EU GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCE

EGE UNIVERSITY

PhD THESIS

INTEGRATION OF ALTO AND SDN PROTOCOLS FOR PEER TO PEER APPLICATIONS

Cihat ÇETİNKAYA

Supervisor: Assoc. Prof. Dr. Müge SAYIT

International Computer Department

Presentation Date: 18.09.2017

Bornova-İZMİR
2017

EGE UNIVERSITY GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCE

(PHD THESIS)

INTEGRATION OF ALTO AND SDN PROTOCOLS FOR PEER TO PEER APPLICATIONS

Cihat ÇETİNKAYA

Supervisor: Assoc. Prof. Dr. Müge SAYIT

International Computer Department

Presentation Date : 18.09.2017

Bornova-İZMİR
2017

Cihat ÇETİNKAYA tarafından Doktora tezi olarak sunulan "Integration of ALTO and SDN protocols for Peer to Peer Applications" başlıklı bu çalışma EÜ Lisansüstü Eğitim ve Öğretim Yönetmeliği ile EÜ Fen Bilimleri Enstitüsü Eğitim ve Öğretim Yönergesi'nin ilgili hükümleri uyarınca tarafımızdan değerlendirilerek savunmaya değer bulunmuş ve 18.09.2017 tarihinde yapılan tez savunma sınavında aday oybirliği/~~oyçokluğu~~ ile başarılı bulunmuştur.

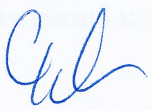
Jüri Üyeleri :		**İmza**
Jüri Başkanı	: Doç. Dr. Müge SAYIT	.....
Raporter Üye	: Prof. Dr. Mehmet Emin DALKILIÇ	.....
Üye	: Prof. Dr. Bülent TAVLI	.....
Üye	: Prof. Dr. Bahar KARAOĞLAN	.....
Üye	: Yrd. Doç. Dr. Enis KARAARSLAN	.....

EGE ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ

ETİK KURALLARA UYGUNLUK BEYANI

EÜ Lisansüstü Eğitim ve Öğretim Yönetmeliğinin ilgili hükümleri uyarınca Doktora Tezi olarak sunduğum "Integration of ALTO and SDN protocols for Peer to Peer Applications" başlıklı bu tezin kendi çalışmam olduğunu, sunduğum tüm sonuç, doküman, bilgi ve belgeleri bizzat ve bu tez çalışması kapsamında elde ettiğimi, bu tez çalışmasıyla elde edilmeyen bütün bilgi ve yorumlara atıf yaptığımı ve bunları kaynaklar listesinde usulüne uygun olarak verdiğimi, tez çalışması ve yazımı sırasında patent ve telif haklarını ihlal edici bir davranışımın olmadığını, bu tezin herhangi bir bölümünü bu üniversite veya diğer bir üniversitede başka bir tez çalışması içinde sunmadığımı, bu tezin planlanmasından yazımına kadar bütün safhalarda bilimsel etik kurallarına uygun olarak davrandığımı ve aksinin ortaya çıkması durumunda her türlü yasal sonucu kabul edeceğimi beyan ederim.

18.09.2017



Cihat ÇETİNKAYA

ÖZET

GÖREVDEŞ UYGULAMALAR İÇİN ALTO VE YAZILIM TANIMLI AĞ MİMARİSİ PROTOKOLLERİNİN ENTEGRASYONU.

ÇETİNKAYA, Cihat

Doktora Tezi, Uluslararası Bilgisayar Anabilim Dalı
Tez Danışmanı: Doç. Dr. Müge SAYIT
Eylül 2017, 82 sayfa

Yazılım Tanımlı Ağ (YTA), bilgisayar ağlarının veri ve kontrol düzlemlerini birbirinden ayrılmasıyla ortaya çıkan yeni bir ağ mimarisidir. Bu ayrıştırma, ağ üzerinde çalışan uygulamaların gereksinimleri göz önünde bulundurularak belirli ağ operasyonlarını geliştirmeyi olanak sağlar. Bu tezde temel olarak, YTA üzerinde koşan video akışlandırma uygulamalarının performanslarının iyileştirilmesi hedeflenmiştir. Bu amaç için ilk olarak, YTA üzerinde koşan HTTP üzerinden dinamik uyarlamalı video akışlandırma kullanan istemcilerin elde ettikleri deneyim kalitesini arttırmak için rota seçimine ilişkin dört farklı yöntem önerilmektedir. İkinci olarak ise, YTA ve Uygulama Katmanı Trafik Eniyilemesi (UKTE) protokolünün bir arada çalıştığı video akışlandırma sistem mimarisi önerilmektedir. Önerilen mimari üzerinde, istemciler için hem uygun video sunucusu hem de akışlandırma rotası seçimine ilişkin iki farklı yöntem önerilmektedir. Benzetim sonuçları, YTA üzerinde çalışan video akışlandırma uygulamalarının geleneksel video akışlandırma uygulamalarına göre daha iyi performansa sahip olduğunu göstermektedir.

Anahtar sözcükler: Yazılım Tanımlı Ağlar, Uygulama Katmanı Trafik Eniyilemesi, HTTP üzerinden dinamik uyarlamalı video akışlandırma, deneyim kalitesi, OpenFlow, Rota seçimi, Sunucu seçimi

ABSTRACT

INTEGRATION OF ALTO AND SDN PROTOCOLS FOR PEER TO PEER APPLICATIONS

ÇETİNKAYA, Cihat

PhD in International Computer Department
Supervisor: Assoc. Prof. Dr. Müge SAYIT
September 2017, 82 pages

Software Defined Networking (SDN) is a recently emerged network architecture by decoupling the control and forwarding planes of computer networks. This seperation enables to develop specific network operations by considering the requierement of the applications running over the network. In general, this thesis aims to improve the performance of video streaming applications running over SDN. For this purpose first, four different route selection methods are proposed to increase the quality of experience (QoE) received by clients using Dynamic Adaptive Video Streaming over HTTP (DASH) running over SDN. Secondly, a video-on-demand (VoD) system architecture based on SDN and Application Layer Traffic Optimization (ALTO) protocol interoperability is proposed. By using the proposed VoD architecture, two different methods are proposed for both the suitable video server and streaming route selection. Simulation results show that video streaming applications running over SDN have better performance when compared to traditional video streaming applications.

Keywords: Software-defined Networks, Application Layer Traffic Optimization, Dynamic Adaptive Streaming over HTTP, Quality of Experience, OpenFlow, Route selection, Server selection

ACKNOWLEDGEMENT

First of all, I would like to express my special appreciation and thanks to my advisor Assoc. Prof. Dr. Müge SAYIT (International Computer Institute, Ege University, Izmir, Turkey) for her academic guidance and never-ending support which motivated me throughout the process. Her constructive feedback and sincere communication encouraged me to make it across the finishing line.

I would also like to thank my committee members Prof. Dr. Mehmet Emin DALKILIÇ (International Computer Institute, Ege University, Izmir, Turkey) and Assoc. Prof. Dr. Oğuz SUNAY (Department of Electrical Electronics Engineering, Özyeğin University, Istanbul, Turkey) for their valuable contributions to the thesis. My sincere thanks also goes to Prof. Dr. Bülent TAVLI (Department of Electrical Electronics Engineering, TOBB University of Economics, Ankara, Turkey) and Assist. Prof. Dr. Hüseyin Uğur Yıldız (Department of Electrical Electronics Engineering, TED University, Ankara, Turkey) for their great support to overcome the hard problems that I faced during the thesis.

I thank all the members in the Multimedia Communication Research Group (International Computer Institute, Ege University, Izmir, Turkey) for the stimulating discussions, for the sleepless nights we were working together before deadlines, and for all the fun we have had in the last six years. I would also like to express my gratitude to all my colleagues.

I would also like to thank TUBITAK-EEEAG for funding this thesis under project number 114E409 and 115E449, and TUBITAK-BIDEB 2211-C for supporting my Ph.D. study.

Last but not the least, I would like to especially thank my precious wife Eda Burcu ÇETİNKAYA for supporting me in every aspect of my life. My work in this thesis is dedicated to her.

TABLE OF CONTENTS

TABLE OF CONTENTS (continued)

TABLE OF CONTENTS (continued)

Page

LIST OF FIGURES

LIST OF FIGURES (continued)

LIST OF FIGURES (continued)

LIST OF TABLES

LIST OF TABLES (continued)

LIST OF ABBREVIATIONS

Abbreviation	Explanation
ALTO	Application Layer Traffic Optimization
BGP	Border Gateway Protocol
CDF	Cumulative Distribution Function
CDN	Content Delivery Network
DASH	Dynamic Adaptive Streaming Over HTTP
DNS	Domain Name Service
GAMS	General Algebraic Modeling System
HAS	HTTP Adaptive Streaming
HTTP	Hyper-Text Transfer Protocol
IETF	Internet Engineering Task Force
ISP	Internet Service Provider
MIP	Mixed Integer Programming
MPD	Media Presentation Description
MPEG	Moving Picture Experts Group
OTT	Over-The-Top
PID	Provider-Defined Identifiers
QoE	Quality Of Experience
QoS	Quality Of Service
RTP	Real-Time Transport Protocol
SDN	Software Defined Networking
SLA	Service Level Agreement

LIST OF ABBREVIATIONS (continued)

Abbreviation	Explanation
SVC	Scalable Video Coding
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
VoD	Video-On-Demand
WAN	Wide Area Network

1. INTRODUCTION

Software-defined Networking (SDN) is a recently emerged network architecture by proposing the seperation of control plane and data plane of the computer networks (Open Networking Foundation, 2012). The seperation of the control plane (where forwarding decisions are made) from the data plane (where packets are forwarded) makes possible the network directly programmable, virtualizable, and managable from a centralized point. One of the key advantages of the SDN is that it enables network operators to develop specific routing algorithms and service negotiations according to the requirements of the applications running on the network.

Cisco estimates that IP video traffic will dominate the Internet bandwidth usage with a 82% of all consumer Internet traffic by 2021 (Cisco, 2017). The increasing demand for the video leads researchers to develop new video streaming applications by considering the underlying network conditions in order to provide a better performance in terms of Quality of Experience (QoE). Hyper-Text Transfer Protocol (HTTP) adaptive streaming has been the most prevalent approach preferred in Internet video streaming applications such as Youtube[1], Netflix[2] and Hulu[3] recently. The main reasons of this tendency are based on HTTP advantages such as usage of existing caching infrastructure and firewall traversal. Moving Picture Experts Group (MPEG) has standardized Dynamic Adaptive Streaming over HTTP (DASH) (Sodagar, 2011) as the only available international standard for HTTP adaptive streaming. In DASH, the video is encoded at various qualities and the client adapts the quality of the video over the time.

Cisco also predicts that 71% of the video traffic will be served by the servers within the Content Delivery Network (CDN) by 2021. Therefore, it is also crucial for a video streaming application to benefit from network related information when directing the clients to appropriate servers in the CDN. A working group in the Internet Engineering Task Force (IETF) has proposed Application Layer Traffic Optimization (ALTO) protocol to provide network related information such as hop-count, available bandwidth, routing-cost to the applications running on the Internet.

The aim of this study is to improve the performance of the video streaming applications utilizing SDN. The studies proposed in this thesis can be categorized in two classes:

[1] www.youtube.com
[2] www.netflix.com
[3] www.hulu.com

- First, with the advantage of developing specific routing algorithms enabled by the SDN, new route selection methods are proposed in order to increase the QoE of the DASH clients in an SDN domain.
- Second, a video streaming architecture is proposed utilizing SDN and ALTO integration in which the video servers can span across a wide area network (WAN).

1.1 Contributions Of The Thesis

In the first class of the proposed studies, four route selection methods are proposed in order to increase the QoE of the DASH clients. The first method aims to improve the QoE of the clients by increasing the average received bitrate and to provide fairness among clients by decreasing the difference between the bitrates received by the clients. In the second method, an optimization framework is proposed for increasing QoE of DASH clients and for providing service classes in an SDN domain. A novel segment-based route selection method by considering the bitrate of the requested segment is proposed in the third method. Fourth, based on the third method, an algorithm for determining the streaming paths for each client in the system by considering the bitrates of the future segments is proposed.

The second class of the proposed studies focuses on utilizing SDN and ALTO for video streaming applications. For this purpose first, the integration of SDN and ALTO is proposed for multimedia services. Second, a CDN-based Video-on-Demand (VoD) architecture is proposed based on the SDN and ALTO integrated multimedia service. Third, a method is proposed to improve the QoE of the clients for suitable CDN server selection process in the CDN-based VoD architecture. In the fourth study, differing from the third, the proposed method aims to improve the QoE of the clients while reducing the cost of Internet Service Providers (ISP) for suitable CDN server selection process. Table 1.1 gives the list of the publications that were produced during the thesis study.

1.2 Organization Of The Thesis

This thesis is organized as follows:

- In Chapter 2, the SDN controller developed for the studies proposed in the thesis are explained.

Table 1.1. Publications produced during the thesis study

No	Title	Book Title	Status
1	SDN for Segment based Flow Routing of DASH **(Best Paper Award)**	IEEE 4th International Conference on Consumer Electronics (ICCE-Berlin), 2014.	Published
2	Route Optimization for DASH over Software Defined Networks	IEEE 23rd Signal Processing and Communications Applications Conference (SIU), 2015.	Published
3	Video-on-Demand System Architecture with ALTO-SDN Integration **(STG Award)**	IEEE 2016 International Black Sea Conference on Communications and Networking (BlackSeaCom), 2016.	Published
4	ALTO-assisted CDN-based Video Streaming over SDN	IEEE 25th Signal Processing and Communications Applications Conference (SIU), 2017.	Published
5	Demonstration of SDN & ALTO Interoperability for Multimedia Services	IEEE 3rd International Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), 2017	Submitted
6	Segment-aware Dynamic Routing for DASH Flows over Software Defined Networks	Elsevier, Journal of Network and Computer Applications	Submitted
7	DASH-QoS: A Service Differentiation Architecture for DASH over Software Defined Networks	IEEE Transactions on Multimedia	Submitted
8	SDN-ALTO Collaboration For CDN-based Layered Video Streaming	IEEE/ACM Transactions on Networking	Draft
9	Dynamic Server Selection for VoD Systems Utilizing SDN-ALTO Integration	Elsevier, Computer Networks Journal	Draft

- Chapter 3 describes the proposed route selection methods for DASH over SDN.
- Chapter 4 presents the proposed video streaming architecture with SDN-ALTO integration.
- Chapter 5 concludes the thesis and discusses the future works.

2. SOFTWARE-DEFINED NETWORKING

In this chapter, a brief introduction to SDN and the SDN controller modules developed for the studies proposed in the thesis are given.

SDN is a recently emerged network architecture introducing the separation of the data plane and the control plane of the computer networks (Open Networking Foundation, 2012). In the SDN architecture, the complexity and the costs of a network are reduced, therefore network management becomes easier and more flexible. SDN capabilites enable network operators to have flexible control over network services such as routing, traffic engineering and quality of service (QoS). The overview of the SDN architecture is given in Fig. 2.1. The SDN architecture divided into three planes. The data plane consists of network devices such as routers, switches and firewalls. Differing from the conventional network architecture, the data plane does not make forwarding decisions. Instead, the data plane receives the forwarding information from control plane through a Southbound API. The control plane is the most important element of the SDN architecture. A device called controller is responsible for the management of the network. The controller talks to the devices in data plane through the Southbound API. OpenFlow (McKeown et al., 2008) is the standard protocol communicating the controller and data plane. The application plane is the layer where services and the policies of the network are implemented. The control plane and the application plane communicates through a Northbound API. In the next section, the design of the SDN controller is given.

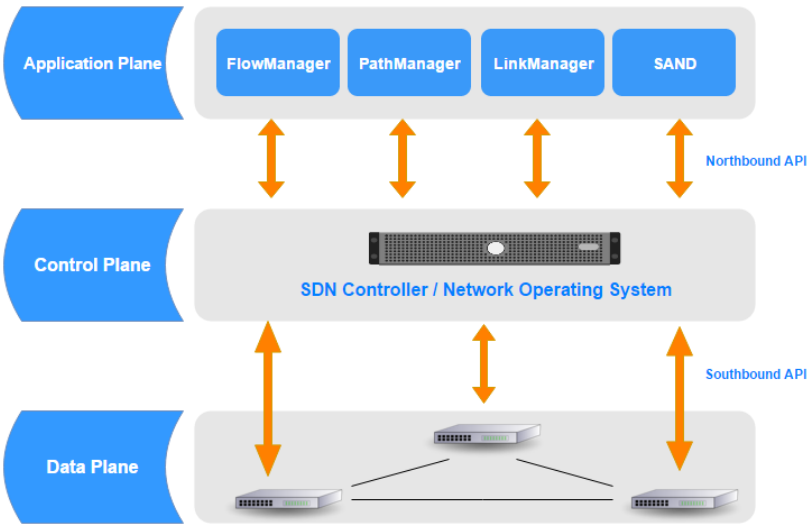


Figure 2.1. Overview of the SDN Architecture.

2.1 Design Of The SDN Controller Modules

Fig. 2.2 shows the overall design of the developed SDN controller modules. In the architecture, the forwarding layer (data plane) consists of clients, servers and OpenFlow switches which is emulated using Mininet (Lantz et al., 2010) software. The SDN controller modules are implemented on Floodlight[1].

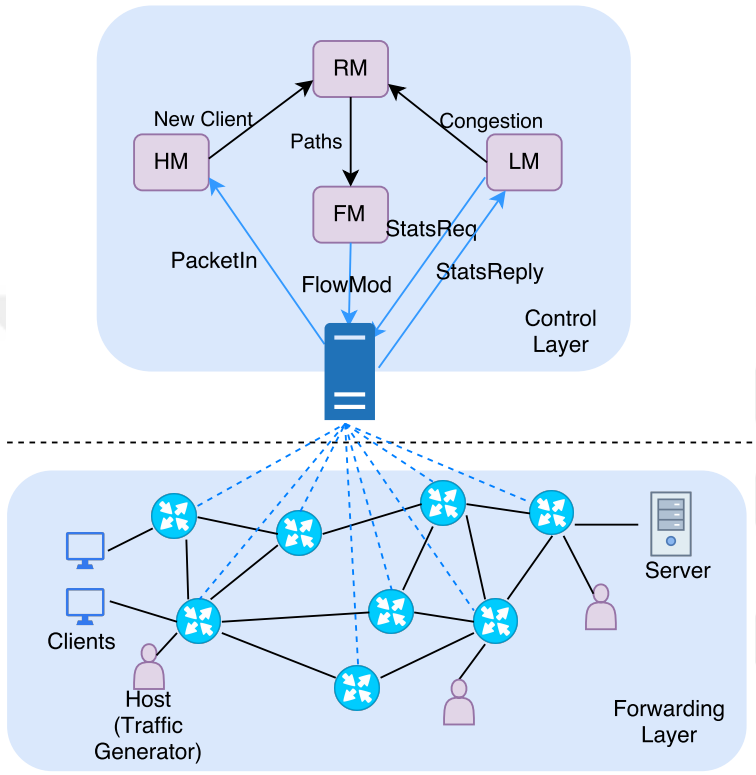


Figure 2.2. Design of the SDN controller.

- **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 video streaming application, it establishes a Transmission Control Protocol (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* period-

[1]http://www.projectfloodlight.org/floodlight/

ically. If congestion occurs on a link, it informs the RouteManager module with the new values of available bandwidth of the links. The formula of the available bandwidth calculation of a link is given in Equation (2.1) and Equation (2.2). LM module obtains the current traffic on the links after receiving the *StatsReply* message from the switches. By using the smoothing formula given in Equation (2.1), the estimated traffic is calculated. Finally available bandwidth value of a link l, which is denoted as abw_l , is calculated by using formula (2.2)

$$tr_l = \alpha \times (recently\_measured\_traffic) + (1 - \alpha) \times tr_{l-1} \tag{2.1}$$

$$abw_l = capacity_l - tr_l \tag{2.2}$$

α is set to 0.6 to give a slightly more importance to the recently measured traffic (Cetinkaya et al., 2014).

- **RouteManager (RM)** : It is invoked by the HM when a new client joins the system or by the LM module when congestion occurs. It is responsible for the assignment of the streaming paths based on requirement of the video streaming system. 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.

Table 2.1 gives the list of publications -each of which is coauthored by the author of this thesis- that uses the SDN controller which is developed in this thesis study.

Table 2.1. Publications produced using the proposed SDN controller.

No	Title	Book Title	Status
1	Design of A Layer-based Video Streaming System over Software-Defined Networks	8th International Conference on the Network of the Future (NoF17), 2017	Accepted
2	Towards QoS-aware Routing for DASH Utilizing MPTCP over SDN	IEEE 3rd International Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), 2017	Accepted
3	Evaluation of MPTCP Congestion Control for DASH	IEEE 7th International Conference on Consumer Electronics-Berlin (ICCE-Berlin 2017)	Accepted
4	Rate Adaptation Algorithm with Backward Quality Increasing Property for SVC-DASH **(Distinguished Paper Award)**	IEEE 7th International Conference on Consumer Electronics-Berlin (ICCE-Berlin 2017)	Accepted
5	Server Selection For Video Streaming Applications Over Software Defined Networks	IEEE 24th Signal Processing and Communications Applications Conference (SIU), 2016.	Published
6	Learning-based Approach for Layered Adaptive Video Streaming over SDN	Elsevier, Computer Networks, Volume 92, Part 2, 2015, Pages 357-368, ISSN 1389-1286	Published
7	An SDN-assisted System Design for Improving Performance of SVC-DASH	Federated Conference on Computer Science and Information Systems (FedCSIS), 2015	Published
8	Performance Evaluation of DASH Rate Adaptation Algorithms	IEEE 23rd Signal Processing and Communications Applications Conference (SIU), 2015.	Published

3. ROUTE SELECTION FOR DASH OVER SDN

In this chapter, the proposed route selection methods for improving the QoE of the DASH clients running over SDN are presented. The rest of the chapter is organized as follows: First, background information about DASH standard is given. Second, the related works about DASH over SDN is summarized. Third, an optimization model is proposed to improve the QoE of the clients by increasing the average received bitrate and to provide bitrate fairness among clients. Fourth, a service differentiated optimization framework is proposed for increasing the QoE of DASH clients and for providing service classes in an SDN domain. Fifth, a segment based route selection method by considering the bitrate of the requested segments is proposed for DASH clients running over SDN. And for last, based on the segment based route selection method, an algorithm for estimating the current network load by considering the bitrates of video representations and an algorithm for determining the streaming paths for each client in the system are proposed.

3.1 Dynamic Adaptive Video Streaming Over HTTP

In the recent years, Hyper-Text Transfer Protocol (HTTP) adaptive streaming has been the most prevalent approach preferred in Internet video streaming applications. Youtube, Microsoft Smooth Streaming, Netflix, Apple HTTP Live Streaming can be cited as examples for such applications. The main reasons of this tendency are based on HTTP advantages such as usage of existing caching infrastructure and firewall traversal. Moving Picture Experts Group (MPEG) has standardized Dynamic Adaptive Streaming over HTTP (DASH) as the only available international standard for HTTP adaptive streaming (Sodagar, 2011).

Fig.3.1 illustrates the general overview of the DASH standard. In DASH, video files are encoded at various bitrates to construct different representations of the same video in a server. Each representation is further divided into smaller units called segments. At the server side, there is a Media Presentation Description (MPD) file which includes the segment information such as bitrate and URL. Clients adapt the video quality by requesting segments of different representations over time after connecting the server and retrieving the MPD file. As a client driven architecture, the decisions determining the request time and the quality of the representation is based on the quality adaptation logic of the clients. Since the rate adaptation algorithm running at the client side is out of the scope of DASH standard, developing a rate adaptation algorithm has aroused great attention of both industry and academia. The proposed rate adaptation algorithms may take into account the estimated band-

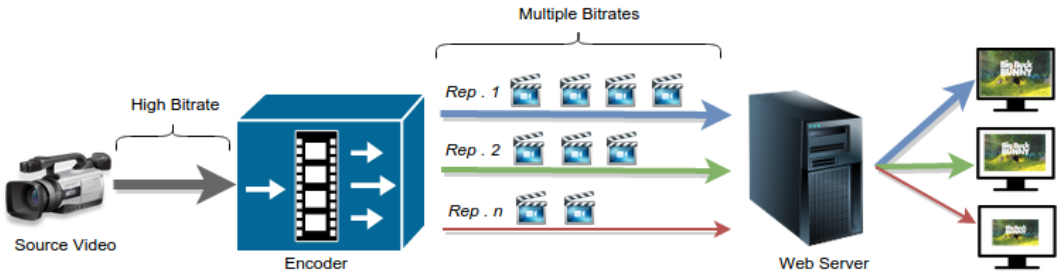


Figure 3.1. Overview of the DASH

Table 3.1. The average bitrate of the representations in 720p video.

Representation Id	Average Bitrate (Kbps)
1	790
2	1030
3	1240
4	1540

width of the paths between the client and the server (Rainer et al., 2012), (Li et al., 2014), (Jiang et al., 2014), buffer status (Huang et al., 2014) or both (Cicco et al., 2013) in order to decide the representation to be requested. The adaptation logic of the DASH client software used during the simulations of the proposed studies in this chapter is based on throughput measurements as in (Rainer et al., 2012).

In the simulations, two different DASH video data with resolutions of 1280x720 (720p) and 1920x1080 (1080p) are served by the DASH server (Lederer et al., 2012). The 720p video has four different representations while 1080p has six different representations. These representations are splitted into 300 segments, each with a duration of 2 seconds. The average bitrate of the representations in both video data are given in Table 3.1 and Table 3.2.

Table 3.2. The average bitrate of the representations in 1080p video.

Representation Id	Average Bitrate (Kbps)
1	2200
2	2500
3	3100
4	3600
5	3900
6	4220

3.2 Related Work

The flexibility on determining the routing algorithms provided by SDN leads researchers to propose video streaming application specific routing strategies in the literature. In (McDonagh et al., 2013), an OpenFlow (McKeown et al., 2008) controller monitors the Real-time Transport Protocol (RTP) traffic on the video streaming path and if it detects RTP loss, it sends the commands to the switches to switch the alternative streaming path. A streaming path selection strategy based on Dijkstra algorithm by considering available bandwidth and video data rate is proposed in (Karl et al., 2013). Similar to (Karl et al., 2013), the authors in (Valdivieso Caraguay et al., 2015) propose Dijkstra algorithm to determine multimedia streaming paths and give a framework for providing Quality of Service (QoS) by using the parameters of bandwidth and loss rate of the links. In (Uzakgider et al., 2015), a routing strategy for scalable video streams based on a learning model is proposed. In addition to that study, several video streaming systems are presented in the literature (Cetinkaya et al., 2015b; Egilmez et al., 2011, 2013; Laga et al., 2014) proposing various flow routing strategies to define different streaming paths for scalable video layers. These studies mainly focus on User Datagram Protocol (UDP) video streaming over SDN. However, designing new routing algorithms for transferring the video packets of an adaptive HTTP streaming application requires considering DASH specific parameters such as representation bitrates and TCP behavior as well as available bandwidth information.

In (Seddiki et al., 2014), 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 (Seddiki et al., 2014). An optimization model providing service differentiation by determining the queue allocations among the paths between the server and the clients in an SDN domain is proposed in (Bagci et al., 2016). In (Petrangeli et al., 2015), when a DASH client requests for a segment, the controller checks the buffer fullness of the client, the duration and the size of the segment and sends a prioritization command to the switches if necessary. Although the system proposed in (Petrangeli et al., 2015) uses the information related to the segment bitrate, no path selection strategy is defined in this study. In (Georgopoulos et al., 2013), DASH clients select the representation according to the commands received from the controller. The controller decides to the representations by considering fairness and device characteristics of the users (Georgopoulos et al., 2013). In (Nam et al., 2014), 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. In (Quinlan et al., 2015), authors show that if streaming flows are limited such that the total bandwidth is distributed evenly among clients, then perceived video quality is balanced compared to the case of no limitation. Leveraging SDN technology to monitor QoE achieved by the clients is proposed in (Farshad et al., 2015). The authors propose an SDN based DASH architecture in which QoE related parameters of each client are collected and network resources are allocated by an external device communicating SDN controller in (Bentaleb et al., 2016). In (Kleinrouweler et al., 2016), SDN controller allocates the bandwidth of the bottleneck link for DASH client flows. Similar to (Kleinrouweler et al., 2016), bandwidth allocation for HAS client flows is proposed in (Ramakrishnan et al., 2015). SDN controller allocates bandwidth by using the QoE related information received from the clients. In (Mu et al., 2016), SDN based network resource allocation method for providing fairness among DASH clients is proposed. In (Egilmez et al., 2012), streaming paths for adaptive HTTP streaming clients are determined by using constrained shortest path optimization taking congestion, delay and jitter parameters into consideration in an SDN domain. Various path selection strategies including round robin path selection, path selection based on deep packet inspection, path selection according to the level of loads are proposed to increase QoE achieved by HAS clients in (Jarschel et al., 2013).

3.3 Route Optimization With Fairness For DASH Over SDN

In this section, a multiobjective optimization model is proposed for the selection of the paths between the DASH clients and the server (Cetinkaya et al., 2015a). The aim of the optimization model is to improve the QoE of the clients by increasing the average received bitrate and to provide fairness among clients by decreasing the difference between the received bitrates of the clients.

3.3.1 Proposed Work

Let P denotes the set of paths between the clients and the server and H denotes the set of online clients in the session. The optimization model maximizes the available bandwidths of the paths assigned to the clients (Equation 3.1) in order to improve the received bitrate of the clients and minimizes the standard deviation of the available bandwidths of the assigned paths (Equation 3.2) in order to provide fairness among clients with respect to the received bitrate. The optimization model

is as follows:

$$\max \sum_{i=1}^{|H|} abw_i \tag{3.1}$$

$$\min \sigma_{abw_i}, i \in H \tag{3.2}$$

subject to,

$$\sum_{i=1}^{|H|} abw_i \leq C_{total} - tr_{other} \tag{3.3}$$

In Equation 3.1, abw_i denotes the available bandwidth of the path assigned to the i th client. In Equation 3.3 C_{total} is the total capacity of the paths in the network where tr_{other} is the amount of traffic except the DASH video packets. SDN controller runs the optimization model when a new client joins the system and assigns the streaming path between the video server and the all clients in the system as an output of the model.

3.3.2 Simulation Study

The network topology used in the simulation study is shown in Fig.3.2. In the network, there are four fully-connected switches that produces five paths between clients and the server. The capacity of the paths are limited during the simulations. There are four DASH clients that join the system with 30 seconds of interval. The video served by the DASH server is 720p video which details are given in Table 3.1. Each representation is divided into 300 segments with a length of 2 seconds. Thus, the conducted simulations last for 600 seconds. To evaluate the performance of the proposed work, best-effort routing method is also implemented on the SDN controller. The proposed work and the best-effort routing methods are tested under two different set of path capacities. In *Setting1*, the capacity of the paths are [1000, 4000, 4000, 1000, 1000] Kbps, where the capacity of the paths in *Setting2* are [3000, 7000, 3000, 3000, 3000] Kbps. The capacity of the shortest-path is 4000 Kbps in *Setting1* and 3000 Kbps in *Setting2*. The experiments are repeated 10 times and the received bitrate, outage durations and startup delay values of the clients are measured. The average received bitrate shows the quality of the received video. The outage duration refers to the freezes that client experiences due to the rebuffering. The startup delay is the amount of time that clients wait the video to playback after the requesting

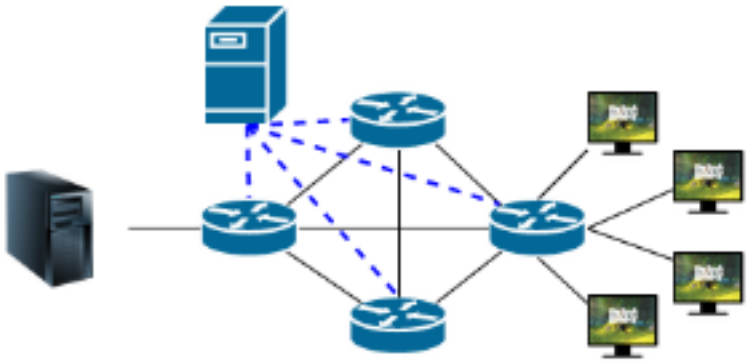


Figure 3.2. Network topology for the simulations.

it. In the simulations clients start to playback the video after receiving 15 segments.

Fig.3.3 and Fig.3.4 shows the time-dependent average received bitrate of the clients in the proposed work and best-effort routing method under *Setting1* and *Setting2*, respectively. Since the clients join the system with 30 seconds of interval, the measurement in the graphs are given from 90th seconds when the last client joins the system. It is seen from Fig.3.3 and Fig.3.4 that the average received bitrate of the clients in proposed work is much more higher than the received bitrates of the clients in the best-effort routing method in both settings. Also, in the proposed work, the average received bitrate of the clients are higher than 1540 Kbps which is the average video bitrate of the highest quality representation. Thus, the clients in the proposed work playback the highest quality video and keep the occupancy of video buffers high. Keeping buffer occupancy rates high ensures client to playback the video without experiencing freezes even if there is not enough capacity in the system due to the sudden decreases in bandwidth of the paths.

Table 3.3 shows the outages in seconds when there are no video packets to play in the clients' buffers. In the proposed method, the client does not experience any outages in both *Setting1* and *Setting2*. In best-effort routing method, the outages are observed in *Setting2*. In Table 3.4 the average startup delay of the clients are given. As seen from the Table 3.4, poor network capacity also delays to transmit the packets necessary to start playing the video, thus causing the startup delay to increase.

One of the objectives of the proposed optimization model is to provide fairness among the clients by reducing the standard deviation of the assigned paths' available bandwidths. In order to evaluate the fairness of the proposed work, the Jain fairness

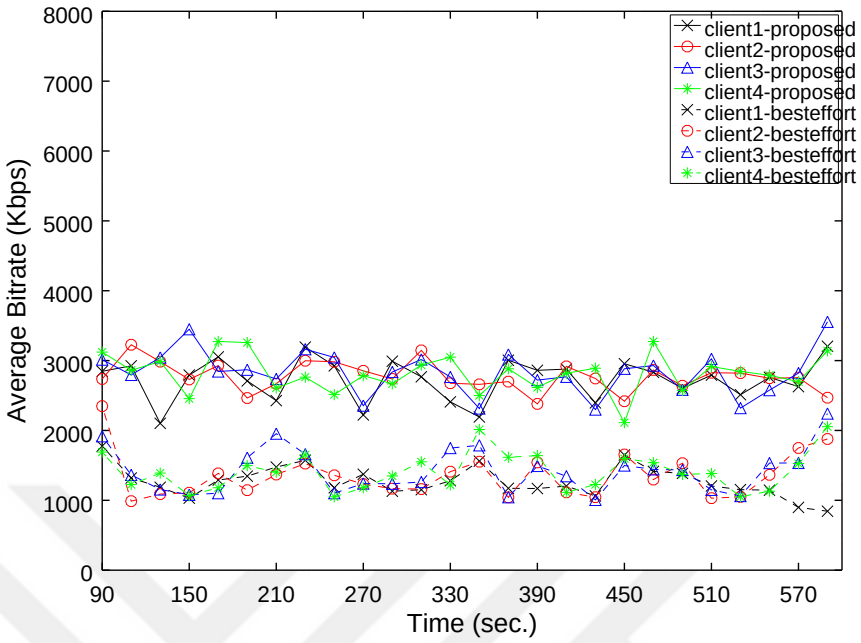


Figure 3.3. Average received bitrate of the clients under *Setting1*.

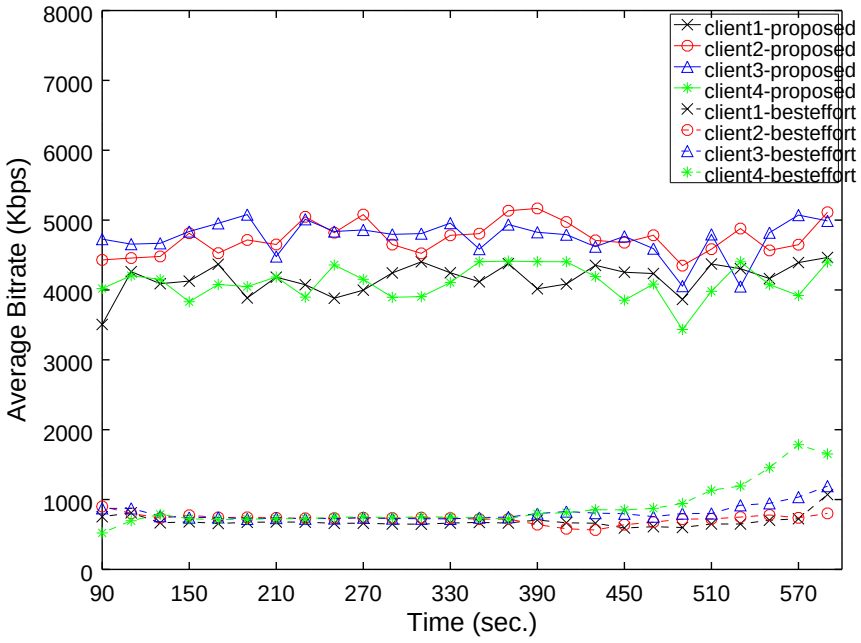


Figure 3.4. Average received bitrate of the clients under *Setting2*.

Table 3.3. Outage duration

Outage Duration (sec)	*Setting1*			*Setting2*		
	Min	Max	Avg	Min	Max	Avg
Proposed work	0	0	0	0	0	0
Best-effort routing	0	0	0	1	2	13.9

Table 3.4. Startup delay

Startup delay (sec)	*Setting1*	*Setting2*
Proposed work	13.5	14.2
Best-effort routing	17.3	22.2

index (Jain et al., 1984) is calculated for both *Setting1* and *Setting2*. In *Setting1*, the fairness index is 1 since clients are assigned to paths with same bandwidth as a result of the proposed optimization model. The fairness index is measured as 0.998 for *Setting2*. These results show that the fairness is achieved in proposed work.

In this study, an optimization model is proposed for the selection of video streaming paths between DASH clients and server running on an SDN domain. The aim of the optimization model is to increase the QoE of the clients while providing fairness among clients. The proposed work also aims to improve the system-wide performance by dynamically changing the paths between the server and the client. The simulation results show that the proposed work outperforms the Internet's best-effort routing method in terms of QoE.

3.4 A Service Differentiation Architecture For DASH Over SDN

In this section, a service differentiated architecture is proposed for DASH clients residing within an SDN domain. In the proposed work, different service classes are defined and congestion-related problems are solved in order to provide an increase in QoE according to the users' service classes. In order to take traffic characteristics specific to the DASH applications into account, the average bitrate of the video files stored in the server and current available bandwidth of the links between the server and the clients are considered. Providing service differentiated classes requires providing the same service level among the users within the same class. In addition to that, 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:

- An architecture providing service differentiation to DASH clients is proposed. This is the first study that proposes an SDN based fair service differentiation architecture considering the characteristics of DASH applications.
- An optimization scheme based on Mixed Integer Programming (MIP) is designed 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, a decomposition based heuristic algorithm is developed to reduce the computational complexity of the MIP based optimization scheme.
- The performance of the proposed architecture is evaluated 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.

3.4.1 The Proposed Work

In the proposed work, SDN controller determines the streaming paths between the DASH clients and server, controller then 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 (triggered by *HostManager*)
- When congestion occurs on the streaming paths between the clients and server (triggered by *LinkManager*)

In both cases, the controller runs the path assignment procedure given in the next subsection.

3.4.1.1 MIP Model

In this subsection, the MIP framework is presented. 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 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 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. (3.4).

$$0 \leq f_{ij}^k \leq C_{mx} \; \forall (i,j) \in A \; \forall k \in U \tag{3.4}$$

The objective function can be expressed as:

$$\max \left[\gamma_1 \sum_{i \in U_P} s_i - \gamma_2 \sum_{i \in U_P} e_i + \gamma_3 \sum_{i \in U_S} s_i \right]. \tag{3.5}$$

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. The constraints for this optimization problem are defined after this point.

We define binary variables a_{ij}^k as flow indicators in Eq. (3.6). 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. (3.8) 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. (3.7) forces $a_{ij}^k = 1$.

$$a_{ij}^k \in \{0,1\}\ \forall (i,j) \in A\ \forall k \in U \tag{3.6}$$

$$f_{ij}^k \leq C_{ij} \times a_{ij}^k\ \forall k \in U\ \forall (i,j) \in A \tag{3.7}$$

$$f_{ij}^k \geq \epsilon \times a_{ij}^k\ \forall k \in U\ \forall (i,j) \in A \tag{3.8}$$

Binary variables x_{ij} given in Eq. (3.9) are utilized in Eq. (3.10) and (3.11) 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. (3.10), however if all $a_{ij}^k = 0$ then $x_{ij} \geq 0$ (by Eq. (3.10)) and $x_{ij} \leq 0$ (by Eq. (3.11)) will force $x_{ij} = 0$.

$$x_{ij} \in \{0,1\}\ \forall (i,j) \in A \tag{3.9}$$

$$x_{ij} \geq a_{ij}^k\ \forall k \in U\ \forall (i,j) \in A \tag{3.10}$$

$$x_{ij} \leq \sum_{k \in U} a_{ij}^k\ \forall (i,j) \in A \tag{3.11}$$

We need to ensure that all end-to-end paths are single path routes. Eq. (3.12) guarantees that only one link going out of the host switch (H) is used for the flow of user-k. Eq. (3.13) 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. (3.14).

$$\sum_{j \in N} a_{Hj}^k = 1\ \forall k \in U \tag{3.12}$$

$$\sum_{j \in N, j \neq i} a_{ij}^k \leq 1\ \forall k \in U,\ \forall i \in N \tag{3.13}$$

$$0 \leq s_i \leq C_{mx}\ \forall i \in U \tag{3.14}$$

Eqs. (3.15), (3.16), and (3.17), jointly guarantee the flow conservation at switches. Flow conservation constraint of host switch is given in Eq. (3.15) 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. (3.16) is used to ensure that the entire flow of user-k terminates at the server (S) node. Eq. (3.17) is used to perform flow balancing at

intermediate switches (M).

$$\sum_{j \in N, j \neq H} f_{Hj}^k = s_k \; \forall k \in U \tag{3.15}$$

$$s_k = \sum_{j \in N, j \neq S} f_{jS}^k \; \forall k \in U \tag{3.16}$$

$$\sum_{i \in N, i \neq j} f_{ij}^k = \sum_{l \in N, l \neq j} f_{jl}^k \; \forall k \in U \; \forall j \in M \tag{3.17}$$

To avoid any loop backs to the host, we zero all incoming flows to host node (from the switches) as

$$\sum_{i \in N-H} f_{iH}^k = 0 \; \forall k \in U. \tag{3.18}$$

To avoid any loop backs to the server, we zero all outgoing flows from server node (back to the switches) as

$$\sum_{i \in N-S} f_{Si}^k = 0 \; \forall k \in U. \tag{3.19}$$

We define the slack flow (g_{ij}^k) that represents the amount of allocated but unutilized capacity for user-k over link-(i, j). Variable bound for g_{ij}^k is the same as that of f_{ij}^k and given in Eq. (3.20). Eqs. (3.21) and (3.22) 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. (3.23).

$$0 \leq g_{ij}^k \leq C_{mx} \; \forall (i, j) \in A \; \forall k \in U, \tag{3.20}$$

$$\sum_{k \in U} \{f_{ij}^k + g_{ij}^k\} \geq x_{ij} \times C_{ij} \; \forall (i, j) \in A \tag{3.21}$$

$$\sum_{k \in U} \{f_{ij}^k + g_{ij}^k\} \leq C_{ij} \; \forall (i, j) \in A \tag{3.22}$$

$$g_{ij}^k \leq a_{ij}^k \times C_{ij} \; \forall (i, j) \in A \; \forall k \in U \tag{3.23}$$

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. (3.24) which is equal to zero if $g_{ij}^k = 0$ and equal to one if $g_{ij}^k > 0$ as given in Eqs. (3.25) and (3.26). Eq. (3.27) 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).

$$w_{ij}^k \in \{0,1\}\ \forall (i,j) \in A\ \forall k \in U \tag{3.24}$$

$$g_{ij}^k \leq w_{ij}^k \times C_{ij}\ \forall k \in U\ \forall (i,j) \in A \tag{3.25}$$

$$g_{ij}^k \geq \epsilon \times w_{ij}^k\ \forall k \in U\ \forall (i,j) \in A \tag{3.26}$$

$$\sum_{(i,j)\in A} w_{ij}^k \leq \sum_{(i,j)\in A} a_{ij}^k - 1\ \forall k \in U \tag{3.27}$$

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. (3.28) 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. (3.28) 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.*, $f_{58}^2 + g_{58}^2 = f_{58}^4 + g_{58}^4$).

$$f_{ij}^k + g_{ij}^k \leq f_{ij}^h + g_{ij}^h + C_{ij}(2 - a_{ij}^k - a_{ij}^h)\ \forall (i,j) \in A\ \forall k,h \in U \tag{3.28}$$

Eq. (3.29) guarantees that data flow for premium users is larger than standard users. Furthermore, Eq. (3.30) and (3.31) 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 $\epsilon = \zeta_m$.

$$s_k > s_h\ \forall k \in U_P\ \forall h \in U_S \tag{3.29}$$

$$s_k \geq \zeta_g\ \forall k \in U_P \tag{3.30}$$

$$s_h \geq \zeta_m\ \forall h \in U_S \tag{3.31}$$

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-4 and node-5 sending each other the same amount of data or 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. (3.32) 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. (3.33) and (3.34). 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. (3.35).

$$p_i^k \in \{0,1\}\ \forall k \in U\ \forall i \in N \tag{3.32}$$

$$p_H^k = 1\ \forall k \in U \tag{3.33}$$

$$p_S^k = 1\ \forall k \in U \tag{3.34}$$

$$p_i^k = \sum_{j \in N, i \neq j} a_{ji}^k\ \forall k \in U\ \forall i \in M \tag{3.35}$$

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. (3.36). 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. (3.37) 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. (3.38).

$$0 \leq r_i^k \leq |N|\ \forall k \in U\ \forall i \in N \tag{3.36}$$

$$r_H^k = 1\ \forall k \in U \tag{3.37}$$

$$r_S^k = 1 + \sum_{(i,j) \in A} a_{ij}^k\ \forall k \in U \tag{3.38}$$

We set the degrees of switches which are not relaying the data of user-k by using Eq. (3.39) 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. (3.40) and (3.41) 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. (3.40) and (3.41) 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. (3.40) and (3.41).

$$r_i^k \leq |N| p_i^k\ \forall k \in U\ \forall i \in M \tag{3.39}$$

$$r_i^k \geq r_j^k + a_{ji}^k + |N|(a_{ji}^k - 1)\ \forall k \in U\ \forall j \in N\ \forall i \in N - j \tag{3.40}$$

$$r_i^k \leq r_j^k + a_{ji}^k + |N|(1 - a_{ji}^k)\ \forall k \in U\ \forall j \in N\ \forall i \in N - j \tag{3.41}$$

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 (Quinlan et al., 2015). For example in (Mu et al., 2016), 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 (Shi et al., 2014). 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. (3.42). Note that $-C_{mx} \leq d_i \leq C_{mx}$ as stated in Eq. (3.43) which is a direct result of variable bound on s_k in Eq. (3.14).

$$d_i = s_i - \frac{1}{|U_P|} \sum_{k \in U_P} s_k \; \forall i \in U_P \tag{3.42}$$

$$-C_{mx} \leq d_i \leq C_{mx} \; \forall i \in U \tag{3.43}$$

Since we need to find the absolute values of d_i's (*i.e.*, $e_i = |d_i|$) which is achieved by utilizing Eq. (3.46)–(3.49). We utilize binary variables b_i given in Eq. (3.46) for obtaining the absolute values of d_i's. The variable bound on e_i's are given in Eq. (3.45). When d_i is positive, Eq. (3.46) and (3.49) always hold, however, for Eq. (3.47) to hold binary indicator variable b_i must be set to zero, hence, the effective constraints are Eq. (3.46) and (3.48) 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. (3.46) to hold, $b_i = 1$ which makes Eq. (3.47) and (3.49) the effective constraints (*i.e.*, $-d_i \leq e_i$ and $-d_i \geq e_i$) resulting in $-d_i = e_i$.

$$b_i \in \{0, 1\} \; \forall i \in U \tag{3.44}$$

$$0 \leq e_i \leq C_{mx} \; \forall i \in U \tag{3.45}$$

$$d_i + 2 \times C_{mx} \times b_i \geq e_i \; \forall i \in U_P \tag{3.46}$$

$$-d_i + 2 \times C_{mx} \times (1 - b_i) \geq e_i \; \forall i \in U_P \tag{3.47}$$

$$d_i \leq e_i \; \forall i \in U_P \tag{3.48}$$

$$-d_i \leq e_i \; \forall i \in U_P \tag{3.49}$$

3.4.1.2 Heuristic Algorithm

Algorithmic complexity of our MIP model is in the NP-complete 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 (Baldi et al., 2012). 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 has much lower computational complexity. We present an analysis of the MIP model and the heuristic algorithm in next subsection to compare the computational complexities and performances of both approaches. The pseudo-code of the heuristic is outlined in Fig. 3.5. The most important factor increasing 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. Note that we still use the MIP model in the heuristic algorithm (in fact multiple times sequentially), however, the problem instances we utilize the MIP model use very low number of variables and constraints in comparison to the original MIP model. Furthermore, 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 $C_{3,2}^0 = 5000$ 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

Algorithm 1 The algorithm to determine the paths for each user in a given network.

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.

1: Define $C^1 = C^0$.

Determining paths for standard users :

2: Assume that there are no premium users ($U_P \in \varnothing$).
3: 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.
4: **while** max-flow(C^1)$\geq \zeta_m$ **do**
5: 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: $f_{ij}^k + g_{ij}^k \leq \frac{C_{ij}^0}{\eta_{ij}^1} \; \forall (i,j) \in A, \forall k \in U_S$.
6: Mark the links in P_l and increment η_{ij}^1 by 1.
7: Update capacity values in C^1 by subtracting the flows in the links of P_l.
8: Include P_l to $\overline{\mathbf{S}}$, which contains all possible paths determined so far.
9: $l++$
10: **end while**

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

11: 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 $\overline{\mathbf{S}}$.
12: 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 :

13: Assume that there are no standard users ($U_S \in \varnothing$).
14: Assume that $\eta_{ij}^2 = \eta_{ij}^1$ and $m = 1$ (path index).
15: **while** max-flow(C^2)$\geq \zeta_g$ **do**
16: 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: $f_{ij}^k + g_{ij}^k \leq \frac{C_{ij}^0}{\eta_{ij}^2} \; \forall (i,j) \in A, \forall k \in U_P$.
17: Mark the links in P_m and increment η_{ij}^2 by 1.
18: Update capacity values in C^2 by using the bandwidth value of the P_m.
19: Include P_m to $\overline{\mathbf{P}}$, which contains all possible paths determined.
20: $m++$
21: **end while**

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

22: 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 $\overline{\mathbf{P}}$.
23: 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 $\hat{\mathbf{P}}$.

Figure 3.5. Heuristic algorithm

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.

3.4.2 Simulation Study

General Algebraic Modeling System (GAMS)[1] 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.

The video served by the DASH server is 720p video which details are given in Table 3.1. 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[2]. 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.

3.4.2.1 Comparison of MIP model and Heuristic Algorithm

In Table 3.5, 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. It takes prohibitively large amounts of time when we attempt to solve the exact optimiza-

[1] http://www.gams.com/
[2] http://www.topology-zoo.org/

tion 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 which increases the number of variables and constraints drastically. The difference in total bandwidth assignment of the MIP model and the heuristic algorithm can be as low as 0.32% (for 14 users) and can be as high as 8.32% (for 16 users). Since the heuristic algorithm cannot assign the premium user bandwidths optimally, in some scenarios heuristic algorithms's bandwidth allocation to standard users are higher than the MIP model's assignment of bandwidth to standard users.

Table 3.5. 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 3.6, 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. Note that solution times given in Table 3.6 are obtained by implementing SDN controller software on a personal computer. It is clear that solution times tend to decrease when SDN controller runs on a higher-capacity server (i.e. with multi-core processors and with higher memory capacity).

3.4.2.2 Simulation Results

As mentioned in previous section, the proposed architecture aims to provide QoE 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,

Table 3.6. 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

k-shortest path are selected among the paths 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 (Egilmez et al., 2013), 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 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 QoE 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 the outcome of each test.

In Figure 3.6, 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.*, Figure 3.6a and 3.6b) 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 QoE. In Figure 3.7, the number of quality changes observed in the clients are given for all approaches. The decrease in the number of quality changes observed with the proposed approach is up to 39% 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 next higher representation.

The percentage of received segments from each type of representation is given in Figure 3.8. Especially for premium users, the video quality perceived by the users 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 3.7, average, max and min durations in seconds observed in a client according to the user types are given. In the table, avg refers to the average of outage durations observed in same type of user while max and min refers to the maximum and minimum outage duration observed in the clients during the streaming session, respectively. The results are given only for the k-shortest path approach due to the fact that we did not observe any outages in the proposed architecture.

Table 3.7. Outages observed in Dfn topology (s)

# of clients	Proposed-max	Proposed-fair	CSP-QoS	k-shortest
10	0.2	0.1	0	75.4
20	0	0	3.4	78
30	0	0	4.4	10

Table 3.8. Standard Deviation Values Observed in the Proposed Architecture

	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

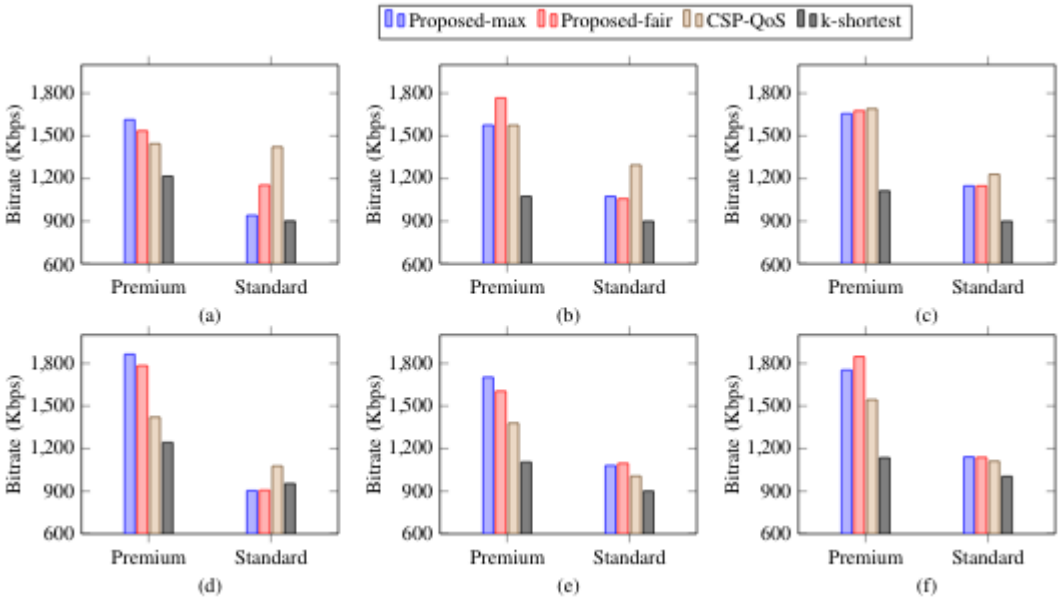


Figure 3.6. Average received bitrate - (a) Compuserve - 10 clients, (b) Compuserve - 20 clients, (c) Compuserve - 30 clients, (d) Dfn - 10 clients, (e) Dfn - 20 clients, (f) Dfn - 30 clients.

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 3.8, 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).

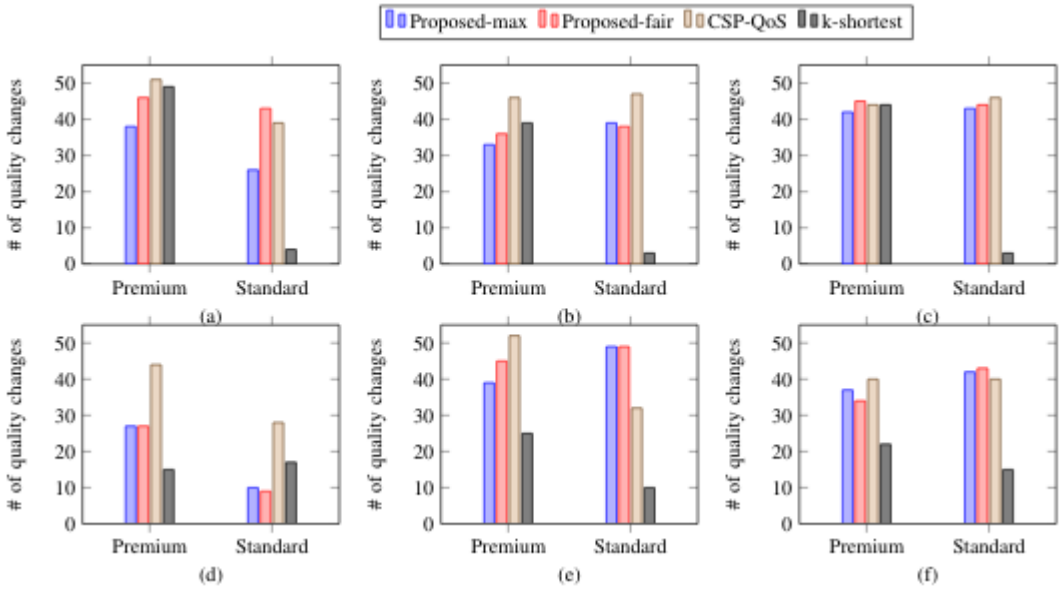


Figure 3.7. Number of quality changes - (a) Compuserve - 10 clients, (b) Compuserve - 20 clients, (c) Compuserve - 30 clients, (d) Dfn - 10 clients, (e) Dfn - 20 clients, (f) Dfn - 30 clients.

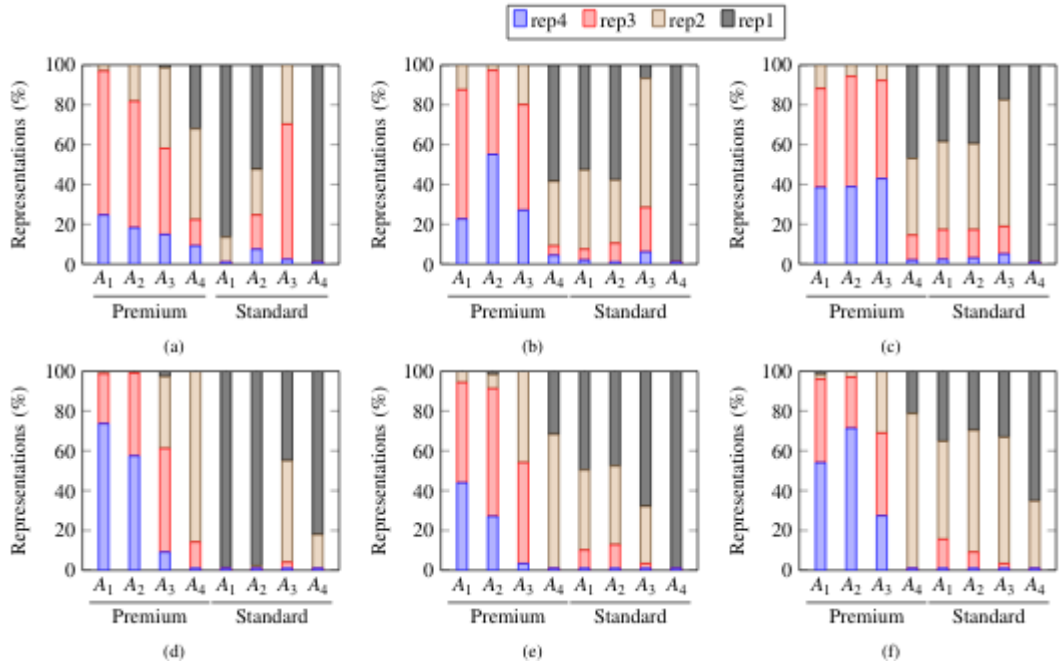


Figure 3.8. Percentage of the received representations - (a) Compuserve - 10 clients, (b) Compuserve - 20 clients, (c) Compuserve - 30 clients, (d) Dfn - 10 clients, (e) Dfn - 20 clients, (f) Dfn - 30 clients.

In this study, we proposed an architecture for increasing QoE of DASH clients and for providing service classes in an SDN domain. The controller dynamically changes the streaming paths between the server and the clients by taking the underlying network conditions into account. Streaming paths are determined according to the optimization framework. Although the optimization framework is given for two different service classes, it can be enhanced for more service classes.

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. The simulation results show that the proposed architecture, as well as providing fairness, provided increase in QoE in terms of received bitrate, achieved quality, outage durations and number of quality changes when compared to other service differentiation approaches. We observed 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.

3.5 Segment-Based Route Optimization For DASH Over SDN

In this section, a route optimization solution is provided to the problem of what is the best route between the server and a client under the changing networks conditions and variable bitrate of the segments (Cetinkaya et al., 2014). In the proposed work, the paths between the DASH client and the server are selected by considering available network throughput, bitrate of the segment and path length. Each time the client requests a segment, the path is changed between the server and the client. Hence, flows are rerouted for each segment during the streaming session in order to maximize overall throughput.

3.5.1 Proposed Work

In our study, the route selection of the flows is done by considering the paths from the server to the clients. The proposed system is illustrated in Fig. 3.9. As shown in the figure, each time the client requests a segment (1), the server informs the controller about the client and the requested video segment (2). The controller selects a path considering the bitrate of the requested segment and network load (3). After selecting an optimum path for the requested segment, the controller sends flow information to the corresponding switches along the selected path via *FlowMod* message which is defined in OpenFlow protocol (4). Finally, the server starts to send the requested video segment over the defined path (5).

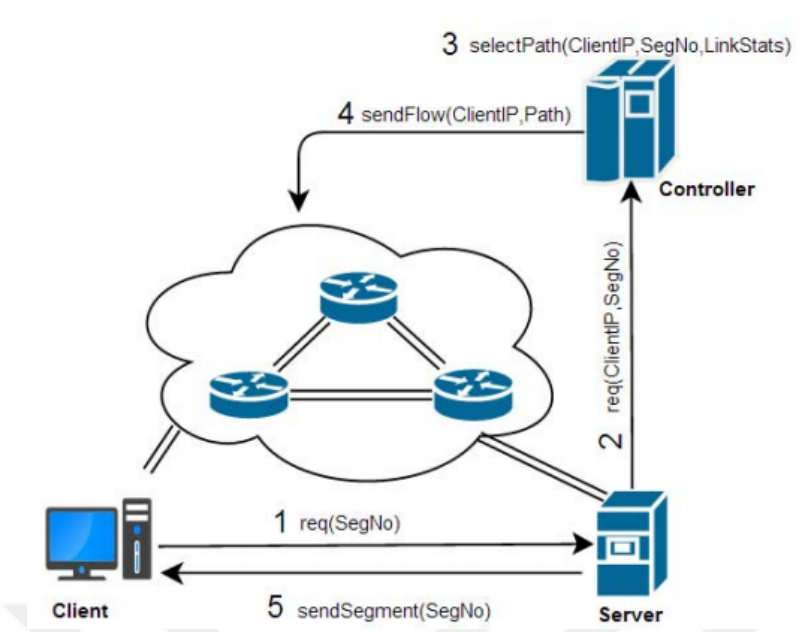


Figure 3.9. Steps of segment based rerouting process.

In the third step of Fig. 3.9, the controller determines all possible paths between the client and the server to select a path by using multiobjective optimization. The aim of the optimization is selecting the path having adequate capacity to send the segment at that moment while minimizing the path length. In order to determine the path, the controller needs to calculate the available bandwidth of the paths. For this purpose, the controller queries the switches periodically via sending PortStats messages which is defined in OpenFlow protocol to obtain information about current load on the links. Since the bitrate of the DASH segments are variable, the load on a link changes dynamically. Instant load values are averaged by using the smoothing formula given in Equation (3.50). In Equation (3.50), LD_{ij}^{t} is the current load, α is the smoothing factor and LD_{ij} is the averaged load of the link i on the jth path. α is set to 0.6, hence we give slightly more importance to the current load.

$$LD_{ij} = \alpha \times LD_{ij}^{t} + (1-) \times LD_{ij} \tag{3.50}$$

After determining the load of each link in a path, the available bandwidths of all paths are calculated by measuring the bottleneck link bandwidth as shown in Equation (3.51). In the formula, M_j is the number of links, C_{ij} is the capacity of the link i. Note that the bottleneck link bandwidth gives the available bandwidth of the path.

$$abw_j = \min(C_{ij} - LD_{ij}), 1 \leq i \leq M_j \tag{3.51}$$

Nevertheless, the HTTP streaming throughput does not achieve the maximum

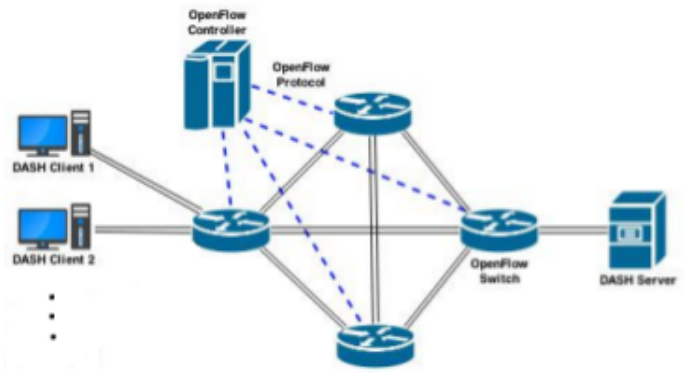


Figure 3.10. Network topology used during the experiments.

available link bandwidth due to the behavior of the competing TCP flows. Hence, we integrate a factor, called beta factor, for adding some buffer for the measured available bandwidth value. The optimization model is,

$$\min_{(p_1,\ldots,p_j)}^{(bw)} = \min_{(p_1,\ldots,p_j)} \left\{ \min_{(1\leq i\leq M_j)} \left\{ (C_{ij} - LD_{ij}) \times \beta - s_{br}^{t} \right\} \right\} \tag{3.52}$$

$$\min_{(p_1,\ldots,p_j)}^{(length)} = \min \{M_j\} \tag{3.53}$$

subject to,

$$bw \geq 0 \tag{3.54}$$

where β represents beta factor and s_{br}^{t} represents the bitrate of the current segment. In the next section, we observe the performance of the optimization model by using different values of β factor.

3.5.2 Simulation Study

We used the topology as shown in Fig. 3.10, with one DASH server and five DASH clients. In the figure, the dotted line represents the OpenFlow protocol. The capacity of all links equals to 4 Mbps. The video served by the DASH server is 720p video which details are given in Table 3.1.

The proposed model is tested on the given network topology and its performance is compared to that of Internet's best effort (shortest path) service. We measured the bitrate of the requested segment, which shows the quality of the played video, outage duration and startup delay. Outage duration refers to display freeze

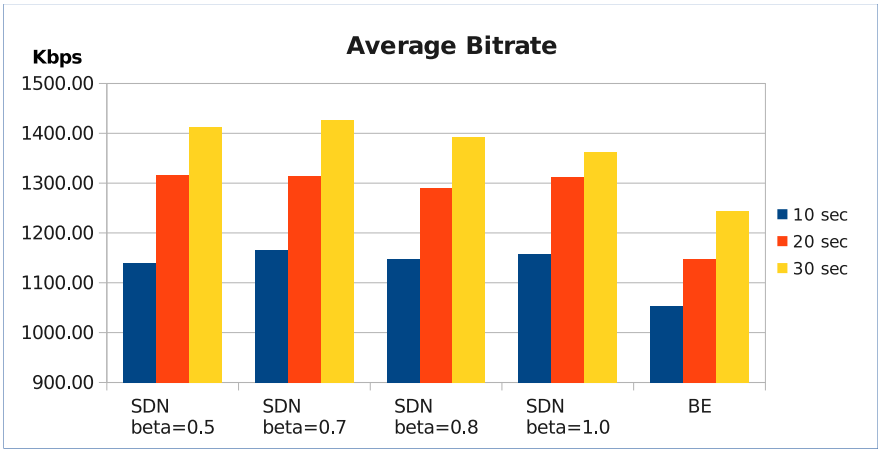


Figure 3.11. Average bitrate values of requested segments.

in case a client needs to rebuffer and it is measured in seconds. Startup delay is the time passed until the video starts to play after the client sends its first request to the server. Hence, long pre-buffering time increases the startup delay. The clients start to play the video after receiving 15 segments. The simulations are repeated 5 times and the results are averaged. In the simulations, the clients join the system with 10, 20 and 30 seconds of intervals.

The average bitrate values according to these intervals are given in Fig. 3.11 According to Fig. 3.11, the results show that the proposed system provides up to 180 Kbps increase in bitrate when compared to best-effort routing. The best performance is obtained when $\beta = 0.7$ and 30 seconds of interval. Since the duration of competing flows is shorter than the case of 10 seconds and 20 seconds of interval, 30 seconds of interval gives best results in all simulations.

Averaged outage duration values are given in Fig. 3.12. Minimum outage duration values are observed when $\beta = 0.7$. This shows that when $\beta = 0.7$, the bitrate of the requested segment is compatible with underlying network capacity. On the other hand, if some buffer value is not added to the measured available bandwidth, i.e. beta factor equals to 1.0, the outage duration is high when the number of TCP competing flows is increased. When traditional best effort routing is used, even if the number of competing flows is not high, i.e. where the clients join the system with 30 seconds of interval, outage duration equals to 14.60 seconds on average. This is due to the client rate control algorithm does not adapt fast enough to the changing network conditions and the requested bitrates are too high in some situations. The best-effort performance can be improved if the more sophisticated rate adaptation algorithms are implemented.

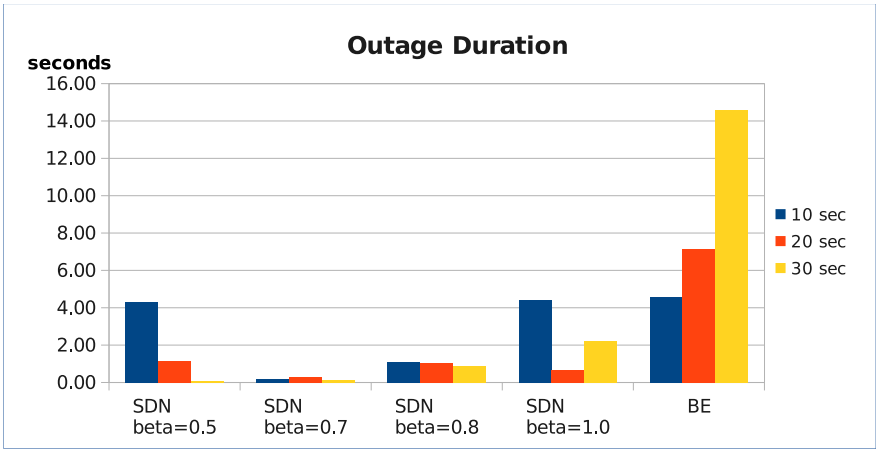


Figure 3.12. Outage duration values.

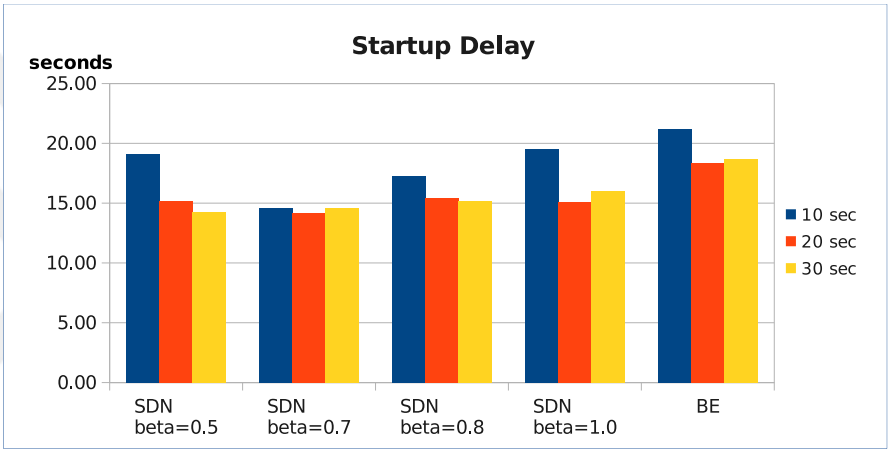


Figure 3.13. Startup delay values.

In Fig. 3.13, averaged startup delay values are given. If the pre-buffering time is high, clients experience long startup delay. Pre-buffering time increases when the bitrate of the requested segment exceeds the available bandwidth. Hence, if the bitrate of the requested segment is not compatible with underlying network capacity, both outage duration and startup delay values are increased. As seen in the graph given in Fig. 3.13, minimum startup delays are observed when $\beta = 0.7$. As expected, startup delays are increased when the clients join the system with 10 seconds of interval, since the arrival of video packets are delayed due to the competing flows.

In this study, a novel segment based path routing method for DASH clients running over SDN network is proposed. The proposed routing algorithm outperforms traditional best-effort routing by optimizing route selection for each requested segment. According to the bitrate of the current segment, packet flows are determined by the controller according to an optimization model. The proposed optimization model aims to find the most suitable path by considering the available

bandwidth, bitrate of the current segment, competing flows and the path length. We also observe required buffer ratio for the measured available bandwidth to provide seamless streaming.

3.6 Segment-Aware Dynamic Routing For DASH Over SDN

In this study, we propose to consider the bitrates of the individual segments as well as the representation bitrates and network capacity in determining routing paths for DASH over SDN. Although we focus on DASH standard, our proposal can also be utilized for DASH-like HTTP Adaptive Streaming (HAS) architectures. The contributions of the proposed study of segment based rerouting for DASH applications can be listed as follows:

- We propose a path selection algorithm taking segment bitrates, representation bitrates and current network capacity into account for segment based rerouting of DASH flows.
- The proposed algorithm aims to provide high QoE while providing fairness among clients. In order to provide fairness along the streaming session, the next path assignment for a client is done by considering the throughput received by that client until that moment.
- We consider the bitrates of the prospective future segments when estimating the near future network capacity and reroute the flows based on this estimation. This is a proactive approach used in determining the streaming paths for DASH clients.
- We show the performance of the proposed approach by giving the results of various rerouting periods comparatively under various network conditions and topologies.

In previous section, we proposed to select the streaming path for a given client by considering the bitrate of its currently requested segment (Cetinkaya et al., 2014). In this work, we propose an approach taking the bitrates of the segments into account while determining routing paths of DASH packets in an SDN domain. This study differs from the study given in the previous section (Cetinkaya et al., 2014) since the streaming paths are determined for all clients in the system and packets are dynamically rerouted by considering the current network load and bitrates of the future segments. In the next section, we give the motivation behind segment based rerouting approach.

3.6.1 Motivation

In this section, we give the motivation of taking the segment bitrates into account in the path selection strategy for DASH application running over SDN. In DASH architecture, although the bitrate information given in the MPD file indicates the bitrates of the representations, these values are average values of bitrates of the segments in the representations. However, the bitrates of the segments vary significantly. In Fig. 3.14 and Fig. 3.15, the distribution of the segment bitrates of Big Buck Bunny video's (Lederer et al., 2012) 720p and 1080p representations are given, respectively. The figure shows the difference between the bitrates of the segments belonging to same representation can be up to 2 Mbps for 720p video and 8.5 Mbps for 1080p video.

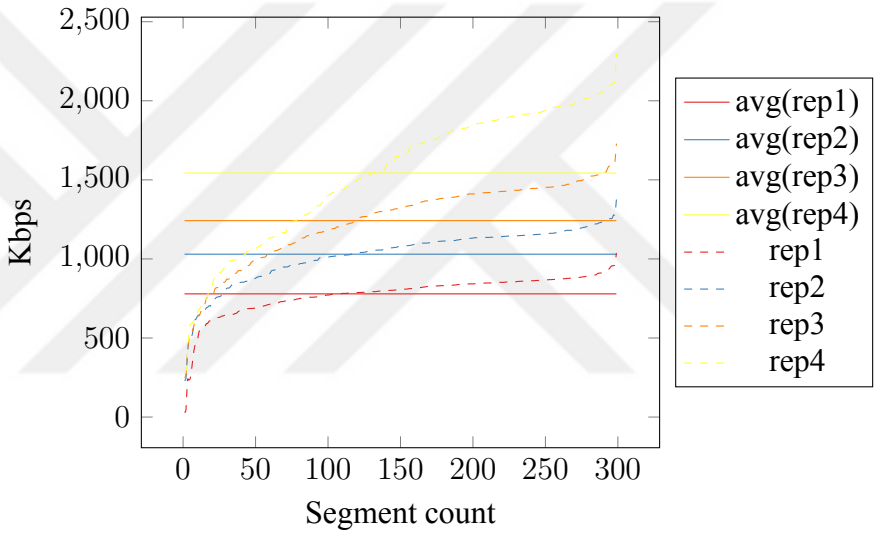


Figure 3.14. Representation: 720p

Consider the problem of path selection for conveying the packets of DASH clients in an SDN domain. One solution is to select the path having maximum capacity for each client connecting to the server. Later, we will show that this approach may not provide the expected performance in terms of QoE and network utilization. Suppose that the controller assigns paths by taking the bitrate of the representations and available bandwidth of the paths into account. However, considering the bitrate of the representations may lead the controller to make non-optimal path assignment due to the variations of segment bitrates as seen from the values given in Fig. 3.14 and Fig. 3.15.

In this work, we propose a path selection strategy by considering segment bitrates and available bandwidth of the paths. The details of the proposed algorithm

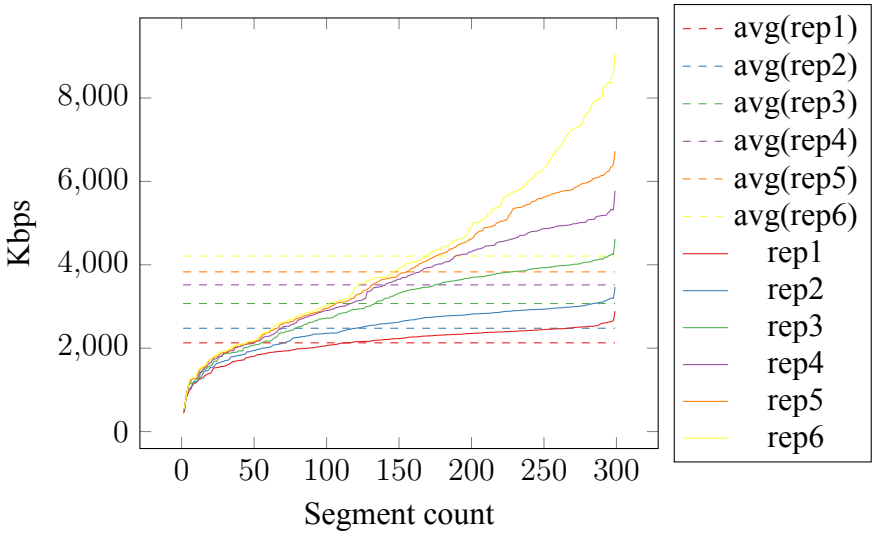


Figure 3.15. Representation: 1080p

are given in the next section.

3.6.2 Proposed Work

At the beginning of a streaming session, forwarding tables of OpenFlow enabled switches are empty. When a DASH client starts the streaming application, it sends the TCP SYN packet to the server to establish the TCP connection. Upon receiving this message, the switch sends a *PacketIn* message to the controller to learn the forwarding rule determined for this client. The controller determines the path according to the segment based flow routing algorithm explained in this section and sends the related commands to the switches along the selected path based on the output of the algorithm. As an input of the segment based flow routing algorithm, the controller requires having the information of the requested segments bitrate, hence it must have the information about the number of the currently requested segments. The controller can acquire this information by communicating with the clients. However, communicating with all clients in the system is not a scalable approach. In the proposed system, the server periodically communicates with the controller to give information about the requested segments by the clients. The communication steps of the proposed system are given in Fig. 3.16. As shown in the figure, while clients request segments from the server according to their rate adaptation algorithm (step 1), the server periodically informs the controller about the requested video segments (step 2). The controller determines the set of paths for the clients in the system considering the bitrates of the requested segments and the current network capacity (step 3). After selecting the set of paths for the clients, the controller sends related flow information to the corresponding switches via *Flow-*

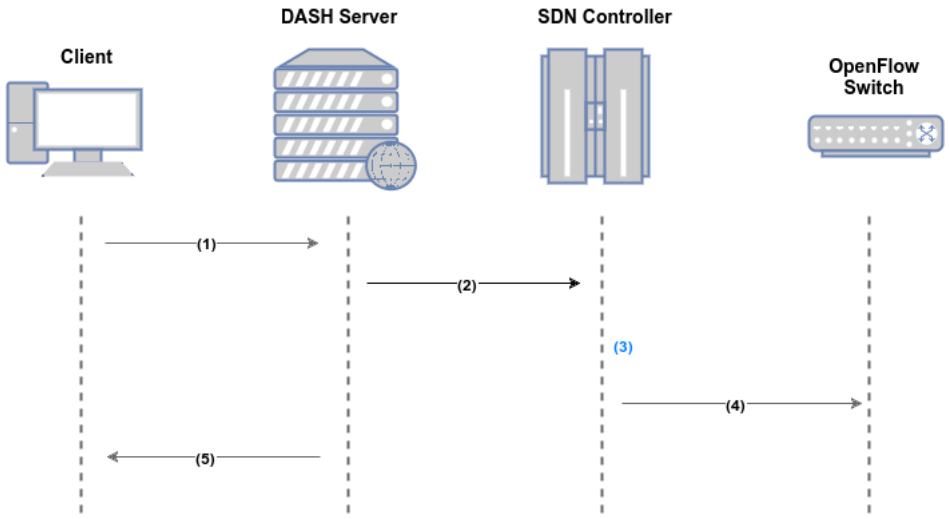


Figure 3.16. The communication steps of the proposed system.

Mod messages (step 4). Finally, the server starts sending the video requested segments over the newly defined paths for each client (step 5).

In the process given in step 3 in Fig. 3.16, the controller determines the streaming paths for all clients by considering the bitrate of the current requested segments, average throughput received by the clients and available bandwidth information. Determining the optimal set of streaming paths requires controller to have the knowledge about current network capacity. In order to acquire information about current network load, the controller runs an algorithm estimating the total available capacity of the network before determining the streaming paths for the clients. The algorithm used for estimating the current network capacity is given in Fig. 3.17. Available bandwidth information for all paths is required by the controller since network capacity is determined by using this information. In order to measure available bandwidth, the controller periodically communicates with the switches and obtains traffic statistics for the flows other than DASH flows. These statistics are smoothed by using formula (3.55) and available bandwidth value of a path p, which is denoted as abw_p , is calculated by using formula (3.56). In the formulas, tr_p and $capacity_p$ represent the estimated traffic load and capacity of path p, respectively. α is set to 0.6 to give a slightly more importance to the recently measured traffic (Cetinkaya et al., 2014).

$$tr_p = \alpha \times (recently\_measured\_traffic) + (1 - \alpha) \times tr_{p-1} \tag{3.55}$$

Input:

P: The set of paths between the server and the clients
abw_p: The available bandwidth of path $p \in P$ (measured using formula 2)
abw_l: The available bandwidth of link l
l_p: The set of links of path p
br_{min}:The lowest representation bitrate

Output:

Estimated total capacity

1: *estimated total capacity = 0 ;*
2: **while** (at least one path $p \exists\ abw_p \geq br_{min}$) **do**
3: $p_{max} = max_p \{ abw_p \}$;
4: estimated total capacity = estimated total capacity + $abw_{P_{max}}$;
5: **for** ($\forall\ l \in l_{P_{max}}$) **do**
6: $abw_l = abw_l - abw_{P_{max}}$;
7: **end for**
8: $P = P / \{P_{max}\}$;
9: **for** ($\forall\ p \in P$) **do**
10: $abw_p =$min $\{ abw_l, l \in l_p \}$;
11: **end for**
12: **end while**
13: **return** estimated total capacity ;

Figure 3.17. Network Capacity Estimation Pseudocode

$$abw_p = capacity_p - tr_p \tag{3.56}$$

The network capacity estimation algorithm is run by the controller when a new flow other than DASH flow started or ended. The controller detects that a new flow is started when it receives a *PacketIn* message from the switch. On the other hand, if a flow is idle for a pre-defined threshold, the switches detecting idle flows inform the controller by sending a *FlowRemoved* message. Hence, the controller detects the ended flows and reruns the network capacity estimation algorithm given in Fig. 3.17. In line 3 of the algorithm, the capacity of a path having maximum available capacity, p_{max}, is added to the estimated total capacity if the available bandwidth of the path is greater than or equal to the lowest representation bitrate (this condition is tested in line 2 of the algorithm). In the for loop starting in line 5 of the algorithm, the available bandwidth of the links composing p_{max} are decreased because the capacity values of the links are used in the capacity calculation. p_{max} is removed from the path set (line 8) and the capacities of the remaining paths are recalculated (line 10). The algorithm continues to run until there is no path having the capacity greater than or equal to the lowest representation bitrate remaining in the path set.

When a client starts the video streaming application, it first receives the MPD file from the server and starts sending its requests for downloading the segments consecutively. In the proposed system, the server periodically sends the information of the requested segments to the controller. This period is managed by a static variable called *rerouting_period (rp)*. The server determines one of the clients in the system as the pivot client [3] and it sends the id of the requested segments of all clients in each multiple of *rerouting_period* based on the id of the requested segments by the pivot client. Suppose there are 3 clients connected to the DASH server, *rerouting_period* is 10 and the 1st client is the pivot client. When the first client requests the 10^{th}, 20^{th}, 30^{th} ... segments; the server sends the id of the latest requested segments by these 3 clients.

When the controller receives the message containing the id of the requested segments of the clients from the server, it executes the algorithm given in Fig.3.18. Note that this algorithm is run at the step 3 in the communication steps shown in Fig. 3.16. The output of the algorithm is the streaming paths determined for each client. In the for loop in line 1 of the algorithm, the streaming paths are assigned to the clients, starting from the first client in the list. The client list is sorted by the controller in descending order according to the average received bitrate from the beginning of the streaming session and the list order is updated each time the algorithm 2 runs. If the estimated total capacity, which is the output of algorithm 1, is sufficient enough to send video at moderate quality to the clients (line 1), the paths are assigned in ascending order according to the available bandwidth of the paths (line 3). Hence, the first client in the list, which has the lowest average received bitrate, has the highest priority when assigning streaming paths. Here, available bandwidth is calculated by using the formula 3.56.

If total capacity of the network is limited, then the estimated total capacity is compared to the estimated sum of bitrates of possible future segments of the clients (line 6). This summation is calculated by the formula (3.57), where br_s^h represents the bitrate of the segments of the current representation requested by client h. For each client, the future segment having maximum bitrate is considered in the calculation. Although the client may change the representation until the next rerouting period, this calculation gives an approximation for the summation of future segments bitrates and it is sufficient enough for providing seamless streaming service in the worst case.

[3] Generally, the client that first joins the system is selected as pivot client. If the pivot client leaves the system, the server selects the client which is foremost in terms of the requested segments as the new pivot client. The new pivot is the client requested the most amount of segments so far.

$$\sum_{h \in H} (max\{br_s^h, ..., br_s^h + rp\}) \tag{3.57}$$

If estimated total capacity is greater than or equal to the estimated sum of bitrates of possible future segments of the clients, the client having the highest average received bitrate is assigned to the path which has sufficient available bandwidth (line 9). For the remaining clients, the paths having maximum available bandwidth are selected (line 11). The estimated available bandwidth of the path *p*, which is represented as $estimated\_abw\_p$ is calculated according to formula (3.58).

$$estimated\_abw_p = abw_p - \sum_{h \in H} (max\{br_s^h, ..., br_s^h + rp\}) \tag{3.58}$$

If network is highly congested, the path assignment strategy is similar to the previous condition, but average available bandwidth is considered when paths are assigned to the clients not having the highest received bitrate in this case (line 21). The calculation of the average available bandwidth of the path p used in line 21 of the algorithm is given in formula (3.59). In the formula, avg_abw_p represents the average available bandwidth and p_{client_count} represents the number of clients assigned to the path p. If there are more than one DASH flows on the path p, streaming throughput does not achieve the maximum available bandwidth due to the competing TCP flows. By considering the behavior of competing TCP flows, available bandwidth value is multiplied by β, which is set to 0.7 (Cetinkaya et al., 2014). Note that the behavior of competing flows is only considered when the network is highly congested since more accurate bandwidth estimation is needed under congestion in order to minimize the duration of outages.

$$avg\_abw_p = \frac{abw_p}{p_{clientcount} + 1} \times \beta \tag{3.59}$$

The segment based flow route selection algorithm designed for DASH clients aims to provide fairness among clients as well as seamless streaming service. In order to provide fairness, if network is congested, the client having the highest received bitrate is assigned to the path having a limited available bandwidth as much as sufficient enough to transfer the video with minimum quality. In the next section, we present the performance of the proposed segment based path selection algorithm.

Input:

H: List of clients (sorted in descending order according to the average received bitrate)

P: Set of paths between the server and the clients

H_p: Set of clients assigned to the path $p \in P$

abw_p: The available bandwidth of path $p \in P$

P_{client_count}: The number of clients assigned to the path p

l_p: The set of links of path p

br_{min}:The lowest representation bitrate

br_{med}:The medium representation bitrate

Output:

streaming path set assigned to the clients

Initialization:

$\forall$ p $\in$ P, $p_{client_count} = 0$;

$\forall$ p $\in$ P, $h_p = \{\}$;

1: **if** (*estimated total capacity* $\geq | H | \times br_{med}$ **then**
2: **for** (each $h \in H$) **do**
3: selected path p for h $\max_p \{ abw_p \}$;
4: $P_{client\_count} = P_{client\_count} + 1$;
5: **end for**
6: **else if** (*estimated total capacity* $\geq$ the sum of bitrates of possible future segments) **then**
7: **for** (each $h \in H$) **do**
8: **if** (h has the highest average received bitrate) **then**
9: selected path p for $h = \min_p \{estimated\_abw_p \geq br_{min}\}$;
10: **else**
11: selected path p for $h = \max_p \{abw_p\}$;
12: **end if**
13: $H_p = H_p \cup \{h\}$;
14: $abw_p = abw_p - br_{med}$;
15: **end for**
16: **else**
17: **for** (each $h \in H$) **do**
18: **if** (h has the highest average received bitrate) **then**
19: selected path p for h $\min_p \{estimated\_abw_p \geq br_{min}\}$;
20: **else**
21: selected path p for $h = \max_p \{avg\_abw_p\}$;
22: **end if**
23: $H_p = H_p \cup \{h\}$;
24: **end for**
25: **end if**

Figure 3.18. Segment-based flow route selection algorithm

3.6.3 Simulation Study

There are one video server and multiple DASH clients in the network topologies used in the simulations. The video sent by the DASH server is 1080p video which details are given in Table 3.1. We also implemented a different routing approach which is similar to the studies proposed for improving the performance of HTTP streaming over SDN in the literature (Nam et al., 2014), (Jarschel et al., 2013) in order to give comparable results. In the comparison study, when network capacity changes due to the cross traffic, the streaming path of the DASH client having the highest received bitrate is changed to the streaming path having maximum available bandwidth. For this purpose, the controller monitors the traffic in order to detect capacity changes in the network and changes the streaming path after detecting change in the capacity. We refer to the comparison study as *max-flow routing approach*. The topologies and simulation parameters are the same for the proposed and for the comparison study in all simulations. Each simulation is repeated 30 times for each parameter set and results are given by averaging the results. In this section we show the performance of the proposed system by giving QoE related parameters such as the average received bitrate and the duration of pauses and by evaluating the provided fairness among DASH clients. In order to evaluate performance under different network conditions, we execute two sets of simulations under two different network operations as static and dynamic.

3.6.3.1 Static Network Operations

In static set of simulations, two network topologies are used in order to observe the performance of the proposed system under constant traffic. For both topologies, the volume of the cross traffic does not change and the capacity has remained the same during the simulations. In the first topology, there are three paths in the network. The available bandwidth values of the paths equal to 2000 Kbps, 2000 Kbps and 3900 Kbps, respectively. In the second topology, there are two paths and the available bandwidth values of the paths equal to 4100 Kbps and 4500 Kbps, respectively. There are three clients connected to the DASH server in both topologies.

In Fig. 3.19, average, minimum and maximum received bitrate are given comparatively for the simulations running over the first topology. The bitrate values given in the graph represent the received representation bitrates by the clients; hence these values indicate the received video quality by the clients. The average received quality for each rerouting period in the proposed approach is similar to the *max-flow routing* approach. The difference between both approaches is seen in the minimum

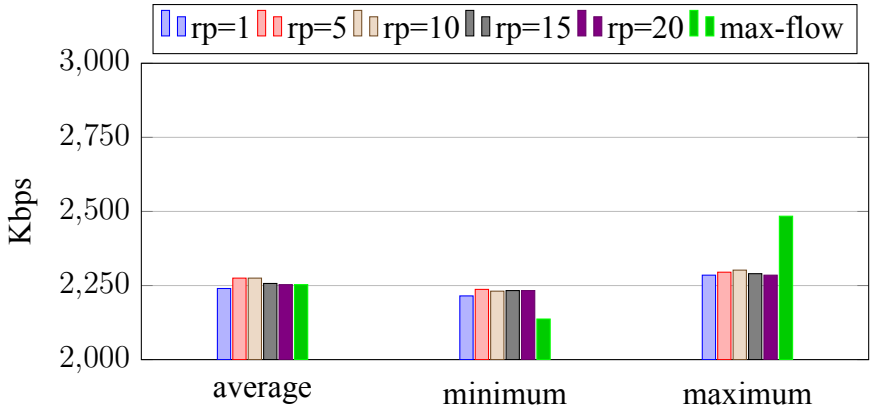


Figure 3.19. Average, minimum and maximum received bitrate observed in the simulations over the first topology.

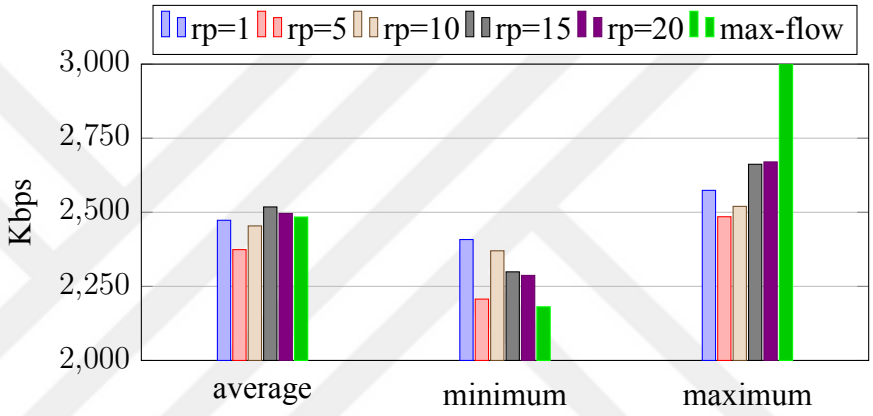


Figure 3.20. Average, minimum and maximum received bitrate observed in the simulations over the second topology.

and maximum quality received by the clients. On the other hand, a significant effect of rerouting period on the received bitrate values is not observed in this topology. Average, minimum and maximum received bitrate values are given for the second topology in Fig. 3.20. Observed results obtained from the second topology are similar to those of the first topology. For both topologies, while the difference between minimum and maximum received quality in the proposed approach is small, the difference between these values is larger in the max-flow routing approach. These results show that fair sharing of network resources is provided. In addition that, average received quality by the clients is higher than the minimum video quality even for the highly congested network. Minimum received bitrate reaches its maximum value when rerouting period is set to 1, while setting this period to 10 gives the best performance in terms of average received bitrate.

In Table 3.9, the observed statistics related to buffer underflow outage duration values are given for the first topology. In the graph, the average outage duration

Table 3.9. Observed outage duration statistics in the simulations over the first topology.

	Average	Max-total	Max-indv.
rp=1	0	0	0
rp=5	0.33	2	1
rp=10	0.22	1	1
rp=15	1.44	3	1
rp=20	6.53	14	1
max-flow	51.77	79	2

Table 3.10. Standard deviation values of the received quality.

	First Topology	Second Topology
rp=1	39	88
rp=5	33	147
rp=10	38	76
rp=15	29	192
rp=20	28	193
max-flow	200	490

value represents the averaged outage duration observed by a client in all simulations. Total outage duration is calculated by summing the outage duration observed in a client during a streaming session. The maximum total outage duration observed in a client is given in the graph. Maximum outage duration refers to the maximum of individual outage durations experienced. Because the links in the first topology are is highly congested, clients frequently experience outages with max-flow routing approach. When max-flow routing approach is executed, the average outage duration observed in the clients lasts 51.77 seconds. This value equals to 6.33 seconds at most in the proposed approach. There is no outage observed in the seconds topology since the link capacities are not limited in this topology.

The standard deviation of the received video bitrate values are given in Table 3.10 in order to reveal how fair the proposed approach is. As seen from the values given in the table, the video quality achieved by DASH clients is similar in the proposed segment-based flow routing approach.

3.6.3.2 Dynamic Network Operations

In the dynamic set of simulations, simulations are performed over larger network topologies. In these simulations, the amount of cross traffic dynamically changes in order to show the performance of the proposed system under dynamic network conditions. In the simulations, there are 3, 5, 10, 15, 20 DASH clients connected to the server, respectively. Total network capacity is set to a value that is proportional to the number of clients in each simulation. Total network capacity values equal to 18 Mbps, 30 Mbps, 60 Mbps, 90 Mbps and 120 Mbps for 3, 5, 10, 15 and 20 clients, respectively. When the simulation starts, clients are assigned to the path having maximum available bandwidth in both proposed and comparison approaches. Cross traffics are launched on 30% of the paths after all clients join the system. These paths are chosen randomly. The amount of cross traffic on a path equals to the 50% of the path capacity and the cross traffic is active for 100 seconds. 100 seconds after cross traffic is ended, another cross traffic is launched on 30% of the paths. This time, cross traffics are active for 200 seconds and the amount of the traffic on a path equals to 80% of the capacity of the path. The simulations last for 600 seconds.

The average received bitrate, the minimum received bitrate and the maximum received bitrate as a function of the number of the clients are provided in Fig. 3.21, Fig. 3.22 and Fig. 3.23, respectively. When there are three clients in the system, the comparison study reaches its best performance. The main reason is that the overall system performance is noticeably improved for limited number of clients since only one flow, i.e. 30% of the flows, is rerouted in the comparison approach. Accordingly, when the number of clients in the system increases, the performance of the comparison approach decreases because rerouting only one flow effects increasingly smaller portion of all flows on the network. Note that even if the average received bitrate is higher than the average received bitrate obtained in the proposed approach, the minimum received bitrate in the comparison approach is lower when 3 clients in the system. The results also indicate that the average received quality obtained in the proposed approach is higher than that of the max-flow routing approach, especially if the number of clients in the system is greater than 3. Hence, these results also show that the performance of the proposed system is preserved at the same level regardless of the number of clients in the system. Smaller rerouting period values provides increase in received bitrate values, however the difference between the received bitrates values obtained with rerouting period being 1 and 20 is at most 3%.

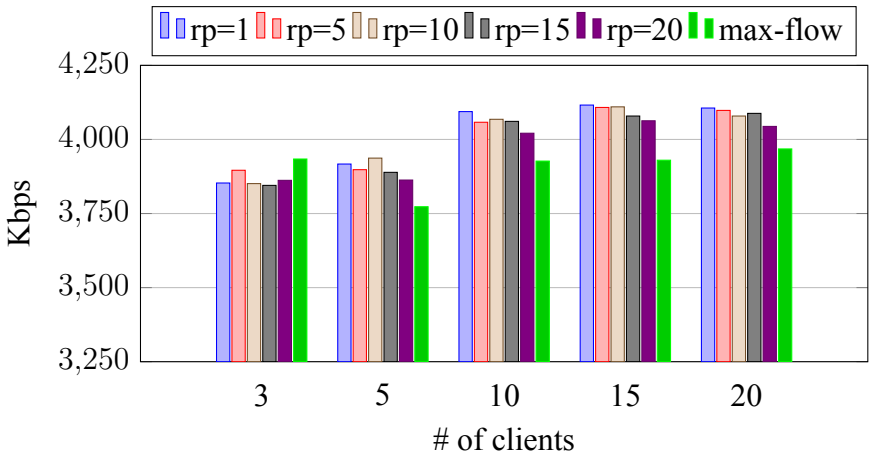


Figure 3.21. Average received bitrate observed in the dynamic network conditions.

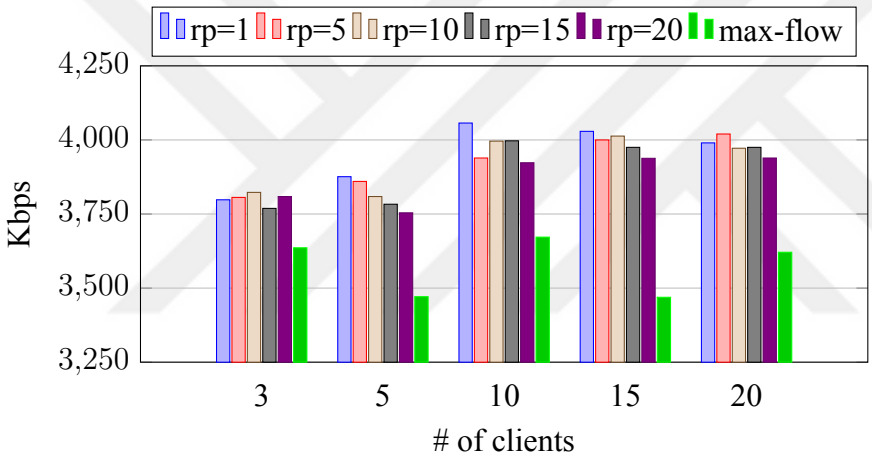


Figure 3.22. Minimum received bitrate observed in the dynamic network condition.

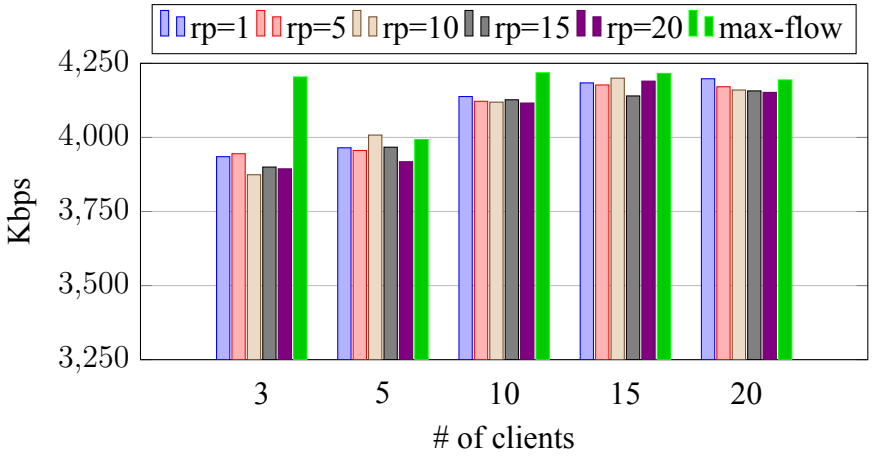


Figure 3.23. Maximum received bitrate observed in the dynamic network condition.

Table 3.11. Outage duration observed in the dynamic network conditions.

		# of clients				
		3	**5**	**10**	**15**	**20**
Average	**rp = 1**	0	0	0	0	0
	rp = 5	0	0	0	0	0
	rp = 10	0	0	0	0	0
	rp = 15	0	0	0	0	0
	rp = 20	0	0	0	0	0
	max-flow	18	37	12	17	14
Max-total	**rp = 1**	0	0	0	0	0
	rp = 5	0	0	0	0	0
	rp = 10	0	0	0	3	0
	rp = 15	0	0	1	2	2
	rp = 20	0	1	3	2	1
	max-flow	54	93	44	64	62
Max-Indv.	**rp = 1**	0	0	0	0	0
	rp = 5	0	0	0	0	0
	rp = 10	0	0	0	1	0
	rp = 15	0	0	1	2	2
	rp = 20	0	1	1	1	1
	max-flow	3	4	2	3	2

The average outage duration, the maximum total outage duration and the maximum individual outage duration are provided in Table 3.11, respectively. It is observed that the outage durations experienced in the proposed approach are small. However, the outage duration observed in the max-flow routing approach is up to 40 seconds. This variation between outage durations between two approaches is mainly based on the fact that the bitrates of the individual segments are considered in the proposed approach. By considering the bitrate of the segments, the clients requiring higher bandwidth are detected and the paths are assigned according to capacity requirements in the proposed approach.

In Table 3.12, the standard deviation values are given for the seconds set of topologies having dynamic network capacity. These values are small for the proposed approach. Hence, this shows that the desired fairness is also provided under dynamic network conditions. When obtained standard deviation values are examined, the best performance in terms of fairness is provided when rerouting period equals to 5 or 10.

Table 3.12. Standard deviation values of the received quality in the second set of simulations

	# of clients				
	3	**5**	**10**	**15**	**20**
rp = 1	72	39	28	51	50
rp = 5	78	35	62	45	46
rp = 10	25	110	41	67	60
rp = 15	68	95	41	46	51
rp = 20	45	94	78	71	62
max-flow	285	212	236	284	223

In this work, the advantages of dynamic rerouting of flows by considering the bitrates of the segments and network conditions are shown for DASH applications. For this purpose, we have presented an algorithm for estimating the current network load by considering the bitrates of video representations and an algorithm for determining the streaming paths for each client in the system. The controller takes current network load and the bitrate of the future segments into consideration in order to determine the streaming paths. Streaming paths determined for each client are rerouted dynamically in order to provide seamless video streaming service and to provide fairness among DASH clients.

When the performance is examined considering the frequency of rerouting periods, the best performance in terms of received video quality is obtained for rerouting period equals to one segment and five segments. The main reason of that is high frequency in rerouting causes to provide quick response to the changes in the segment bitrates and network conditions. However, since rerouting in each segment causes an increase in messaging complexity, it is recommended that rerouting period should be small only if the number of clients in the system is limited. When the number of clients is high, scalability can be preserved by grouping clients and setting different rerouting period to these groups individually. Then the network operator can set the rerouting period by considering the size of these groups and by considering the trade-off between the message complexity and the performance in terms of expected QoE.

The performance of the proposed system is shown by comparing it to max-flow routing approach given in the literature. Extensive simulations performed in Mininet environment show that the proposed flow routing approach outperforms max-flow routing in terms of the received bitrate, the outage duration and the fair allocation of network resources among clients in the system. For dynamic network

conditions, on average, 3% increase in average received bitrate, 11% increase in minimum received quality, 48% decrease in outage duration, and 62% decrease in standard deviation of received bitrates are observed in the proposed system compared to the max-flow routing approach.

4. SDN-ALTO INTEGRATION FOR VIDEO STREAMING APPLICATIONS

Today, most people enjoy the video streaming services running over Internet. Video-on-Demand (VoD), Live Streaming, Over-the-Top (OTT) TV services offered by the companies such as Netflix[1], Youtube[2], Hulu[3] can be given as examples to video streaming services. For a video streaming system company or a network operator that also serves a video streaming service to its customers, maximizing the QoE or providing the guaranteed QoS based on SLAs (SLA) is the most crucial part of the streaming system. The works proposed in the previous chapter shows that using SDN improves the performance of the video streaming systems. Also, SDN technology can be a good alternative to improve the performance of the video streaming systems especially if network operators co-operate with video streaming service company (Wichtlhuber et al., 2015).

Since video streaming clients and the servers are distributed over the world, the end-to-end streaming path between the servers and the clients may across multiple ISPs. Hence, some level of information on network characteristics of end-to-end paths is required by the video streaming applications to provide a certain level of performance. Application Layer Traffic Optimization (ALTO) protocol is standardized by taking such Internet application's requirements into account. ALTO provides abstracted physical network information such as available bandwidth, delay, hop count, routing costs (Alimi et al., 2014). While network operators share some information about its domain, they can also receive information about network characteristics of other ISPs from ALTO server (Alimi et al., 2014).

In this chapter, the proposed works on video streaming architecture utilizing SDN-ALTO integration are presented. The rest of the chapter is organized as follows: First, background information about ALTO protocol is given. Second, the related works about SDN-ALTO integration is summarized. Third, the design of the SDN-ALTO interoperability for multimedia services is presented. Fourth, a CDN-based Video-on-demand system architecure is proposed. Fifth, a CDN server selection algorithm is proposed to increase the QoE of the clients. And for last, a CDN server selection algorithm is proposed that reduces the cost of ISPs while increasing QoE of clients.

[1] www.netflix.com
[2] www.youtube.com
[3] www.hulu.com

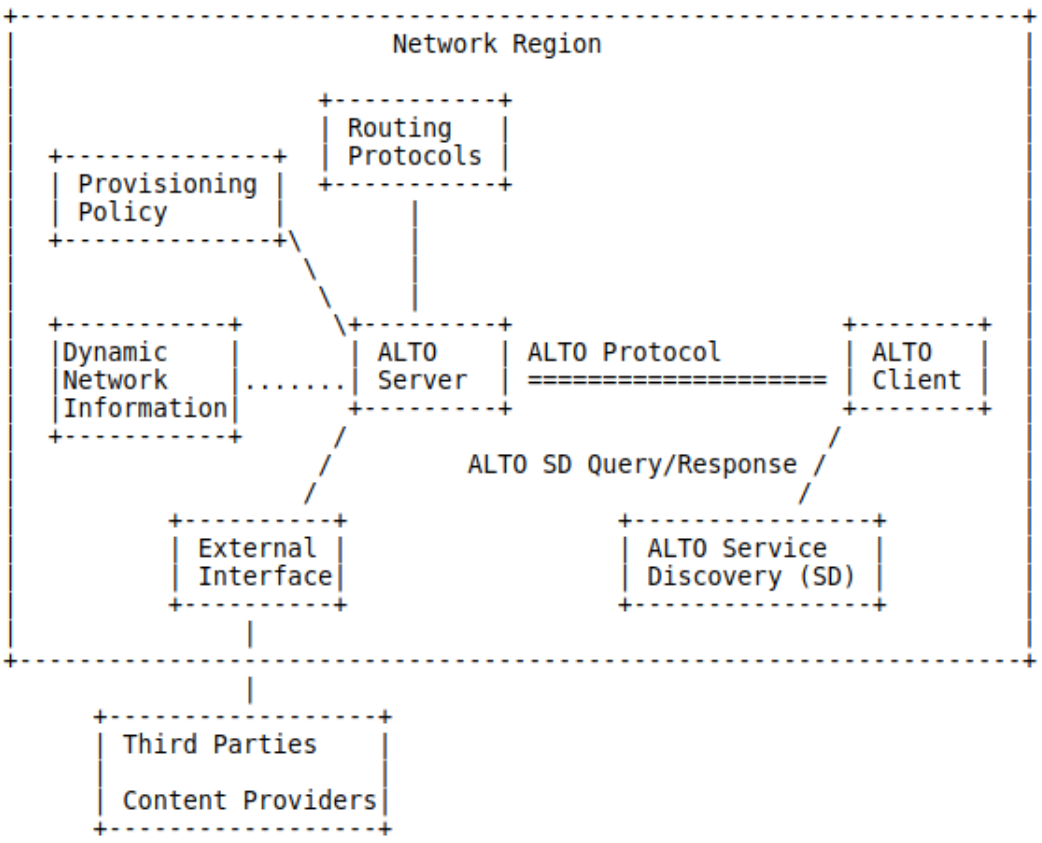


Figure 4.1. General overview of the ALTO architecture (Alimi et al., 2014)

4.1 Application Layer Traffic Optimization

Network applications such as files sharing applications, peer-to-peer video streaming applications and web applications can perform better if they get underlying network-related information. ALTO protocol provides several services related to path or ISP costs to give information to the overlay applications.

General overview of the ALTO system architecure illustrated in Fig.4.1. In the architecture, ALTO client queries the ALTO server by using ALTO protocol and receives network-related information. ALTO client obtains the information of the appropriate ALTO Server that it will communicate by using ALTO server discovery service. ALTO server aggregates the ALTO information from multiple sources such as static network databases, dynamic network information, routing protocols, provisioning policies and third parties. There are four ALTO services, namely Map Service, Map Filtering Service, Endpoint Cost Service and Endpoint Property Service. These services are offered by ALTO server to the ALTO clients via ALTO protocol. The hosts or group of hosts are defined as endpoints. While Map Service and Map Filtering Services provide information of the location of the hosts, End-

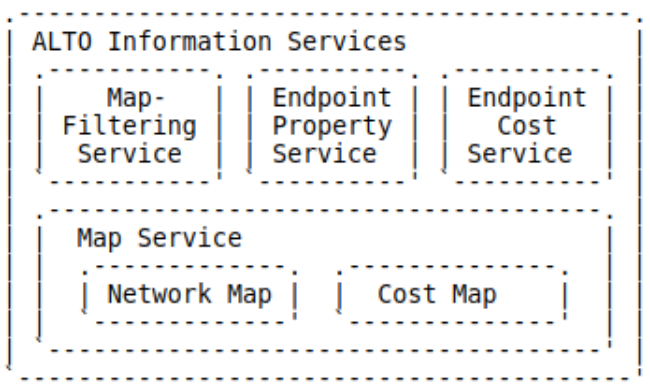


Figure 4.2. General overview of ALTO information service framework (Alimi et al., 2014)

point property service gives the information about the characteristics of the hosts, such as connectivity type and network location. Endpoint Cost Service provides cost information related to the endpoint pairs. Hop count, bandwidth, administrative ISP costs between the pairs of endpoints can be given as examples to the costs. Defined as in (Alimi et al., 2014), in order to obtain more than one cost type with using Endpoint Cost Service, ALTO clients are required to send a request for each cost type. Since this approach increases the messaging traffic and burden on the ALTO server, ALTO working group has recently proposed to use multi-cost ALTO which allows clients to query multiple metrics with a single request.

4.2 Related Works

There are several studies proposing a CDN-based VoD system running over SDN in the literature. In (Egilmez and Tekalp, 2014), the streaming paths between the video server and the clients residing in different SDN domains are defined based on the messages exchanged between the controllers of each domain. In (Rego et al., 2015), a load balancing strategy among virtual servers via OpenFlow is proposed for a cloud-CDN video streaming system. High volume flows' TCP connection migration from a CDN server to another one by deploying SDN switches in a provider network is proposed in (Wichtlhuber et al., 2015). In an SDN domain, a path and server selection framework for Scalable Video Coding (SVC) video manycast is proposed in (Xue et al., 2015). In (Nam et al., 2014), path and server route selection strategy is proposed for DASH clients running over an SDN domain. If a client experiences outage in the video streaming session, SDN controller either changes the intra route between the client and the gateway router or changes the CDN server outside the SDN domain, depending on where the problem lies (Nam et al., 2014). None of the proposed systems utilize ALTO services for obtaining information about

end-to-end paths spanning over multiple ISPs in the above studies.

ALTO and SDN integration is proposed in (Gurbani et al., 2012) and communication models for ALTO and SDN with various use cases are discussed in (Xie et al., 2012). The idea of utilizing ALTO services for SDN controllers in different domains to determine end-to-end paths is presented in (Yin et al., 2017). The authors state that network applications such as peer-to-peer file sharing or video on demand systems can be supported by SDN-ALTO integration. However, application specific methods are not introduced in (Yin et al., 2017). An HTTP video streaming service utilizing the advantages of ALTO and SDN integration is proposed in (Faigl et al., 2014). The server selection request sent by the clients is redirected by the SDN controller which has knowledge about underlying network conditions. In this system there is not any strategy defined for determining end-to-end paths where the server and the clients are in different ISPs.

4.3 SDN & ALTO Interoperability For Multimedia Services

In this section, a video streaming system architecture utilizing SDN and ALTO is proposed. Differing from the literature, in the proposed architecture, SDN controllers feed information to the ALTO server as well as receive abstracted information from it in order to maximize the achieved QoE. The proposed work has the following capabilities:

- Providing ALTO services and two-way communication between ALTO server and SDN controllers.
- Abstraction of SDN domain network characteristics at ALTO server.
- Provision of an SDN controller module that obtaining and evaluating of information of network where the servers reside in and directing the clients to the appropriate server in its domain.

Fig.4.3 illustrates the general overview of the proposed architecture. In the proposed architecture, the SDN controllers are part of the video streaming system and have information about up-to-date capacity/usage information about video servers residing in their domains. The controller of the SDN domain periodically communicates with the ALTO server for two reasons. First, the controller sends the current server capacity information as well as smoothed available bandwidth and delay information of the current route between the gateway router and router that

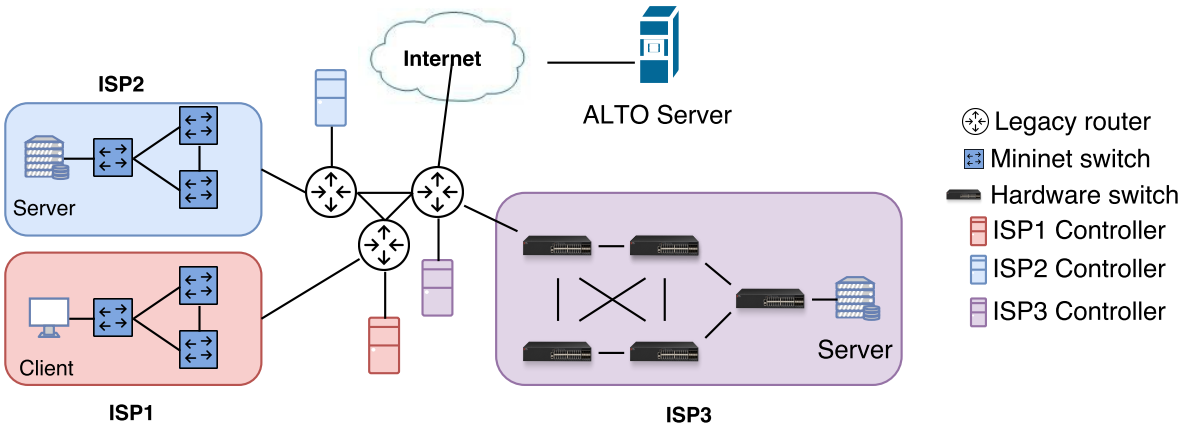


Figure 4.3. Demonstration of the proposed video streaming architecture utilizing SDN and ALTO.

the server(s) connected to. Note that if the network operator follows a policy that limits the flow amount of outside traffic, it may send smoothed available bandwidth information by considering this policy. Second, the controller obtains abstracted network information related to remote ISPs by querying ALTO server.

ALTO protocol provides Map Filtering Service, Endpoint Property Service and Endpoint Cost Service. Map and Map filtering services are related to location of endpoints, i.e. hosts. In this architecture we implemented and enhanced the Endpoint property service of ALTO protocol where this service provides information about the endpoints. Endpoint groups are defined as Provider-defined Identifiers (PID) by ALTO. We define each SDN domain as a PID. When ALTO server receives server capacity and smoothed available bandwidth information from a controller, ALTO server normalizes these values. Normalized values are ranked in a list with its numerical values determined based on normalization. SDN controller acting as an ALTO client queries ALTO server by using ALTO Endpoint property service and obtains the information about ISPs where video servers reside.

When a client starts the video streaming application, it sends Domain Name System (DNS) query to its local DNS in order to obtain IP address of a server. SDN controller also serves as the local DNS server. By using the information received from ALTO server, the controller determines the server having the maximum value of capacity and available bandwidth pair in the ranked list. The controller sends the IP address of the selected server to the client as DNS reply. After establishing TCP connection, the client starts to download video segments from the server based on the output of its rate adaptation algorithm.

4.4 CDN-Based Video-On-Demand System Architecture Utilizing SDN-ALTO Interoperability

In this section, a VoD system architecture utilizing SDN-ALTO integration for CDN-based video streaming applications is proposed (Cetinkaya and Sayit, 2016). To the best of our knowledge, this is the first study proposing a VoD architecture which utilizes ALTO-SDN integration.

In traditional CDN-based video streaming applications, the video is stored in CDN servers that is geographically distributed over the world and residing in different ISPs (Nygren et al., 2010). Whenever a client requests to play a video, the client is redirected via DNS messages to the suitable CDN server. The selection of the suitable CDN server is based on the policy of the CDN company. In order to redirect the client to a CDN server, client's local DNS server sends a query to the authoritative DNS server of the video streaming company. Authoritative DNS relays this query to the authoritative DNS server of the CDN company, where the IP address of the selected server is returned to the client in the end. The selection of the suitable CDN server can be done by taking the delay between the local DNS of the client and the CDN servers or load of the CDN servers into consideration (Kurose and Ross, 2012). In order to measure the delay between the CDN servers and the client to determine the best server selection, the query of local DNS can be sent to the different CDN servers or the client's TCP connection are directed to random servers. This approach may degrade the performance of the video streaming applications (Kurose and Ross, 2012).

In the proposed VoD architecture, the clients that run the CDN-based video streaming application are considered residing in an SDN-enabled access network. SDN controller determines the most suitable CDN server for clients and the intra-ISP routing path between the CDN server and the clients.

SDN controller of the domain communicates with the ALTO server to get the abstracted information such as hop-count, delay, routing-cost between itself and CDN servers residing in other ISPs. SDN controller also exchanges messages with the CDN company to obtain information about CDN servers such as IP addresses and loads. With the information received from both ALTO server and CDN company, the SDN controller can redirect the clients to CDN server. The benefits of this approach are twofold. First, since CDN company has no information about underlying network information of the client, selecting the optimal CDN server based with information that SDN controller has, reduces the costs of CDN company. On

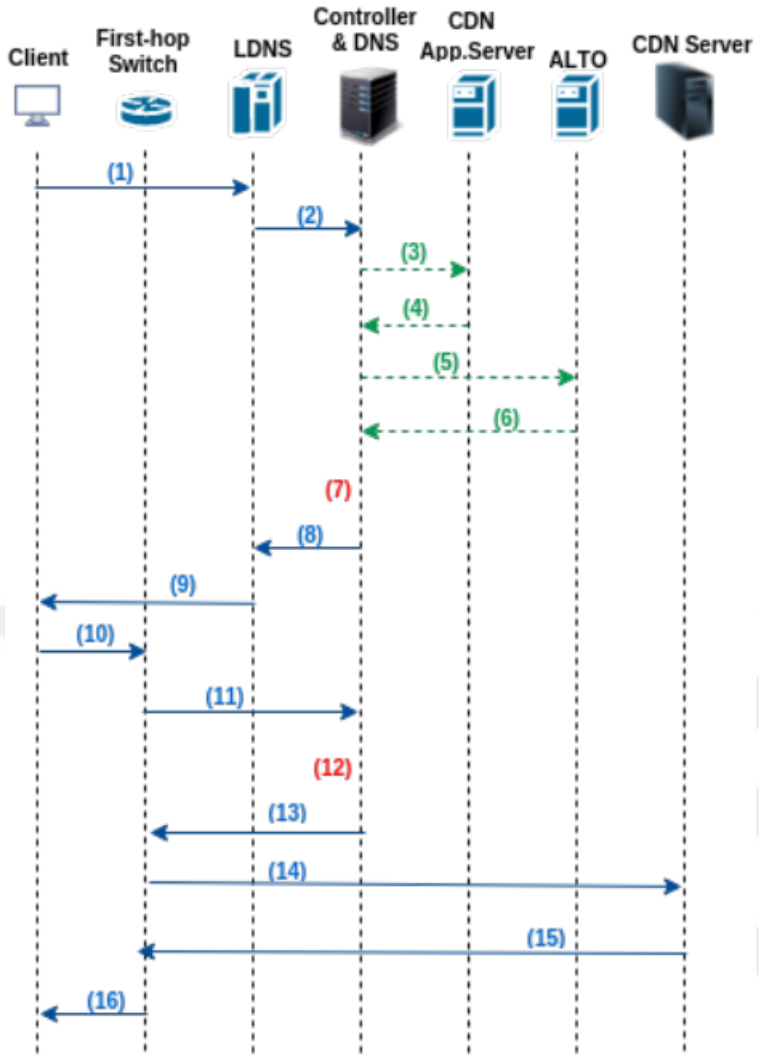


Figure 4.4. Messaging timeline of the proposed VoD system arhitecture.

the other hand, considering the loads of the CDN servers in server selection process prevent extra burden on the servers. Fig. 4.4 illustrates the messaging timeline of the proposed VoD system architecture.

- When a client requests the video e.g. clicks the URL of the requested video, the browser triggers a DNS query to the local DNS server (LDNS) of the client (1) in order to receive the IP address.
- SDN controller also runs a DNS server and by default all DNS queries from LDNS to authoritive DNS server of the CDN company are redirected to DNS server running on SDN controller (2). Therefore, SDN controller receives the client's video request by analyzing the DNS query.
- SDN controller determines the suitable CDN server for client (7) by using

information received from CDN company (Step 3-4) and ALTO server (Step 5-6).

- DNS server running on the SDN controller returns the IP address of the determined CDN server to LDNS (8).
- The LDNS returns the IP address of the CDN server to client (9).
- After receiving the IP address of the CDN server, the client sends video request to the CDN server (10).
- The first-hop switch that receives packets of the client's request looks up it's flow table if there is any matching flow for the client request. Since the flow-table does not contain any flows related to client's request, the switch sends a *PacketIn* message to the SDN controller (11) in order to learn where to forward the packets of the request.
- After receiving the *PacketIn* message from the switch, the controller determines a suitable path between client and the CDN server (12).
- The controller sends the flow information via *FlowMod* messages to the corresponding switches along the path (13).
- Finally the video streaming session starts after the connection between the client and the CDN server is set (14-16).

4.4.1 CDN Server Selection

As mentioned earlier at step (7) in Fig. 4.4, by using the network-related information received from ALTO server and CDN company, the SDN controller determines a suitable CDN server for the client. In the decision process of the suitable CDN server, the SDN controller may consider the requirement of the SLAs guaranteed by the video streaming company. In Section 4.5 and Section 4.6, different CDN server selection algorithms are given.

4.4.2 Intra-ISP Route Selection

After determining the CDN server at step (7), the controller should select the intra-ISP path for the flow between client and the selected CDN server at step (12) in Fig. 4.4. There are two cases for the selection of the intra-ISP route:

- CDN server is within the SDN domain, then the controller can determine the end-to-end path between client and the selected CDN server.

- CDN server is in another ISP, then the controller determines the path between the client and the gateway router sending packets to that ISP. Note that, this gateway router is determined by Border Gateway Protocol (BGP) (Rekhter et al., 2006) since it requires an inter- ISP communication.

In both case, the SDN controller determines the streaming path by considering available bandwidth of the intra-ISP paths. In the proposed studies given in Section 4.5 and Section 4.6, the SDN controller selects the path which has maximum available bandwidth as the intra-ISP streaming path.

4.5 CDN Server Selection For Improving The Quality Of Experience

In this section, a multiobjective optimization model is proposed to select a suitable CDN server that improves the QoE of the clients (Cetinkaya and Sayit, 2016).

4.5.1 Proposed Work

As mentioned in previous section, SDN controller determines the suitable CDN server for client (7) by using information received from ALTO server (3-4) and CDN company (5-6). In the proposed work, at step (3-4) the SDN controller receives the IP addresses, loads of CDN servers and the average end-to-end delay between the CDN servers and the SDN domain from CDN company. At step (5-6), the SDN controller receives the routing-cost from the ALTO server. In this study, the routing-cost received from the ALTO server is the available bandwidth information between the ISPs.

Let N represents the video servers that CDN company distributes the video content through. After completing step (3-6) the SDN controller has three values for each CDN server; load, delay and cost. The optimization model aims to find the server which minimizes these three values. For this purpose, we define an utopia point which indicates the optimal solution. The solution of the optimization is the closest point to the utopia among the solutions. First, the variables of each CDN server are normalized. Let S represent the set of CDN servers. For each of $x \in S$; $norm_x^{ld}$ represents the normalized value of the load of x, $norm_x^{d}$ represents the normalized value of the delay to the x and, $norm_x^{c}$ represents the normalized cost value for connecting x. These normalized values are obtained by the equation (4.1) and the row vector of normalized variables (4.2) are constructed for each x. In (4.1),

y_x represents the original y value of x, y_{min} represents minimum value of y and y_{max} represents maximum value of y where $y \in \{ld, d, c\}$.

$$norm_x^y = \frac{y_x - y_{min}}{y_{max} - y_{min}} \tag{4.1}$$

$$\mathbf{N_{(x)}} = \left[norm_x^{ld}, norm_x^d, norm_x^c\right] \tag{4.2}$$

Network capacity and server load directly affects the received video quality while delay effects startup latency. Therefore, minimizing server load and ALTO cost is more important than minimizing delay. To give more importance to the load and cost variables, we assign different weights to them and normalized the variables by taking the weights into account. Let ω denotes the row vector of the weights assigned to the objective variables. Normalized objective variables are weighted (4.3) calculating the inner product of ω and $\mathbf{N_{(x)}}$.

$$\nabla_x = \langle \omega, \mathbf{N_x} \rangle \tag{4.3}$$

Let λ denotes the utopia point. The optimization model is given as follows:

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & \|\nabla_x, \lambda\| \\ \text{subject to} \quad & x \in S \end{aligned} \tag{4.4}$$

The model finds the server x, which optimization variables are closest to the utopia point. The optimization model is solved by exhaustive search. If the number of CDN servers is excessive, then a subset of the CDN servers can be used in the model to put a limit to the computational complexity of the calculations. The CDN server subset can also be selected by considering geographic proximity.

4.5.2 Simulation Study

The network topology of the simulation setup is shown in Fig. 4.5. The topology consists of an access SDN-ISP and three ISP networks. There are four CDN servers located in each ISP. The bandwidths of the links connecting CDN servers are limited as [5000-20000] Kbps. The average bitrate of the video is 2500 Kbps. There are ten video clients requesting video during the simulation session. ω weight

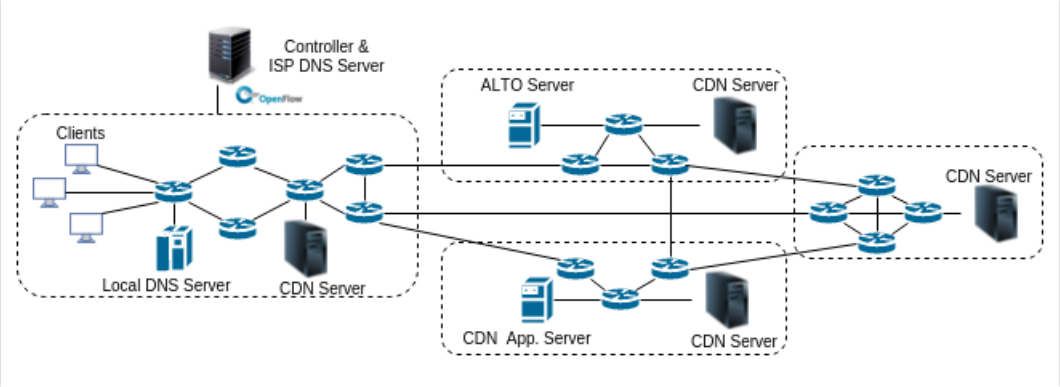


Figure 4.5. Network topology used in the simulations.

vector used in the optimization model is set to $\langle 0.35, 0.50, 0.15 \rangle$ where the elements of the vector denote the CDN load, ALTO cost and delay respectively.

In order to evaluate the performance of the proposed system, we have implemented another approach and present the results of the both approaches in comparison. In the second approach, the clients are redirected to CDN servers using a round-robin load balancing. The simulations are repeated 10 times and the results are obtained by taking the outcome of each simulation average into consideration. We measure the received bitrate, startup delays and outage durations observed in the clients during the simulations.

In Fig. 4.6, the Cumulative Distribution Function (CDF) graph of average bitrate (throughput) received by the clients in each simulation are given for both approaches. As seen from the graph, 0.05% of the average received bitrate values are less than 5000 Kbps in the proposed approach while 25% of these values are less than 5000 Kbps in the load balanced CDN selection. On the other hand, more than 20% of the average received bitrate values measured with load balanced approach are higher than 14 Mbps. The distribution of the measured throughput shows that network resources are not fairly shared among clients even if the loads of the CDN servers are balanced.

Since TCP is used as underlying transport protocol, packets are never lost and arrive to the clients eventually. However, a delay in the arrival time of packets might occur if the client is connected to a heavily-loaded CDN server or the streaming path between the client and the CDN server has inadequate bandwidth. As a result, the video buffer of the client might drain. In this case, client suffers from outages. In Table 4.1, minimum, maximum and average outage durations observed in the

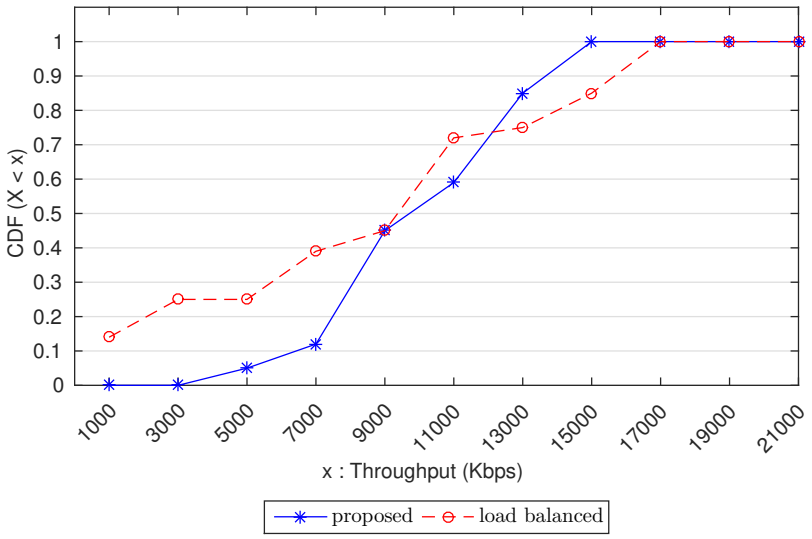


Figure 4.6. CDF graph of the throughput

Table 4.1. Outage Duration Statistics

Outage Duration (in sec.)	**Min**	**Max**	**Avg**
Proposed	0	0	0
Load-Balanced	0	207	73.33

clients are given. In the table, maximum refers to the maximum of outage durations observed in a client while maximum total refers to the maximum of total outage duration observed in all clients during a streaming session. Average outage duration values are obtained by averaging the measured outage durations of all clients in all simulations. The clients play the video seamlessly with the proposed approach while the clients experience outage durations with the load balanced approach.

Startup delay is the amount of time for video to start up after client's request. In Table 4.2, the startup delay values that clients experience are given. In the table, minimum, maximum and average values are calculated as the same way in outage durations. The minimum startup delays are same in both approaches. However, the clients in the load balanced approach may experience a startup delay up to 18 seconds.

The simulation results show that QoE in terms of average received throughput, startup delay and outage duration increased with ALTO guidance compared to the traditional method using load balancing among CDN servers.

Table 4.2. Average Startup Delay

Startup Delay (in sec.)	Min	Max	Avg
Proposed	4	8	5.8
Load-Balanced	4	18	8.8

4.6 CDN Server Selection For Reducing ISP Cost

In this section, a multiobjective optimization model is proposed to select a suitable CDN server that improves the QoE of the clients and reduces the cost of ISPs (Cetinkaya and Sayit, 2017). In this study, differing from the study that proposed in the previous section, SDN controller uses ALTO multi-cost queries in order to obtain the information about underlying network characteristics requiring inter-ISP communication.

4.6.1 Proposed Work

In the proposed work, at step (3-4) the SDN controller receives the IP addresses, loads of CDN servers from CDN company. At step (5-6), the SDN controller queries for the multi-cost values including routing costs, delay and ISP administrative cost between the client and CDN servers.

The information retrieved from ALTO server is numerical multicost values. Although these values are not the real cost or delay values, they are calculated by ALTO server based on the real values. SDN controller runs a multi-objective optimization algorithm by using ALTO multi-cost values and load of CDN servers as the input variables of the optimization. The optimization model is given in Eq. (4.5-4.11)

$$\min(a_1 \cdot l_s) \tag{4.5}$$

$$\min(a_2 \cdot costISP_s) \tag{4.6}$$

$$\min(a_3 \cdot delay_s) \tag{4.7}$$

$$\max(a_4 \cdot capacity_s) \tag{4.8}$$

with subject to

$$s \in S \tag{4.9}$$

$$delay_s < thr_{delay} \tag{4.10}$$

$$l_s < thr_{load} \tag{4.11}$$

The expressions from Equation 4.5 to Equation 4.8 are the optimization objectives, where l_s represents the normalized load of CDN server s, $costISP_s$, $delay_s$ and $capacity_s$ represents the normalized values of ISP administrative cost, delay and capacity of the path between ISP of the client and ISP of CDN server s, respectively. In the model, S is the set of CDN servers, thr_{delay} is the delay threshold value defining maximum acceptable delay and thr_{load} is the load threshold value defining maximum acceptable load of a CDN server. Delay threshold value for live video streaming applications can be selected more stringent than for VoD applications.

The optimization model is solved for finding a solution s representing the selected CDN server. Optimization objectives are weighted according to the importance of the parameters and a values given from Equation 4.5 to Equation 4.8 represents these weights. In order to solve the optimization, we define the utopia point where administrative cost, delay and CDN load equals to zero and capacity equals to its maximum value. Optimal solution is the server s having the minimum distance to this utopia point.

4.6.2 Simulation Study

The network topology of the simulation setup is shown in Fig. 4.7. The topology consists of an access SDN-ISP and four ISP networks. There is one CDN server located in each ISP. The bandwidths of the links connecting CDN servers are limited as [7-50] Mbps. The delays between the ISPs are selected in between 0.13 and 1.20 msec. The average bitrate of the video is 4 Mbps. There are 20 clients requesting video during the simulation session.

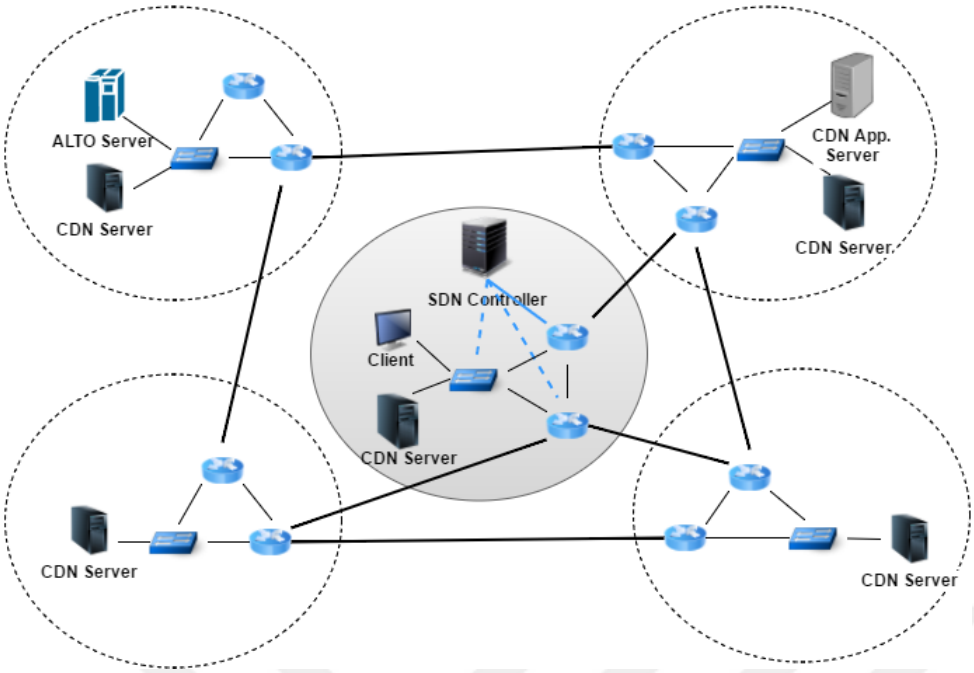


Figure 4.7. Network topology used during the simulations.

In the simulations, the proposed optimization model is tested under two different weight sets. In the first weight set (*Proposed-w1*), the weight of the paths' capacity between ISP of the client and ISP of CDN servers (a_4) is selected as 0.4 where the other weights remain 0.2. In this set, the optimization model tends to select the CDN server which has the most inter-ISP path capacity. In the second weight set (*Proposed-w2*), the weight of the ISP administrative cost (a_2) and the weight of the paths' capacity (a_4) are selected as 0.33 where the weight of CDN server load (a_1) and ISP delay (a_3) remain 0.17. Hence, in the second set, CDN servers are selected by giving more importance to reduce inter-ISP communication.

In order to evaluate the performance of the proposed system, we have implemented another approach and present the results of the both approaches in comparison. In the second approach, the clients are redirected to CDN servers using a round-robin load balancing. The simulations are repeated 10 times and the results are obtained by taking the outcome of each simulation average into consideration. We measure the received bitrate, startup delays and outage durations observed in the clients during the simulations. We also give the ISP administrative cost values measured in all approaches.

In Fig. 4.8, the graph of the average bitrate (throughput) received by the clients in each simulation are given. *Proposed-w1* and *Proposed-w2* represents the simulations performed with the first and the second set of weights, respectively. Average

received bitrate values are obtained by taking the average of the measured throughput in all clients for all simulations. In the graph, max throughput and min throughput values represent the maximum and minimum measured throughput in one client during all simulations, respectively. As seen from the graph, the highest average received throughput and the highest minimum throughput is measured with the first set of weights *Proposed-w1* in the proposed approach. If the importance of the ISP administrative cost is increased, the received bitrate reduces. However, the quality is still good enough as the minimum throughput of clients equals to the video bitrate with *Proposed-w2*.

Since TCP is used as underlying transport protocol, packets are never lost and arrive to the clients eventually. However, a delay in the arrival time of packets might occur if the client is connected to a heavily-loaded CDN server or the streaming path between the client and the CDN server has inadequate bandwidth. As a result, the video buffer of the client might drain. In this case, client suffers from outages. Minimum, maximum and average outage durations observed in the clients are given in Table 4.3. Maximum and minimum outage duration refers to the maximum and minimum outage durations observed in a client, respectively. Average values in the outage durations are obtained by averaging the total outage duration observed in the clients for all simulations. The clients play the video almost seamlessly with both proposed approaches while the clients in the load balanced approach experience long duration of outages.

Startup delay is the amount of time for video to start up after client's request. In Table 4.4, the startup delay values that clients experience are given. In the table, minimum, maximum and average values are calculated as the same way in outage durations. The minimum startup delays are similar for all approaches. However, the clients in the load balanced approach may experience a startup delay up to 27 seconds.

In Fig. 4.9, the cost of inter-ISP communication is shown. The values show the average cost for all clients during the streaming session. These values are calculated based on the cost values given by the ALTO server. We define cost as a cost paid for transferring one block of communication unit (packet or bps). The maximum possible cost value given by ALTO server equals to 100. In the simulations, intra-ISP administrative cost value is set to 0 and inter-ISP administrative cost values are set to 20, 45, 30 and 100 for remaining four ISPs, respectively.

Performance evaluations show that the proposed architecture helps to increase

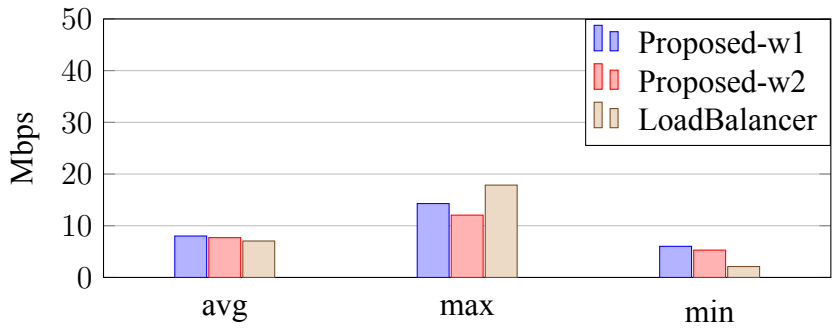


Figure 4.8. Average throughput graph.

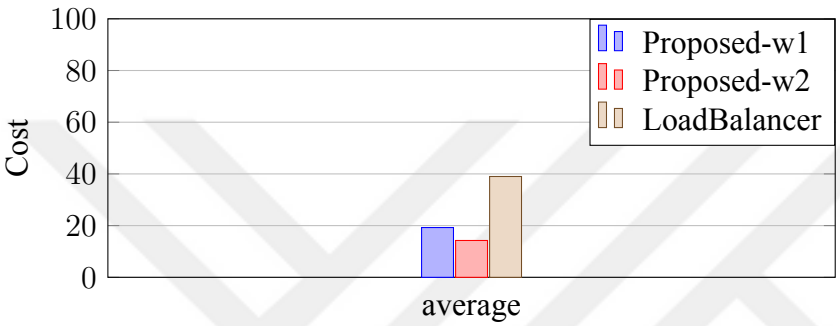


Figure 4.9. Inter-ISP administrative cost graph

Table 4.3. Outage Duration Statistics (sec)

	min	max	avg
Proposed-w1	0	0	0
Proposed-w2	0	0	0
LoadBalancer	0	99	22.9

Table 4.4. Startup Delay (sec)

	min	max	avg
Proposed-w1	5	10	7.9
Proposed-w2	4	12	8.75
LoadBalancer	3	27	12.35

the performance of the VoD and live video streaming applications when compared to load balanced CDN server selection approach. If more weight is given to ISP administrative costs in the optimization, inter-ISP communication costs are reduced while received video quality is still satisfactory.

5. CONCLUSION

This thesis aims to improve the performance of the video streaming applications utilizing SDN. The studies proposed in this thesis can be classified in two categories.

As the first category, some novel route selection methods are proposed for DASH running over an SDN domain. For this purpose first, a route selection method is proposed for the selection of the paths between the DASH clients and the server. The aim of the proposed work is to improve the QoE of the clients by increasing the average received bitrate and to provide fairness among clients by decreasing the difference between the bitrates received by the clients.

Second, an optimization framework is proposed for increasing QoE of DASH clients and for providing service classes in an SDN domain. The controller dynamically changes the streaming paths between the server and the clients by taking the underlying network conditions into account. Streaming paths are determined according to the optimization framework. Although the optimization framework is given for two different service classes, it can be enhanced for more service classes.

Third, a novel segment based route selection method by considering the bitrate of the requested segments is proposed for DASH clients running over SDN. The proposed optimization model aims to find the most suitable path by considering the available bandwidth, bitrate of the current segment, competing flows and the path length. We also observe required buffer ratio for the measured available bandwidth to provide seamless streaming.

Fourth, based on the segment based route selection method, an algorithm for estimating the current network load by considering the bitrates of video representations and an algorithm for determining the streaming paths for each client in the system are proposed. The controller takes current network load and the bitrate of the future segments into consideration in order to determine the streaming paths. Streaming paths determined for each client are rerouted dynamically in order to provide seamless video streaming service and to provide fairness among DASH clients.

Simulation results show that, thanks to the SDN that allows to develop specific routing algorithms considering the requirement of the applications, the QoE of the DASH clients is improved in terms of received bitrate, outage duration, startup delay and quality switching.

As the second category, SDN-ALTO integration for video streaming applications is proposed. For this purpose first, an architecture is proposed for integrating SDN and ALTO for multimedia services. In the architecture, the SDN controller acts as an ALTO client and receives the network related information from ALTO. The SDN controller provides ALTO server with the inter-domain network information such as available bandwidth of the paths.

Second, a novel CDN-based VoD architecture is proposed based on the SDN and ALTO integrated multimedia service.

Third, a method is proposed to improve the QoE of the clients for suitable CDN server selection process in the CDN-based VoD architecture.

Fourth, differing from the third, a method is proposed to improve the QoE of the clients while reducing the cost of ISPs for suitable CDN server selection process.

Performance evaluations show that the proposed CDN-based VoD architecture helps to increase the QoE of video streaming applications in terms of average received throughput, startup delay and outage duration increased with ALTO guidance compared to the traditional systems. We believe that network application architectures based on SDN-ALTO interaction can be considered as one of the future internet frameworks. The design of the proposed CDN-based VoD architecture can also be used as guidance for other types of network applications.

REFERENCES

Alimi, R., Penno, R., Yang, Y., Kiesel, S., Previdi, S., Roome, W., Shalunov, S. and Woundy, R., 2014. Application-Layer Traffic Optimization (ALTO) Protocol. RFC 7285. URL `https://rfc-editor.org/rfc/rfc7285.txt`.

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

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

Bentaleb, A., Begen, A.C. and Zimmermann, R., 2016. SDNDASH: Improving QoE of HTTP Adaptive Streaming Using Software Defined Networking. In Proceedings of the 2016 ACM Conference on Multimedia Conference, MM 2016, Amsterdam, The Netherlands, October 15-19, 2016. p. 1296–1305.

Cetinkaya, C., Karayer, E., Sayit, M. and Hellge, C., 2014. SDN for segment based flow routing of DASH. In 2014 IEEE Fourth International Conference on Consumer Electronics Berlin (ICCE-Berlin). p. 74–77.

Cetinkaya, C., Ozveren, Y. and Sayit, M., 2015a. Route optimization for DASH over Software Defined Networks. In Signal Processing and Communications Applications Conference (SIU), 2015 23th. IEEE, p. 2454–2457.

Cetinkaya, C., Ozveren, Y. and Sayit, M., 2015b. An SDN-assisted system design for improving performance of SVC-DASH. In Computer Science and Information Systems (FedCSIS), 2015 Federated Conference on. p. 819–826.

Cetinkaya, C. and Sayit, M., 2016. Video-on-demand system architecture with ALTO-SDN integration. In Black Sea Conference on Communications and Networking (BlackSeaCom), 2016 IEEE International. IEEE, p. 1–5.

Cetinkaya, C. and Sayit, M., 2017. ALTO-assisted CDN-based video streaming over SDN. In Signal Processing and Communications Applications Conference (SIU), 2017 25th. IEEE, p. 1–4.

Cicco, L.D., Caldaralo, V., Palmisano, V. and Mascolo, S., 2013. ELASTIC: A Client-Side Controller for Dynamic Adaptive Streaming over HTTP. In 2013 20th Int'l Packet Video Workshop. p. 1–8.

Cisco, 2017. Cisco Visual Networking Index: Forecast and Methodology, 2016–2021.

REFERENCES (continued)

Egilmez, H.E., Civanlar, S. and Tekalp, A.M., 2013. An Optimization Framework for QoS-Enabled Adaptive Video Streaming Over OpenFlow Networks. *IEEE Transactions on Multimedia*, 15(3):710–715.

Egilmez, H.E., Dane, S.T., Bagci, K.T. and Tekalp, A.M., 2012. OpenQoS: An OpenFlow controller design for multimedia delivery with end-to-end Quality of Service over Software-Defined Networks. In Signal Information Processing Association Annual Summit and Conference (APSIPA ASC), 2012 Asia-Pacific. p. 1–8.

Egilmez, H.E., Gorkemli, B., Tekalp, A.M. and Civanlar, S., 2011. Scalable video streaming over OpenFlow networks: An optimization framework for QoS routing. In 2011 18th IEEE International Conference on Image Processing. p. 2241–2244.

Egilmez, H.E. and Tekalp, A.M., 2014. Distributed QoS Architectures for Multimedia Streaming Over Software Defined Networks. *IEEE Transactions on Multimedia*, 16(6):1597–1609.

Faigl, Z., Szabó, Z. and Schulcz, R., 2014. Application-layer traffic optimization in software-defined mobile networks: A proof-of-concept implementation. In 2014 16th International Telecommunications Network Strategy and Planning Symposium (Networks). p. 1–6.

Farshad, A., Georgopoulos, P., Broadbent, M., Mu, M. and Race, N., 2015. Leveraging SDN to provide an in-network QoE measurement framework. In 2015 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS). p. 239–244.

Georgopoulos, P., Elkhatib, Y., Broadbent, M., Mu, M. and Race, N., 2013. Towards Network-wide QoE Fairness Using Openflow-assisted Adaptive Video Streaming. In Proceedings of the 2013 ACM SIGCOMM Workshop on Future Human-centric Multimedia Networking. ACM, New York, NY, USA, FhMN '13, p. 15–20.

Gurbani, V.K., Scharf, M., Lakshman, T.V., Hilt, V. and Marocco, E., 2012. Abstracting network state in Software Defined Networks (SDN) for rendezvous services. In 2012 IEEE International Conference on Communications (ICC). p. 6627–6632.

REFERENCES (continued)

Huang, T.Y., Johari, R., McKeown, N., Trunnell, M. and Watson, M., 2014. A Buffer-based Approach to Rate Adaptation: Evidence from a Large Video Streaming Service. In Proc. of the 2014 ACM Conf. on SIGCOMM. p. 187–198.

Jain, R., Chiu, D.M. and Hawe, W.R., 1984. A quantitative measure of fairness and discrimination for resource allocation in shared computer system.

Jarschel, M., Wamser, F., Hohn, T., Zinner, T. and Tran-Gia, P., 2013. SDN-Based Application-Aware Networking on the Example of YouTube Video Streaming. In Proceedings of the 2013 Second European Workshop on Software Defined Networks. IEEE Computer Society, Washington, DC, USA, EWSDN '13, p. 87–92.

Jiang, J., Sekar, V. and Zhang, H., 2014. Improving Fairness, Efficiency, and Stability in HTTP-Based Adaptive Video Streaming With Festive. *IEEE/ACM Transactions on Networking*, 22(1):326–340.

Karl, M., Gruen, J. and Herfet, T., 2013. Multimedia optimized routing in OpenFlow networks. In 2013 19th IEEE International Conference on Networks (ICON). p. 1–6.

Kleinrouweler, J.W., Cabrero, S. and Cesar, P., 2016. Delivering Stable High-quality Video: An SDN Architecture with DASH Assisting Network Elements. In Proceedings of the 7th International Conference on Multimedia Systems. ACM, New York, NY, USA, MMSys '16, p. 4:1–4:10.

Kurose, J.F. and Ross, K.W., 2012. Computer Networking: A Top-Down Approach (6th Edition). Pearson, 6th edition.

Laga, S., Cleemput, T.V., Raemdonck, F.V., Vanhoutte, F., Bouten, N., Claeys, M. and Turck, F.D., 2014. Optimizing scalable video delivery through OpenFlow layer-based routing. In 2014 IEEE Network Operations and Management Symposium (NOMS). p. 1–4.

Lantz, B., Heller, B. and McKeown, N., 2010. A Network in a Laptop: Rapid Prototyping for Software-defined Networks. In Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks. ACM, New York, NY, USA, Hotnets-IX, p. 19:1–19:6.

Lederer, S., Müller, C. and Timmerer, C., 2012. Dynamic Adaptive Streaming over HTTP Dataset. In Proceedings of the 3rd Multimedia Systems Conference. ACM, New York, NY, USA, MMSys '12, p. 89–94.

REFERENCES (continued)

Li, Z., Zhu, X., Gahm, J., Pan, R., Hu, H., Begen, A.C. and Oran, D., 2014. Probe and Adapt: Rate Adaptation for HTTP Video Streaming At Scale. *IEEE JSAC*, 32(4):719–733.

McDonagh, P., Olariu, C., Hava, A. and Thorpe, C., 2013. Enabling IPTV Service Assurance Using OpenFlow. In Advanced Information Networking and Applications Workshops (WAINA), 2013 27th International Conference on. p. 1456–1460.

McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S. and Turner, J., 2008. OpenFlow: Enabling Innovation in Campus Networks. *SIGCOMM Comput. Commun. Rev.*, 38(2):69–74.

Mu, M., Broadbent, M., Farshad, A., Hart, N., Hutchison, D., Ni, Q. and Race, N., 2016. A Scalable User Fairness Model for Adaptive Video Streaming Over SDN-Assisted Future Networks. *IEEE Journal on Selected Areas in Communications*, 34(8):2168–2184.

Nam, H., Kim, K.H., Kim, J.Y. and Schulzrinne, H., 2014. Towards QoE-aware video streaming using SDN. In 2014 IEEE Global Communications Conference. p. 1317–1322.

Nygren, E., Sitaraman, R.K. and Sun, J., 2010. The Akamai Network: A Platform for High-performance Internet Applications. *SIGOPS Oper. Syst. Rev.*, 44(3):2–19.

Open Networking Foundation, 2012. Software-Defined Networking: The New Norm for Networks. White paper, Open Networking Foundation, Palo Alto, CA, USA.

Petrangeli, S., Wauters, T., Huysegems, R., Bostoen, T. and Turck, F.D., 2015. Network-based dynamic prioritization of HTTP adaptive streams to avoid video freezes. In 2015 IFIP/IEEE International Symposium on Integrated Network Management (IM). p. 1242–1248.

Quinlan, J.J., Zahran, A.H., Ramakrishnan, K.K. and Sreenan, C.J., 2015. Delivery of adaptive bit rate video: balancing fairness, efficiency and quality. In The 21st IEEE International Workshop on Local and Metropolitan Area Networks. p. 1–6.

Rainer, B., Lederer, S., Muller, C. and Timmerer, C., 2012. A seamless Web integration of adaptive HTTP streaming. In EUSIPCO. p. 1519–1523.

REFERENCES (continued)

Ramakrishnan, S., Zhu, X., Chan, F. and Kambhatla, K., 2015. SDN Based QoE Optimization for HTTP-Based Adaptive Video Streaming. In 2015 IEEE International Symposium on Multimedia (ISM). p. 120–123.

Rego, P.A.L., Bonfim, M.S., Ortiz, M.D., Bezerra, J.M., Campelo, D.R. and de Souza, J.N., 2015. An OpenFlow-Based Elastic Solution for Cloud-CDN Video Streaming Service. In 2015 IEEE Global Communications Conference (GLOBECOM). p. 1–7.

Rekhter, Y., Hares, S. and Li, D.T., 2006. A Border Gateway Protocol 4 (BGP-4). Technical Report 4271. URL `https://rfc-editor.org/rfc/rfc4271.txt`.

Seddiki, M.S., Shahbaz, M., Donovan, S., Grover, S., Park, M., Feamster, N. and Song, Y.Q., 2014. FlowQoS: QoS for the Rest of Us. In Third Workshop on Hot Topics in Software Defined Networking. p. 207–208.

Shi, H., Prasad, R.V., Onur, E. and Niemegeers, I.G.M.M., 2014. Fairness in Wireless Networks:Issues, Measures and Challenges. *IEEE Communications Surveys Tutorials*, 16(1):5–24.

Sodagar, I., 2011. The MPEG-DASH Standard for Multimedia Streaming Over the Internet. *IEEE MultiMedia*, 18(4):62–67.

Uzakgider, T., Cetinkaya, C. and Sayit, M., 2015. Learning-based approach for layered adaptive video streaming over {SDN}. *Computer Networks*, 92, Part 2:357 – 368. Software Defined Networks and Virtualization.

Valdivieso Caraguay, A.L., Puente Fernández, J.A. and García Villalba, L.J., 2015. Framework for Optimized Multimedia Routing over Software Defined Networks. *Comput. Netw.*, 92(P2):369–379.

Wichtlhuber, M., Reinecke, R. and Hausheer, D., 2015. An SDN-Based CDN/ISP Collaboration Architecture for Managing High-Volume Flows. *IEEE Trans. on Network and Service Management*, 12(1):48–60.

Xie, H., Tsou, T., Lopez, D., Yin, H. and Gurbani, V., 2012. Use cases for ALTO with software defined networks. *Working Draft, IETF Secretariat, Internet-Draft draft-xie-alto-sdn-extension-use-cases-01. txt*.

Xue, N., Chen, X., Gong, L., Li, S., Hu, D. and Zhu, Z., 2015. Demonstration of OpenFlow-Controlled Network Orchestration for Adaptive SVC Video Manycast. *IEEE Trans. on Multimedia*, 17(9):1617–1629.

REFERENCES (continued)

Yin, H., Zou, T. and Xie, H., 2017. Defining data flow paths in software-defined networks with application-layer traffic optimization. URL `https://www.google.com/patents/US9729424`, uS Patent 9,729,424.

CURRICULUM VITAE

Cihat ÇETİNKAYA

Address: International Computer Institute, Izmir/TURKEY
Mobile: (+90) 507 708 79 10
E-mail:cihat.cetinkaya@ege.edu.tr

Personal Information
Nationality: Turkish
Birth Place and Date: Ankara, 17.10.1986

Education
Ph.D. : 2011-, Ege University, International Computer Institute, Izmir, Turkey
M.Sc. : 2008-2011, Ege University, International Computer Institute, Izmir, Turkey
B.Sc. : 2004-2008, Pamukkale University, Computer Engineering, Denizli, Turkey

Foreign Languages
Turkish : First Language
English : Advanced

Programming Skills
C/C++, C#, Java, JavaScript, Python, Matlab

Projects

- 2015- : The Integration of ALTO protocol and SDN Technology for Maximizing Quality of Experience in Video Streaming Applications (TÜBİTAK: 115E449)

- 2015-2017 : Improvement of Cloud-based Data Services using Software Defined Networking Architecture (Ege University: 2015-UBE-002)

- 2014-2015 : Dynamic Adaptive HTTP Video Streaming over Software Defined Networks (TÜBİTAK: 114E409)

- 2012-2014 : Automatic Extraction of Vascular Landmarks for Monitoring Temporal Differences in Retinal Images Using Fuzzy RBF (Ege University: 2012-UBE-002)

- 2011-2013 : Development of an Agent based Alternative Path Selection Method for Peer-to-Peer Video Streaming Applications (TÜBİTAK: 111E022)

Publications

R. Shokri, C. Cetinkaya, M. Sayit, 2017. Design of A Layer-based Video Streaming System over Software-Defined Networks. *to appear in 2017 8th International Conference on the Network of the Future (NoF17), London, United Kingdom, 22-24 Nov. 2017.*

K. Herguner, R. Shokri, C. Cetinkaya, M. Sayit, 2017. Towards QoS-aware Routing for DASH Utilizing MPTCP over SDN. *to appear in 2017 3rd IEEE Conference on Network Function Virtualization and Software Defined Network (NFV-SDN'17), Berlin, Germany, 6-8 Nov. 2017.*

S. Bikas, C. Cetinkaya, M. Sayit, 2017. Evaluation of MPTCP Congestion Control for DASH. *In proceedings of the 2017 IEEE 7th International Conference on Consumer Electronics-Berlin (ICCE-Berlin 2017), Berlin, Germany, 3-6 Sep. 2017.*

S. Ozcan, T. Kivilcim, C. Cetinkaya, M. Sayit, 2017. Rate Adaptation Algorithm with Backward Quality Increasing Property for SVC-DASH **(Distinguished Paper Award).** *In proceedings of the 2017 IEEE 7th International Conference on Consumer Electronics-Berlin (ICCE-Berlin 2017), Berlin, Germany, 3-6 Sep. 2017.*

C. Cetinkaya, M. Sayit, 2017. ALTO-assisted CDN-based video streaming over SDN. *In proceedings of the 2017 IEEE 25th Signal Processing and Communications Applications Conference (SIU), Antalya, Turkey, 16-19 May 2017.*

C. Cetinkaya, M. Sayit, 2016. Video-on-Demand System Architecture with ALTO-SDN Integration **(STG Award)**. *In proceedings of the IEEE 2016 International Black Sea Conference on Communications and Networking (BlackSeaCom), Varna, Bulgaria, 6-9 Jun 2016.*

E. Kaysudu, C. Cetinkaya, K. Herguner, M. Sayit, 2017. Server Selection For

Video Streaming Applications Over Software Defined Networks. *In proceedings of the 2016 IEEE 24th Signal Processing and Communications Applications Conference (SIU), Zonguldak, Turkey, pp. 1965-1968, 16-19 May 2016*.

T. Uzakgider, C. Cetinkaya, M. Sayit, 2015. Learning-based Approach for Layered Adaptive Video Streaming over SDN. *Computer Networks, Volume 92, Part 2, 9 December 2015, Pages 357-368, ISSN 1389-1286*

C. Cetinkaya, Y. Ozveren, M. Sayit, 2015. An SDN-assisted System Design for Improving performance of SVC-DASH. *In proceedings of the 2015 Federated Conference on Computer Science and Information Systems (FedCSIS), Łódź, Poland, pp. 819-826, 13-16 Sept. 2015*.

C. Cetinkaya, Y. Ozveren, M. Sayit, 2015. Route Optimization for DASH over Software Defined Networks. *In proceedings of the 2015 IEEE 23rd Signal Processing and Communications Applications Conference (SIU), Malatya, Turkey, pp. 2349-2352, 16-19 May 2015*.

Y. Ozveren, C. Cetinkaya, M. Sayit, 2015. Performance Evaluation of DASH Rate Adaptation Algorithms. *In proceedings of the 2015 IEEE 23rd Signal Processing and Communications Applications Conference (SIU), Malatya, Turkey, pp. 1618-1621, 16-19 May 2015*.

C. Cetinkaya, M. Sayit, E. Karayer, C. Hellge, 2014. SDN for Segment based Flow Routing of DASH **(Best Paper Award)**. *In proceedings of the 2014 IEEE 4th International Conference on Consumer Electronics (ICCE-Berlin), Berlin, Germany, pp. 74-77, 9-11 Sept. 2014*.

M. Sayit, E. Karayer, K. D. Teket, Y. Kaymak, C. Cetinkaya, S. Demirci, G. Kardas, 2013. A Score-based Packet Retransmission Approach for Push-Pull P2P Streaming Systems. *In proceedings of the 2013 Federated Conference on Computer Science and Information Systems (FedCSIS), Krakow, Poland, pp. 627-633, 8-11 Sept. 2013*.

M. Sayit, Y. Kaymak, K. D. Teket, C. Cetinkaya, S. Demirci, G. Kardas, 2013. Parent Selection via Reinforcement Learning in Mesh-based P2P Video Streaming. *In proceedings of the 2013 10th International Conference on Information Technology: New Generations (ITNG), Las Vegas, Nevada, USA, pp. 546-551, 15-17 April 2013*.

C. Candemir, C. Cetinkaya, O. Kilincceker, M. Cinsdikici , 2013. Vascular landmark classification in retinal images using fuzzy RBF . *In proceedings*

of the 2013 IEEE 21st Signal Processing and Communications Applications Conference (SIU), Cyprus, pp. 1-4, 24-26 April 2013.

Awards and Grants

- (2017) Distinguished Paper Award, ”Rate Adaptation Algorithm with Backward Quality Increasing Property for SVC-DASH”, ICCE-Berlin 2017
- (2016) Student Travel Grant, ”Video-on-Demand System Architecture with ALTO-SDN Integration”, BlackSeaCom 2016
- (2015) TUBITAK ULAKBIM International Scientific Publication Promotion
- (2014) Best Paper Award, ”SDN for Segment based Flow Routing of DASH”, ICCE-Berlin 2014
- (2013) TUBITAK BIDEB 2211-C Ph.D. Scholarship (for 3 years)