makes Eq. (A.42) and (A.44) the effective constraints (*i.e.*, $-d_i \leq e_i$ and $-d_i \geq e_i$) resulting in $-d_i = e_i$.

$$b_i \in \{0, 1\} \ \forall i \in U \tag{A.39}$$

$$0 \leq e_i \leq C_{mx} \ \forall i \in U \tag{A.40}$$

$$d_i + 2 \times C_{mx} \times b_i \geq e_i \ \forall i \in U_P \tag{A.41}$$

$$-d_i + 2 \times C_{mx} \times (1 - b_i) \geq e_i \ \forall i \in U_P \tag{A.42}$$

$$d_i \leq e_i \ \forall i \in U_P \tag{A.43}$$

$$-d_i \leq e_i \ \forall i \in U_P \tag{A.44}$$

[1] I. Sodagar, The MPEG-DASH standard for multimedia streaming over the internet, IEEE MultiMedia 18 (4) (2011) 62–67.

[2] B. Rainer, S. Lederer, C. Muller, C. Timmerer, A seamless web integration of adaptive HTTP streaming, in: Proc. European Signal Processing Conference (EUSIPCO), 2012, pp. 1519–1523.

[3] Z. Li, X. Zhu, J. Gahm, R. Pan, H. Hu, A. C. Begen, D. Oran, Probe and adapt: Rate adaptation for HTTP video streaming at scale, IEEE Journal on Selected Areas in Communications 32 (4) (2014) 719–733.

[4] J. Jiang, V. Sekar, H. Zhang, Improving fairness, efficiency, and stability in HTTP-based adaptive video streaming with festive, IEEE/ACM Transactions on Networking 22 (1) (2014) 326–340.

[5] T.-Y. Huang, R. Johari, N. McKeown, M. Trunnell, M. Watson, A buffer-based approach to rate adaptation: Evidence from a large video streaming service, in: Proc. ACM SIGCOMM, 2014, pp. 187–198.

[6] L. D. Cicco, V. Caldaralo, V. Palmisano, S. Mascolo, ELASTIC: A client-side controller for dynamic adaptive streaming over HTTP, in: 2013 20th Int'l Packet Video Workshop, 2013, pp. 1–8.

[7] R. Jmal, L. C. Fourat, Content-centric networking management based on software defined networks: Survey, IEEE Transactions on Network and Service Management 14 (4) (2017) 1128–1142.

[8] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, J. Turner, Openflow: Enabling innovation in campus networks, SIGCOMM Comput. Commun. Rev. 38 (2) (2008) 69–74.

[9] S. Velrajan, Application-aware routing in software-defined Networks, Tech. rep., Aricent (2014).

[10] Z. A. Qazi, J. Lee, T. Jin, G. Bellala, M. Arndt, G. Noubir, Application-awareness in sdn, SIGCOMM Comput. Commun. Rev. 43 (4) (2013) 487–488.

[11] A. Kassler, L. Skorin-Kapov, O. Dobrijevic, M. Matijasevic, P. Dely, Towards qoe-driven multimedia service negotiation and path optimization with software defined networking, in: SoftCOM, IEEE, 2012, pp. 1–5.

[12] P. McDonagh, C. Olariu, A. Hava, C. Thorpe, Enabling IPTV service assurance using openflow, in: AINA Workshops, IEEE Computer Society, 2013, pp. 1456–1460.

[13] J. Huang, L. Xu, M. Zeng, C. C. Xing, Q. Duan, Y. Yan, Hybrid scheduling for quality of service guarantee in software defined networks to support multimedia cloud services, in: Services Computing (SCC), 2015 IEEE International Conference on, 2015, pp. 788–792.

[14] O. Dobrijevic, A. J. Kassler, L. Skorin-Kapov, M. Matijasevic, Q-POINT: QoE-Driven Path Optimization Model for Multimedia Services, Springer International Publishing, 2014, pp. 134–147.

[15] S. Laga, T. V. Cleemput, F. V. Raemdonck, F. Vanhoutte, N. Bouten, M. Claeys, F. D. Turck, Optimizing scalable video delivery through OpenFlow layer-based routing, in: NOMS, IEEE, 2014, pp. 1–4.

[16] H. Egilmez, S. Civanlar, A. Tekalp, An optimization framework for qos-enabled adaptive video streaming over openflow networks, IEEE Transactions on Multimedia 15 (3) (2013) 710–715.

[17] H. E. Egilmez, A. M. Tekalp, Distributed QoS architectures for multimedia streaming over software defined networks, IEEE Transactions on Multimedia 16 (6) (2014) 1597–1609.

[18] T. Uzakgider, C. Cetinkaya, M. Sayit, Learning-based approach for layered adaptive video streaming over sdn, Comput. Netw. 92 (P2) (2015) 357–368.

[19] N. Xue, X. Chen, L. Gong, S. Li, D. Hu, Z. Zhu, Demonstration of OpenFlow-controlled network orchestration for adaptive SVC video manycast, IEEE Trans. Multimedia 17 (9) (2015) 1617–1629.

[20] R. Dubin, O. Hadar, Y. Freifeld, A. Ruham, A. Dvir, N. Harel, R. Barkan, Hybrid clustered peer-assisted DASH-SVC system, in: 2015 IEEE International Conference on Computer and Information Technology, 2015, pp. 1651–1656.

[21] C. Ozcinar, E. Ekmekcioglu, J. Ćalić, A. Kondoz, Adaptive delivery of immersive 3D multi-view video over the internet, Multimedia Tools and Applications 75 (20) (2016) 12431–12461.

[22] A. Seema, L. Schwoebel, T. Shah, J. Morgan, M. Reisslein, WVSNP-DASH: Name-based segmented video streaming, IEEE Transactions on Broadcasting 61 (3) (2015) 346–355.

[23] M. F. Majeed, S. H. Ahmed, S. Muhammad, H. Song, D. B. Rawat, Multimedia streaming in information-centric networking: A survey and future perspectives, Computer Networks 125 (2017) 103 – 121.

[24] M. S. Seddiki, M. Shahbaz, S. Donovan, S. Grover, M. Park, N. Feamster, Y.-Q. Song, FlowQoS: QoS for the rest of us, in: Proc. Workshop Hot Topics in Softw. Defined Netw., 2014, pp. 207–208.

[25] P. Georgopoulos, Y. Elkhatib, M. Broadbent, M. Mu, N. Race, Towards network-wide qoe fairness using openflow-assisted adaptive video streaming, in: ACM SIGCOMM Workshop on Future Human-centric Multimedia Networking, 2013, pp. 15–20.

[26] S. Petrangeli, T. Wauters, R. Huysegems, T. Bostoen, F. D. Turck, Network-based dynamic prioritization of HTTP adaptive streams to avoid video freezes, in: IEEE/IFIP International Workshop on Quality of Experience Centric Management, IEEE, 2015, pp. 1242–1248.

[27] A. Bentaleb, A. C. Begen, R. Zimmermann, S. Harous, SDNHAS: An SDN-enabled architecture to optimize QoE in HTTP adaptive streaming, IEEE Transactions on Multimedia 19 (10) (2017) 2136–2151.

[28] J. W. Kleinrouweler, S. Cabrero, 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, MMSys '16, 2016, pp. 4:1–4:10.

[29] G. Cofano, L. De Cicco, T. Zinner, A. Nguyen-Ngoc, P. Tran-Gia, S. Mascolo, Design and experimental evaluation of network-assisted strategies for HTTP adaptive streaming, in: Proceedings of the 7th International Conference on Multimedia Systems, MMSys '16, 2016, pp. 3:1–3:12.

[30] H. Nam, K. H. Kim, J. Y. Kim, H. Schulzrinne, Towards qoe-aware video streaming using sdn, in: IEEE Globecom, 2014, pp. 1317–1322.

[31] K. T. Bagci, K. E. Sahin, A. M. Tekalp, Queue-allocation optimization for adaptive video streaming over software defined networks with multiple service-levels, in: 2016 IEEE International Conference on Image Processing (ICIP), 2016, pp. 1519–1523.

[32] K. T. Bagci, K. E. Sahin, A. M. Tekalp, Compete or collaborate: Architectures for collaborative dash video over future networks, IEEE Transactions on Multimedia 19 (10) (2017) 2152–2165.

[33] C. Cetinkaya, E. Karayer, M. Sayit, C. Hellge, Sdn for segment based flow routing of dash, in: IEEE Consumer Electronics Berlin (ICCE-Berlin), IEEE, 2014, pp. 74–77.

[34] M. M. Baldi, T. G. Crainic, G. Perboli, R. Tadei, The generalized bin packing problem, Transportation Research Part E: Logistics and Transportation Review 48 (6) (2012) 1205 – 1220.

[35] S. Lederer, C. Müller, C. Timmerer, Dynamic adaptive streaming over http dataset, MMSys '12, 2012, pp. 89–94.

[36] J. J. Quinlan, A. H. Zahran, K. K. Ramakrishnan, C. J. Sreenan, Delivery of adaptive bit rate video: balancing fairness, efficiency and quality, in: Proc. IEEE LANMAN, 2015, pp. 1–6.

[37] M. Mu, M. Broadbent, A. Farshad, N. Hart, D. Hutchison, Q. Ni, N. Race, A scalable user fairness model for adaptive video streaming over SDN-assisted future networks, IEEE Journal on Selected Areas in Communications 34 (8) (2016) 2168–2184.

[38] H. Shi, R. V. Prasad, E. Onur, I. G. M. M. Niemegeers, Fairness in wireless networks:issues, measures and challenges, IEEE Communications Surveys Tutorials 16 (1) (2014) 5–24.