VSP-Managed DASH Video Services

by

Kemal Emrecan Sahin

A Thesis Submitted to the
Graduate School of Sciences and Engineering
in Partial Fulfillment of the Requirements for
the Degree of

Master of Science

in

Electrical and Electronics Engineering

Koç University

03.08.2017

Koç University

Graduate School of Sciences and Engineering

This is to certify that I have examined this copy of a master's thesis by

Kemal Emrecan Sahin

and have found that it is complete and satisfactory in all respects,
and that any and all revisions required by the final
examining committee have been made.

Committee Members:

_____

Prof. Murat Tekalp(Advisor)

_____

Assoc. Prof Öznur Özkasap

_____

Prof. Fatih Alagöz

Date: _____

Dedicated to my family and friends...

ABSTRACT

Dynamic adaptive streaming over HTTP (DASH) clients compete with each other over one or more bottleneck links in a network, which results in fluctuations in TCP throughput and QoE, QoE unfairness among clients, and under utilization of the network capacity. We propose both centralized and distributed architectures for collaboration between network service provider (NSP), video service provider (VSP), and users (DASH clients) to provide DASH service over software-defined networks (SDN) with quality-of-service (QoS) reserved network slices. We show that QoS reservation alone is not sufficient to overcome QoE fluctuations per client and unfairness between clients, and clients also need to employ TCP receive window adaptation knowing their fair-share bitrate. To this effect, we propose two collaboration architectures to inform clients about their fair-share bitrates. We first present a centralized collaboration architecture between the NSP, VSP, and the users, where VSP assigns a fair-share bitrate to each DASH client. We then present a distributed collaboration architecture, where a group of DASH clients sharing a reserved network slice collaborate among themselves. The collaboration groups are identified by the VSP using information provided by the NSP, and we propose a protocol that allows clients within a group to share critical parameters with each other so that each client can estimate its fair-share bitrate in a distributed manner. We show that collaboration rather than competition between clients also improves utilization of the reserved capacity in addition to avoiding throughput fluctuations and achieving a smooth QoE. Experimental results demonstrate the superior performance of both types of collaboration over competition between the clients.

ÖZETÇE

HTTP üzerinden dinamik adaptif iletimi (DASH) kullanıcıları, bir ağdaki bir veya daha fazla tıkanık bağlantıda birbirleriyle rekabet eder; bu da, TCP akışındaki verimlilik ve QoE' de dalgalanmalara, kullanıcılar arasında QoE adeletsizliği ve ağ kapasitesinin kullanımı oranının düşük bir seviyede kalmasına neden olur. Yazılım tanımlı ağlar üzerinden (SDN) kalite güvencisi (QoS) için ayrılmış ağ dilimleri ile DASH hizmeti sağlamak için ağ servis sağlayıcısı (NSP), video servis sağlayıcı (VSP) ve kullanıcılar (DASH) arasında bir işbirliği için merkezi ve dağıtık mimarileri önermekteyiz. Kullanıcı başına QoE dalgalanmalarını ve kullanıcılar arasındaki adeletsizliği gidermek için tek başına QoS ayırmanın yeterli olmadığını ve müşterilerin adil paylaşım veri hızını bilen TCP alma pencere adaptasyonunu kullanmaları gerektiğini gösteriyoruz. Bu amaçla, müşterilerimize adil paylaşımlı veri hızları hakkında bilgi vermek için iki işbirliği mimarisi önermekteyiz. Birinci olarak NSP'de bir trafik mühendisliği yöneticisinin (TEM) her DASH kullanıcısına adil bir paylaşım veri hızı atadığı NSP, VSP ve kullanıcılar arasında merkezi bir işbirliği arşivi sunmaktayız. İkinci olarak, rezervasyon yapışmış bir ağ dilimi paylaşan bir grup DASH kullanıcısının kendi aralarında işbirliği yaptığı dağıtık bir işbirliği mimarisi sunmaktayız. İşbirliği grupları, NSP tarafından sağlanan bilgileri kullanarak VSP tarafından tanımlanır ve bir kullanıcının bir grup içindeki kullanıcılarla kritik parametreleri paylaşmalarını sağlayan bir protokol önermekteyiz; böylece her kullanıcı, adil paylaşım veri hızını dağıtılan bir biçimde tahmin edebilir. DASH kullanıcıları arasındaki rekabet yerine işbirliği yaptığımızda, TCP akışındaki verimlilik dalgalanmalarından kaçınarak ve sorunsuz bir QoE elde edilmesiyle birlikte rezerve edilmiş kapasitenin kullanımını daha verimli hale geldiğini de gösteriyoruz. Deneysel sonuçlar, iş birliği yapan kullanıcıların rekabet halindeki kullanıcılara göre daha üstün performans sahip olduğunu göstermektedir.

ACKNOWLEDGMENTS

I would like to acknowledge my advisor, Prof. Dr. A. Murat Tekalp for his excellent advisory, inspiration and reliable guidance over the course of graduate studies. I am also grateful of my thesis committee Assoc. Prof. Öznur Özkasap and Prof. Fatih Alagöz for their valuable comment and feedbacks.

First and most, I would like to thank Tolga who always helped and provided his elegant guidance, inspiration and endless patience for academic life. From first day of my graduate studies to the final day, he supported me with his knowledge. He was like a elder brother to me, and he made everything including my thesis, my objective for my major clear, confident, and enthusiastic.

I also thank to all of my lab mates Seyhan, Tuğtekin, Selin, Gonca, Berker, Huseyin for their assistance and sharing during my graduate studies. I owe a debt of gratitude to all my friends making Koç irreplaceable: Katel, Çurkup, Fuslu, Rcicen, Efe, Okirnap, Bozkaranfil, Umutcan, Hazal, Sinan, Genco, Dorukcan, Can, Tuan, Mert, Göksu K., and Sena G.. I also would like to thank TUBITAK for providing financial support for my research.

Finally, I would like to thank my mother Sergül, my father Aytekin and my brother Sercan for their love, patience and support. My lovely thanks goes to Goksu for her invaluable support and patience during my studies.

TABLE OF CONTENTS

List of Tables **ix**

List of Figures **x**

Chapter 1: Introduction **1**

1.1 Motivation 1

1.2 Contributions and Organization 4

Chapter 2: Review of SDN and DASH **6**

2.1 Review of SDN 6

2.2 Review of DASH 8

2.2.1 DASH Architecture 8

2.2.2 Manifest File 10

Chapter 3: Interaction Between Resource Reservation and TCP **12**

3.1 Resource Reservation 12

3.2 TCP Rate Control 12

3.3 Interaction between Resource Reservation and TCP 13

3.4 Queue Management for VSP-Managed Video Services 15

Chapter 4: VSP-Managed Centralized Video Service **17**

4.1 System Architecture 17

4.2 Slice Management by VSP 18

4.2.1 Determining Fair-share Bitrates for Different Video Resolutions 20

4.2.2 Sort the Slice List According to Congestion Ratio 20

4.2.3 Computation of Fair-Share Bitrates 21
4.3 Fair-Share Bitrate Algorithm 22
4.4 Collaborative DASH Implementation of Centralized Video Service . . 23
4.5 VSP-Client Collaboration 24
4.6 TCP Rate Control by Clients 25

Chapter 5: VSP-Managed Distributed Video Service 26
5.1 System Architecture 26
5.2 Slice Management by VSP 28
5.3 Interclient Collaboration and Fair-Share Bitrate Algorithm 30
5.4 Collaborative DASH Implementation of Distributed Video Service . . 32
5.5 VSP-Client Collaboration 33
5.6 TCP Rate Control by Clients 34

Chapter 6: Dynamic Adaptive Video Streaming Evaluations 35
6.1 Evaluation Environment 35
6.2 Overlay Network Description 36
6.3 Evaluation of DASH over Collaborative Architectures 37
6.3.1 Results 37

Chapter 7: Conclusion 55

Bibliography 57

Vita 62

LIST OF TABLES

4.1 Coefficients of the video quality function [18]. 20

6.1 Resolutions and encoding parameters for test video content. 37
6.2 Comparison for four homogeneous 1080p clients. 39
6.3 Comparison for three heterogeneous clients using 'Tears of Steel' with 4 sec. segments. 41
6.4 Comparison for three heterogeneous clients using 'Big Buck Bunny' with 6 sec. segments. 43
6.5 Comparison for three heterogeneous clients using 'Tears Of Steel' with 6 sec. segments. 44
6.6 Comparison when clients enter/exit randomly. 45
6.7 The Result Of 10 Heterogeneous Clients Sharing a 17 Mbps Network Slice with streaming 'Big Buck Bunny' 46
6.8 The Result Of 10 Heterogeneous Clients Sharing a 21 Mbps Network Slice with streaming 'Tears of Steel' 47
6.9 Comparison for 12 heterogeneous clients sharing a 21.5 Mpbs network slice 50
6.10 Three heterogeneous clients streaming Big Buck Bunny with 4 sec. segments over a variable capacity slice 51
6.11 Three heterogeneous clients streaming Tears Of Steel with 4 sec. segments over a variable capacity slice 51
6.12 Analysis of the effect of DASH segment duration. 54

LIST OF FIGURES

2.1 SDN Architecture. 6

2.2 Overall DASH Architecture. 8

2.3 DASH Reference Architecture. 9

2.4 Sample MPD File [31] 11

3.1 Instantaneous TCP rates for default TCP vs. TCP rate control: a) sender and b) receiver rates for a single standard TCP flow in a reserved 5 Mbps network slice, c) sender and d) receiver rates for a single TCP flow with rate control in a reserved 5 Mbps slice, e) sender and f) receiver rates for four standard TCP clients in a reserved 20 Mbps slice, g) sender and h) receiver rates for four TCP clients with rate control in a reserved 20 Mbps slice. 14

4.1 System architecture for centralized collaboration. 18

4.2 Mapping between video quality and bit-rate according to three types of display resolution [18] 21

4.3 Collaborative DASH architecture 24

4.4 VSP to DASH client message 25

5.1 System architecture for distributed collaboration. 27

5.2 Grouping 28

5.3 Collaborative DASH architecture 32

5.4 VSP to DASH client message 33

6.1 The sample built-in Mininet topology 35

6.2 Exemplary network used for evaluations. 36

6.3 Single realizations of competitive vs. collaborative streaming results for three heterogeneous clients using Tears of Steel: a) competitive, b) VSP-managed centralized collaboration, c) VSP-managed distributed collaboration. 40

6.4 Comparison of competitive, centralized-collaborative, and distributed-collaborative systems for three heteregeneous clients: a) competitive, b) VSP-managed centralized collaboration, c) VSP-managed distributed collaboration clients streaming Big Buck Bunny; d) competitive, e) VSP-managed centralized collaboration, f) VSP-managed distributed collaboration clients streaming Tears of Steel. 42

6.5 Single realizations of competitive vs. collaborative streaming results when clients enter/exit randomly: a) competitive, b) VSP-managed central collaboration, c) VSP-managed distributed collaboration. . . . 45

6.6 Comparison of competitive and collaborative systems for ten heterogeneous clients: a) competitive, b) VSP-managed distributed collaboration clients streaming Big Buck Bunny 46

6.7 Comparison of competitive and collaborative systems for ten heterogeneous clients: a) competitive, b) VSP-managed distributed collaboration clients streaming Tears Of Steel 48

6.8 Single realizations of competitive vs. collaborative streaming results for 12 heterogeneous clients: a) competitive, b) VSP-managed distributed collaboration. 49

6.9 Comparison of competitive and collaborative DASH over a variable capacity slice: a) competitive and b) collaborative clients streaming Big Buck Bunny; c) competitive and d) collaborative clients streaming Tears Of Steel 52

6.10 Convergence of video quality in competitive vs. collaborative streaming as the slice capacity increases. 54

Chapter 1

INTRODUCTION

1.1 Motivation

We are witnessing increasing consumption of Hyper-Text Transfer Protocol (HTTP) adaptive video streaming over the Transmission Control Protocol (TCP) on the open (unmanaged) Internet. However, the best-effort nature of the Internet results in a varying degree of unpredictable end-to-end (e2e) quality of service (QoS), which is characterized by unpredictable variations in TCP throughput and packet loss rates. This in turn causes unsatisfactory quality of experience (QoE) for end-users (clients), because of video stalls (due to client play-out buffer drainage) and frequent video quality variations (triggered by TCP throughput fluctuations).

In HTTP adaptive streaming (HAS), the video is encoded in fixed-duration segments that are made available at the video server at multiple bitrates (quality levels), and clients request video segments at a bitrate which best fits their play-out-buffer status and an estimate of available network throughput. Segment durations vary between 2-15 secs, but are fixed and synchronized for multiple representations of the same video at different bitrates. The MPEG dynamic adaptive streaming over HTTP (DASH) standard specifies the format of video content and associated manifest file so that different HAS players can play the same content [1]. While DASH overcomes video-stall events in most cases, throughput fluctuations per transmission control protocol (TCP) flow cause frequent variations in requested DASH video quality level (source bitrate) by clients, which result in reduced end-user QoE.

Since DASH uses TCP transport, there are two separate flow/rate control mecha-

nisms applied at two different levels: (i) TCP congestion/flow control at the transport level of the network, and (ii) DASH video rate adaptation by the client at the application level. These two rate control mechanisms are unaware of each other, and it is well-known that they poorly interact especially in the presence of network congestion [2]. Standard TCP congestion control mechanism and its variants are designed based only on feedback between end-points that is unaware of other competing traffic, and the network service provider (NSP) plays no role in adjusting the TCP behavior. As a result, DASH flows competing for a specific bottleneck link throughput (i) experience unstable bitrates per flow (known as QoE variability), (ii) result in unfair distribution of link capacity between competing flows (leading to QoE unfairness), (iii) cause underutilization of the bottleneck link capacity. An evaluation of three TCP-based rate-control schemes for adaptive streaming using scalable H.264/SVC video coding in terms of the quality of delivered video has been conducted in [3]. However, TCP congestion control is generally not effective in dealing with the problem of competing DASH flows, unless the available link capacity is significantly larger than the total video bitrate.

QoE refers to the quality of video delivered to users, which clearly is related to the network QoS, besides other factors [4]. Analysis of all factors affecting QoE in DASH video can be found in [5]. Metrics for measuring user QoE are discussed in [6]. QoE evaluation of different bitrate adaptation logics has been presented in [7]. Video quality is often measured in terms of PSNR or the structural similarity metric [8]. It is well-known that equivalent video quality (QoE) can be observed at different bitrates for different video (display) resolutions. This means users receiving video at HD resolution needs a higher bitrate than users receiving video at SD or lower resolutions. Instead of sharing bitrate equally between multiple DASH clients, allocating bitrates to clients such that users with different video resolutions experience almost equal QoE is referred to as QoE fairness. Achieving QoE fairness between multiple DASH clients requires some kind of collaboration between the NSP, video service provider (VSP) and users.

We can classify related works for achieving QoE-fairness between competing TCP flows as i) centralized, NSP-enforced solutions, ii) VSP-assisted solutions [9], and iii) features built-into the client player [10]. Centralized NSP-enforced approaches include i) traffic shaping at the network edges by ACK-pacing or active window management (AWM), and ii) network-wide solutions involving SDN and QoS provisioning. Since the TCP congestion window cannot be larger than the receiver window size *rwnd*, the network operator can perform traffic shaping by controlling the downstream flow rate (send rate) for each flow by either manipulating *rwnd* field in ACK headers at an appropriate router or buffering ACK messages at routers along congested paths to delay them, and hence, slow down senders in a fair manner. The load-adaptive ACK pacer [11] adapts to aggregate traffic fluctuations at a router, rather than per flow state as in [9], and triggers ACK pacing when queue fullness exceeds a threshold. It has been demonstrated that AWM, activated based on queue fullness in a router, achieves QoE fairness among multiple competing DASH streams in a DVB-T broadcast network [12]. The AWM method has also been implemented at the access router of the VSP, where computation of *rwnd* relies only on local data available within the VSP router [9]. Traffic shaping at home gateways has also been proposed to provide QoE-fairness between multiple competing heterogeneous DASH flows in home networks [13][14].

Future networks deploying SDN will make e2e QoS provisioning more practical and feasible since the entire topology of the network is visible to an SDN controller [15], which makes localization and solution of the congestion and QoE-fairness problems more accurate. Approaches to provision e2e QoS over SDN has been surveyed in [16]. Methods to provide QoS to DASH flows include (i) metering (policing) flows at select switches, (ii) use ACK pacing techniques (traffic shaping) for flow-based rate control [9], and (iii) allocating and managing special queues for specific flows [17].

Related works that employ SDN to offer QoS to DASH flows in order to achieve network-wide QoE fairness include [18] [19] [21][22][23] [24]. These methods typically involve a central optimization problem to allocate resources to different DASH flows. The complexity of the optimization is influenced by the number of model parameters

and the number of client flows. Georgopoulos et al. [18] propose an OpenFlow-assisted QoE fairness framework (QFF) that monitors the status of all DASH applications in a network and dynamically allocates fair network resources to each client. Nam et al. [19] propose an SDN application to monitor network and dynamically change routing paths using multi-protocol label switching (MPLS) traffic engineering (TE) to provide better QoE performance. Liotou et al.[20] introduce a QoE-Service module using TE over a SDN to enhance the performance of OTT applications. Ramakrishnan et al. [21] propose an SDN-based VQOA module that implements multi-client bandwidth allocation optimization for video rate selections among competing clients in order to increase overall QoE performance of the system. Petrangeli et al. [22] propose QoE-driven rate adaptation heuristic for fair adaptive video streaming. Kleinrouweler et al. [23] develop an SDN architecture for a DASH Service Manager, that signals desired bitrates to DASH clients to adapt and perform TE to overcome the negative effect of TCP. Mu et al. [24] introduce a user-level fairness model, UFair, and its hierarchical variant, UFairHA, which orchestrate DASH media streams using SDN, and incorporate three fairness metrics (video quality, switching impact, and cost efficiency) to achieve user-level fairness. However, none of these works provide an analysis of the interaction between QoS resource reservation and TCP congestion control mechanism.

1.2 Contributions and Organization

The three way interaction between QoS (throughput reservation) at network level and TCP congestion control at transport level and DASH quality level adaptation at the application level has not been studied before. It turns out that resource reservation alone is not sufficient to optimize QoE variability, fairness, and network utilization in DASH video delivery because TCP flows compete for more bandwidth due to nature of TCP congestion control. In this thesis, we mainly propose rate controlled Dynamic Adaptive Video Streaming over Future Networks. The main contributions of this thesis can be summarized in the following three categories:

- We propose centralized and distributed collaboration architectures involving

NSP, VSP and clients to achieve fairness in QoE among competing heterogeneous device flows.

- Minimizing the QoE fluctuations per flow due to competition among heterogeneous flows and the mismatch between Transmission Control Protocol and Dynamic Adaptive Streaming over HTTP applications, and we evaluate the QoE fairness and fluctuations of these architectures for different test scenarios. The results are presented from the user's perspective.

- Evaluating the performance of proposed architectures and standard DASH player under different test scenarios, such as varying bottleneck links, different number of competing clients, and dynamic change in the number of competing clients.

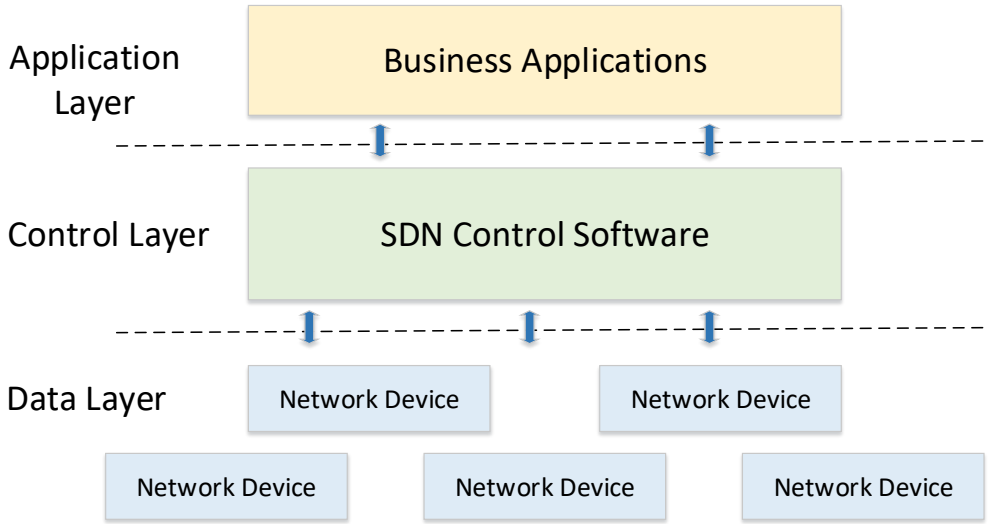
The rest of the thesis is organized as follows: Chapter 2 gives brief information about the SDN and DASH technologies. Chapter 3 covers the interaction between resource reservation and TCP to apply rate control when the TCP rate reaches the rate reserved for that flow in order to properly manage QoE variability and network utilization. Chapter 4 explains the proposed centralized architecture between the NSP, VSP and clients to inform clients about their fair-share bitrates. Chapter 5 explains the proposed distributed architecture between NSP, VSP and clients to perform inter-client collaboration about their fair-share bitrates. Chapter 6 extensively evaluates the QoE fluctuations per flow, QoE fairness among flows and the utilization of the allocated throughput. Finally, we draw our conclusions in Chapter 7.

Chapter 2

REVIEW OF SDN AND DASH

2.1 Review of SDN

Software defined networking illustrated in Figure 2.1 is an innovative technology that provides several solutions the scalability, security, and speed issues of conventional networks. SDN splits the data and control plane into two different planes rather than conventional network in which these planes are bundled. SDN increasing the flexibility and evolution of the network infrastructure since the control plane, controller, can send commands to the routers or switch that are the elements of the data plane without performing any device-centric configuration. [25] SDN has ability to provide high speed transmissions and more efficient resource usage, and the network characteristics can be controlled remotely. SDN concept is easy to implement data centers and data cloud applications for network control ability.



Şekil 2.1: SDN Architecture.

The control mechanism of the SDN is provided with communication between switch and controller using protocols such as OpenFlow. Each switch can have several flow tables, a group table and an OpenFlow channel. Initially each switch has empty tables with some fields, and configured by controller. These configured tables contains the rules for each packet visiting switches, there exist some packet fields (source/destination IP, port numbers...) and defined actions (dropping, forwarding, modifying...). [26]

In general, the main application areas of SDN are (i) internet research, (ii) rural connections, (iii) data centers upgrading, (iv) mobile device offloading, and (v) wireless virtual machines.[27] The seperation between control and data layer allows for experimenting with new addressing, non-standard protocols for new internet architectures. Since SDN simplifies the complex data center and networks, it can also simplify rural networks with using the central controller to fix the issues of the rural environments including small profit margins, and resource constraints. Protocols, OpenFlow etc..., allows data center companies to save money in configuring networks since the central controller can manage switches, and the maintenance become quick and low-priced. The mobile device offloading is provided by the SDN if the security requirements are available, and the controller can regulate the flows. Since the trend of running applications on VM in businesses are increasing, and these VMs allow the companies to have lower operational costs. However, the maintenance of the VM's IP address is not efficient with current methods so that SDN has the potential of solving the mobility issues related IP address handling of VMs.

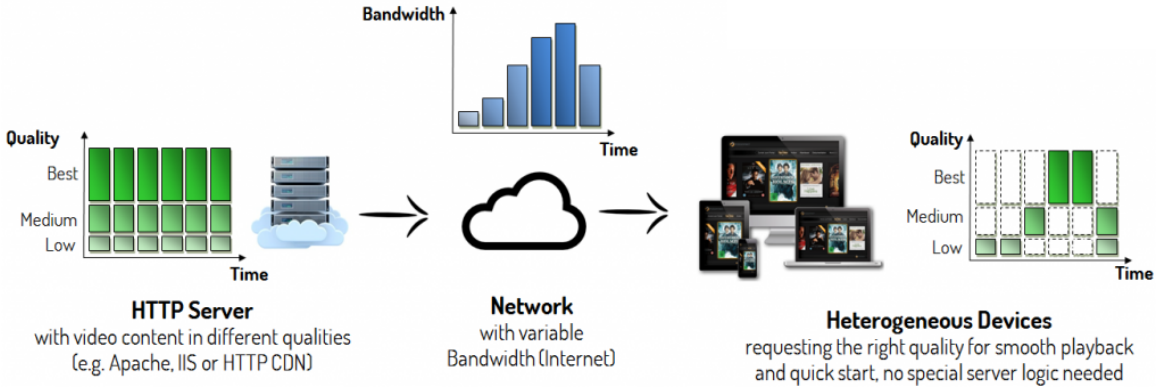
In addition to these application areas, SDN provides traffic engineering solutions to internet traffic flows, and the majority of the traffic flows are video streaming packets which is accounted for approximately 70 percent of all consumer internet traffic. The video traffic flows need to satisfy some requirements related to some maximum delay, minimum quality, etc... so that the quality of service is important. In contrast to conventinal networks, SDN can provide fine-granularity QoS support since protocols like OpenFlow can control packet and flow level data delivery successfully so that

the individual or group flows can be handled in a way that achieves required QoS support. In general, SDN protocols adjust the flow delivery rules, and perform traffic engineering using QoS optimization models in accordance to these rules.

2.2 Review of DASH

2.2.1 DASH Architecture

Dynamic Adaptive Video Streaming (DASH) is a technique that enables efficient and high quality video content streaming in an end-to-end service over the Internet using Transmission Control Protocol [28]. In DASH, the media content is divided into a sequence of small HTTP-based segments containing a finite interval of playback time of this content [29][30]. The content server is HTTP web servers such as Apache, IIS, nginx, GWS. The mechanism of DASH is illustrated in figure 2.2. The reason of using HTTP as delivery protocol for DASH can be itemized below:



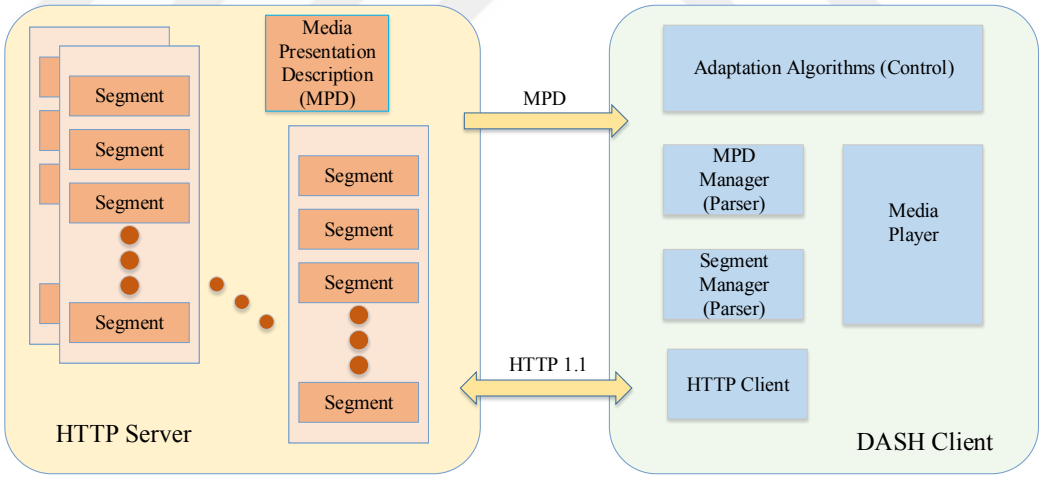
Şekil 2.2: Overall DASH Architecture.

- HTTP is a widely used form of delivery of the Internet video.
- The trend of using HTTP as the main protocol for multimedia delivery over the Internet
- The reliability and deployment simplicity of HTTP and the widely deployment of TCP/IP protocol

- The client is able to control the entire streaming session since the client can open one or several TCP connections and standard HTTP servers.

- The client can choose initial content rate without requiring the negotiation with the content server.

In general, the media content has the segment size that varies between 2-15 secs, and is made available at multiple bitrates(adaptation levels) at the content server. The intelligence of video streaming moves from network to client that enables client differentiation, and the client selects the suitable adaptation level of content based on network, user preferences and device capacity. The overall DASH server and client architectures are illustrated in Fig 2.3.



Şekil 2.3: DASH Reference Architecture.

Since DASH is an adaptive HTTP streaming technology standardized by MPEG,there are also different adaptive HTTP streaming technologies such as Adobe HTTP Dynamic Streaming(HDS), Apple HTTP Live Streaming(HLS),and Microsoft Smooth Streaming(MSS). However, MPEG DASH is the only international standard.

According to media content segment size, different resolution, bandwidth levels, and codec informations, there can be different representations such as multiple screen sizes and bandwidths for a media content. These different representations allows

clients to choose the highest possible quality level without experiencing stall related events or wasting bandwith on necessary pixels such that a SD TV does not need HD content. Moreover, different browsers can support different codecs like MPEG-4 AVC/h.264 or VP8 so that clients are able to request media contents with suitable codecs. In general, representations are chosen by built-in DASH players, but clients are able to override the representations in some DASH players.

2.2.2 Manifest File

Media Presentation Description (MPD) file provides the duration of the media content, frame rate per second, resolution, bandwidth levels, and codec information demonstrated in Fig 2.4. At the beginning of the streaming session, client obtains MPD file from the content server. According to this file, DASH client to choose suitable segment during media content streaming session. In Fig 2.4, the duration of content is given by mediaPresentationDuration parameter at the second line of MPD file. The title of the content is defined in the title container at the sixth line of MPD file. Then, the AdaptationSet parameter contains the information about frame rate, maximum available resolution at the tenth and eleventh line of MPD file. Finally, each representation is given by a unique id with required bandwidth levels and codec information at each representation container.

```
<MPD xmlns="urn:mpeg:dash:schema:mpd:2011" minBufferTime="PT1.500000S" type="static"
mediaPresentationDuration="PT0H12M14.17S" profiles="urn:mpeg:dash:profile:isoff-
live:2011">
  <ProgramInformation moreInformationURL="http://gpac.sourceforge.net">
    <Title>
      dashed/TearsOfSteel_6s_simple_2014_05_09.mpd generated by GPAC
    </Title>
  </ProgramInformation>
  <Period duration="PT0H12M14.17S">
    <AdaptationSet segmentAlignment="true" maxWidth="1920" maxHeight="1080"
    maxFrameRate="24" par="16:9">
      <Role schemeIdUri="urn:mpeg:dash:role:2011" value="main"/>
      <SegmentTemplate timescale="24000"
      media="tos_$Bandwidth$bps/TearsOfSteel_6s_$Number$.m4s" startNumber="1"
      duration="144000" initialization="tos_$Bandwidth$bps/TearsOfSteel_6s__init.mp4"/>
      <Representation id="1920x1080 10.0Mbps" mimeType="video/mp4" codecs="avc1.4d4029"
      width="1920" height="1080" frameRate="24" sar="1:1" startWithSAP="1"
      bandwidth="10036520"/>
      <Representation id="1920x1080 6.0Mbps" mimeType="video/mp4" codecs="avc1.4d4028"
      width="1920" height="1080" frameRate="24" sar="1:1" startWithSAP="1"
      bandwidth="6017652"/>
      <Representation id="1920x1080 4.0Mbps" mimeType="video/mp4" codecs="avc1.4d4028"
      width="1920" height="1080" frameRate="24" sar="1:1" startWithSAP="1"
      bandwidth="4004949"/>
      <Representation id="1920x1080 3.0Mbps" mimeType="video/mp4" codecs="avc1.4d4028"
      width="1920" height="1080" frameRate="24" sar="1:1" startWithSAP="1"
      bandwidth="2998443"/>
      <Representation id="1280x720 2.4Mbps" mimeType="video/mp4" codecs="avc1.4d401f"
      width="1280" height="720" frameRate="24" sar="1:1" startWithSAP="1"
      bandwidth="2412364"/>
      <Representation id="1280x720 1.5Mbps" mimeType="video/mp4" codecs="avc1.4d401f"
      width="1280" height="720" frameRate="24" sar="1:1" startWithSAP="1"
      bandwidth="1503860"/>
      <Representation id="640x360 806.0kbps" mimeType="video/mp4" codecs="avc1.4d401e"
      width="640" height="360" frameRate="24" sar="1:1" startWithSAP="1"
      bandwidth="805931"/>
      <Representation id="640x360 504.0kbps" mimeType="video/mp4" codecs="avc1.4d401e"
      width="640" height="360" frameRate="24" sar="1:1" startWithSAP="1"
      bandwidth="503541"/>
      <Representation id="480x270 253.0kbps" mimeType="video/mp4" codecs="avc1.4d4015"
      width="480" height="270" frameRate="24" sar="1:1" startWithSAP="1"
      bandwidth="252760"/>
```

Şekil 2.4: Sample MPD File [31]

Chapter 3

INTERACTION BETWEEN RESOURCE RESERVATION AND TCP

3.1 Resource Reservation

A network slice is a set that contains varying number of routing paths. Due to limited information that VSP can share with NSP, NSP can perform three possible traffic management policies to provide a network slice. First, NSP determines the optimal routes and resource allocations of VSP clients. According to these resource allocations, NSP can provide a network slice to VSP. Second, NSP can perform best effort routing or the third apporach is using some heuristic methods to determine routes of VSP clients. In second or third case, VSP shall manage its clients, and request a network slice in accordance with the information obtained from NSP.

3.2 TCP Rate Control

Transmission Control Protocol is one of the most important protocols that provides connection-oriented service in the Internet [32][33]. This service is responsible for guaranteed transportation of application-layer messages, and flow control at the Internet's transport layer. The flow control between source and destination is controlled by data transmission/receive window. These data windows are modified in accordance with the acknowledgements (ACKs), duplicate ACKs, and timeouts. The source adjusts the size of its congestion window that limits the number of unacknowledged packets outstanding in the network. As the number of ACKs of successfully transmitted data increases, the source increases its transmission rate by increasing its congestion window. Since the network has a finite capacity, the transmission rate of

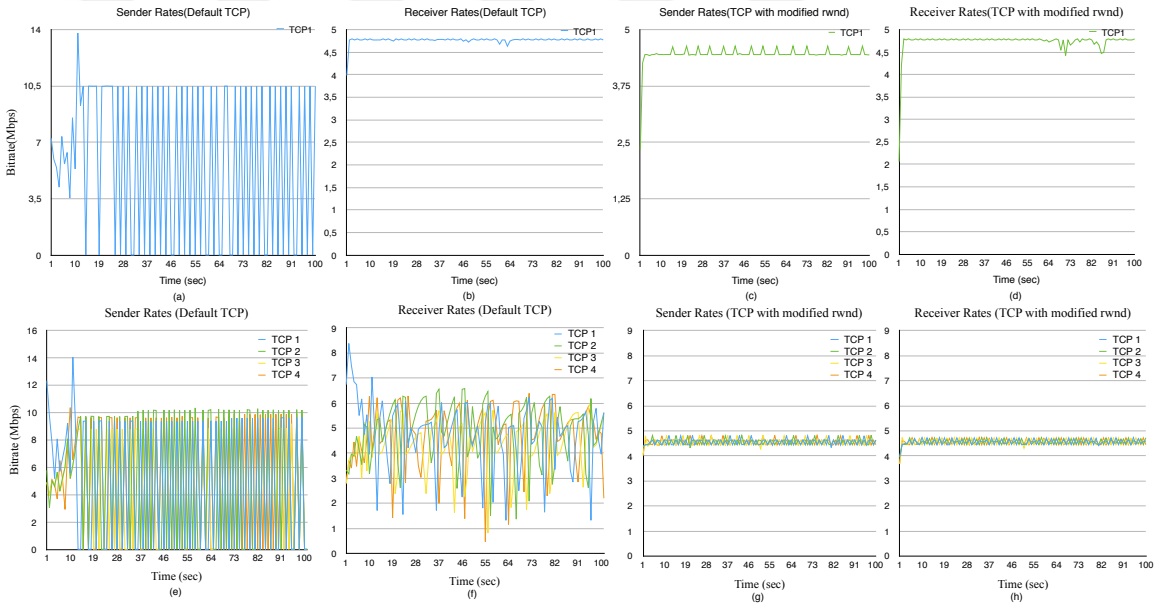
source can exceed the capacity of the network at some point, and the packets will be dropped at routers in the network. Due to the packet loss, TCP reduces its congestion window(transmission rate).

3.3 Interaction between Resource Reservation and TCP

In scheduling multiple DASH streams over a resource-reserved slice, with no client-side TCP receive-window (rwnd) management from the application, there is three-way interaction between network resource reservation, TCP rate control and application level DASH video rate adaptation algorithm, where standard methods for application level DASH rate adaptation are based on the TCP goodput estimated by the client. The main goal of this section is to show that we can stabilize TCP goodput per client by enabling client-side rwnd control, provided that the slice capacity does not change in time and there are no entry/exit of clients. Under these conditions, the three-way interaction can essentially be considered as a two-way interaction since the TCP goodput would not vary. However, when the NSP cannot guarantee a fixed rate to a customer/client or a fixed slice capacity to a VSP (soft QoS) and/or there are entry/exit of clients, TCP goodput per client can still change in time and we again have three-way interaction but in a much more controlled manner compared to basic competitive streaming, since with the knowledge of their fair-share bitrate, clients can now set their DASH video rate adaptation and TCP receive-window size in a consistent manner.

We now analyze the interaction between resource reservation at the network level and TCP congestion control at the transport level. Suppose we reserve 20 Mbps slice in a network and route 4 homogeneous DASH streams over this reserved capacity. Then, the fair share for each DASH flow would be 5 Mbps. It turns out that network resource reservation alone is not sufficient to achieve smooth throughput around 5 Mbps, hence, smooth QoE, for these DASH streams because TCP is unaware of the reserved bitrate, and the flows continue to compete for more bandwidth due to nature of TCP congestion control mechanism. Although the average bitrate (goodput) for

each flow comes near their fair share value over the long run, significant fluctuations of the instantaneous bitrate for each flow results in underutilization of the reserved slice capacity and also variations in the application level DASH video quality levels. Hence, we pose the following question: Assuming we reserve a specific end-to-end (e2e) average bitrate for a set of DASH streams by one of well-known QoS mechanisms, how can we i) minimize instantaneous throughput fluctuations, ii) maximize QoE fairness among flows, and iii) maximize utilization of the allocated average bitrate.



Şekil 3.1: Instantaneous TCP rates for default TCP vs. TCP rate control: a) sender and b) receiver rates for a single standard TCP flow in a reserved 5 Mbps network slice, c) sender and d) receiver rates for a single TCP flow with rate control in a reserved 5 Mbps slice, e) sender and f) receiver rates for four standard TCP clients in a reserved 20 Mbps slice, g) sender and h) receiver rates for four TCP clients with rate control in a reserved 20 Mbps slice.

The answer is that we need to apply rate control for each TCP flow when their rates reach the fair-share rate reserved for that flow in order to properly manage QoE variability and network utilization. We demonstrate that this can be achieved by active receive window management (AWM) or ACK pacing for each TCP flow. The send and receive rates of the four TCP flows over a shared slice configured at 20 Mbps are shown in Figure 3.1(e)-(h) with and without setting optimal receive window sizes. It can be seen that setting *rwnd* to an optimal value, given the fair-share bitrate, at

receiving clients decreases TCP throughput variation significantly and achieves nearly a flat rate for each client near their fair-share value. Moreover, comparing the send rates shows that TCP rate control also helps avoiding bursts at the ingress queues of switches. This observation applies even to the case of dynamic per-flow queue management with one flow per queue (slice). We see from Figure 3.1(a)-(d) that even in the case of a single flow per slice, TCP rate control is needed for a smooth send rate, which helps reducing the flow processing at the sender side switch.

In the following, we propose network-based QoS resource reservation for a group of DASH flows and two different collaboration architectures, namely a centralized collaboration between the NSP, VSP and clients in Section 4 and a distributed inter-client collaboration architecture in Section 5, in order to provide clients information about their fair-share bitrates so that they can regulate their TCP rate at a fair level.

3.4 Queue Management for VSP-Managed Video Services

The NSP employs queue management methods for QoS (throughput) reservation at the network level. Configuring multiple queues at switch ports for packet egress provides per-flow rate-limiting ability and enables weighted fairness among flows. Having received per-flow routes and optimal downstream bandwidths to be allocated for each DASH client from the TEM, the SDN controller configures QoS queues at the ports of network edge switches. Queue management is handled by configuration protocols such as OF-Config [37] or OVSDB [38] based on controller/switch compliancy since both parties need to support the same switch configuration protocol. Once queues are implemented, by using OpenFlow queuing action, traffic flows can be routed over particular queues satisfying the related bitrate requirements. Queues can be configured in two different ways: (i) dynamic configuration of a new queue per flow [17], or (ii) static configuration of a fixed number of queues with some maximum bitrates.

In the dynamic configuration, queues are configured on-demand per-flow. For each incoming video service request, a new queue at a specific rate is created and removed once the flow expires. Deployment of soft virtual switches, such as Open vSwitch

(OVS) [39], by NSPs running over high performance computers with sufficient resources at the network edges makes realization of on-demand per-flow queue management feasible. In this approach, since each flow is assigned to a flow-specific queue, traffic burst in one flow does not affect the QoS of other flows as long as the sum of capacities of all queues over a port stay under the link capacity. However, TCP rate control by clients helps reducing flow processing requirements at the ingress switches as demonstrated in Figure 3.1(a)-(d).

In the static configuration, a fixed number of shared queues with certain operating rates are configured once, and incoming video service requests are dynamically assigned to these queues. The number and rates of queues to be configured by the NSP are determined by taking the adaptation levels of DASH contents into consideration since the NSP and VSP collaborate. The idea is to configure specific queues for subsets of these adaptation levels. Let B be the set of average bitrates for different adaptation levels, and B_i is the average bitrate for the i^{th} adaptation level, where i is between 1 and N_{AL}, the number of adaptation levels. We partition B into non-overlapping N_Q subsets. Let B_k^n represent adaptation bitrate k in the n^{th} subset. We configure N_Q queues, such that the rate for n^{th} queue is set to $max(B_k^n) \cdot U_{max}$, where U_{max} is the number of DASH clients that can be served at the highest quality level by n^{th} queue. It is common for VSPs to encode DASH content at different bitrates ranging from 0.3 Mbps up to 20 Mbps at 10 adaptation levels covering multiple video resolutions, e.g., SD, HD, Full HD, and 4K. For example, these adaptation levels can be partitioned into four non-overlapping subsets and four different queues can be configured where each queue is used for flows of a particular resolution consisting of multiple adaptation levels. The necessity of TCP rate control to achieve smooth QoE per flow and QoE fairness has been demonstrated in Figure 3.1(e)-(h).

Chapter 4

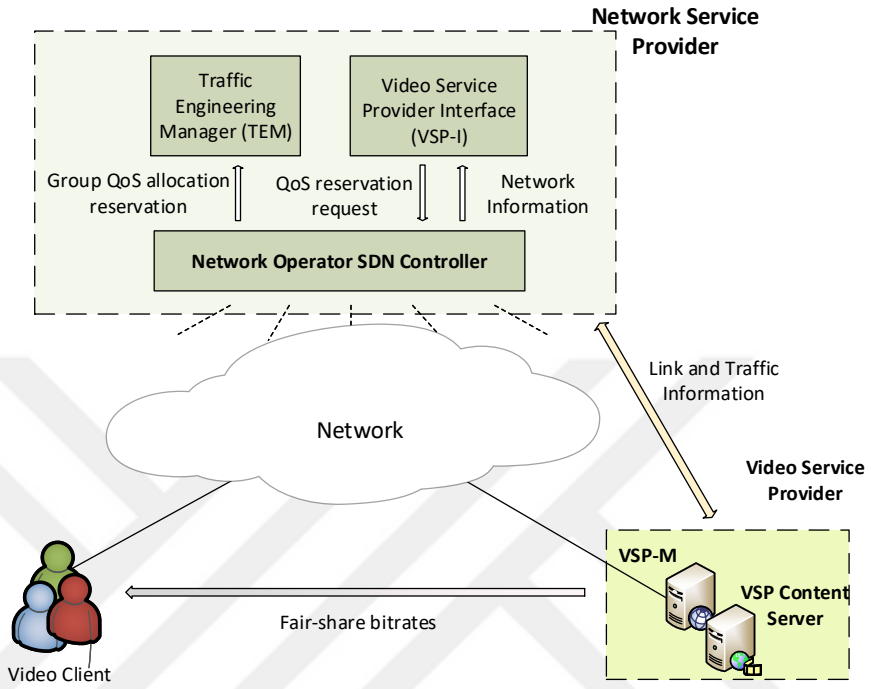
VSP-MANAGED CENTRALIZED VIDEO SERVICE

The VSP-Managed centralized video service archtitecture aims to achieve QoE fairness among heterogeneous HAS clients with the fair-share bitrate allocation from Video Service Provider. In the proposed system, VSP has the link and client information, and VSP determine each clients fair share bitrates. The architecture of proposed system is presented in section 4.1.

4.1 System Architecture

In the proposed system architecture, the VSP supports a Management Server (VSP-M) to manage the collaborative streaming service, and the NSP supports a Traffic Engineering Manager (TEM) for resource provisioning and a VSP Interface (VSP-I) to communicate with the VSP. The proposed system architecture together with the flow of information is depicted in Figure 4.1.

The information flow for VSP-managed video services consist of the following steps: 1) The client requests video service from VSP through the VSP management server (VSP-M). 2) VSP passes source and destination IP to the NSP. 3) NSP-TEM computes shortest-path route for client. 4) NSP passes an updated list of slices on all links with a list of all VSP clients on each slice. 5) VSP re-computes the bitrate it requests from the NSP for slices over all links (knowing the bitrate requirements of all of its clients), 6) NSP updates reserved bitrate for the slice allocated to the VSP over each link and informs the VSP. The capacity of reserved slices are determined by the NSP according to how many other non-VSP users are on each link. It is important that the VSP employs a reserved slice in order to isolate VSP clients performing TCP receive window management from other non-VSP users. 7) VSP manages these slices



Şekil 4.1: System architecture for centralized collaboration.

(see Algorithm 1 and its description in Section 4.2), where in centralized collaboration, the VSP computes the fair-share bitrate for each client over each reserved slice and informs clients about their fair-share bitrate (see Section 4.2.1). 8) Clients set the TCP receive-window size and DASH video adaptation rate based on this information.

4.2 Slice Management by VSP

An end-to-end path from a video server to a client consists of multiple links/slices and the VSP need to manage its clients over each slice. In step 7 above, the VSP manages the fair-share bitrates of clients over the set of slices that the NSP allocates to the VSP (step 6 above). The procedure for VSP-Managed Centralized Video Service slice management is described by means of Algorithm 1, which is run by the VSP-M. The input to Algorithm 1 is a list of slices $L = \{l_1, \ldots, l_N\}$ at each link i, such that $l_i = (C_i, D_i)$, where C_i is the reserved capacity for slice i and D_i represents the set of VSP clients on slice i. We denote the list of all current VSP clients by $D = \bigcup_{i=1}^{N} D_i$. Algorithm 1 involves the following steps:

Input: L
$qV = 0.96$;
$r_i = (\frac{qV - c_i}{a_i})^{\frac{1}{b_i}}, i = 1, \ldots, M$;
do
  foreach *slice* l_j *in* L **do**
    Count N_i^j clients on l_j with class $i = 1, \ldots, M$;
    Compute congestion ratio $CR_j = \frac{\sum_{i=1}^{M} r_i \cdot N_i^j}{C_j}$;
  end
  Sort links in L based on CR;
  Create $D = \bigcup_{i=1}^{N} D_i$;
  Initialize br_k for each client with id k in D;
  foreach *slice* l_j *in* L **do**
    $B_j \leftarrow$ Call Alg. 3 with inputs C_j and N_i^j;
    foreach *bitrate* br_k^j *in* B_j **do**
      if $br_k^j < br_k$ **then**
        $br_k = br_k^j$;
        Notify client k about br_k;
      end
    end
  end
  Sleep for Δt seconds;
  Update link capacities in L;
while *true*;

Algorithm 1: VSP - Slice Management Algorithm Centralized Video Service

4.2.1 Determining Fair-share Bitrates for Different Video Resolutions

This step explains the theoretical foundation behind the steps 2 and 3 of Algorithm 1. In order to achieve QoE fairness among heterogeneous DASH clients, the fair-share bitrate for each DASH client should depend on their video resolution such that all clients receive the same or similar video quality regardless of the resolution of the video they receive.

The empirical relationship between video quality, measured by Structural Similarity Index (SSIM), and bitrate for $M = 3$ different video resolutions is depicted in Figure 4.2 [18] [24]. These curves can be compactly represented by three "video quality coefficients" by fitting a function of the form $qV = a \cdot r^b + c$ [18] [24], where the coefficients a, b, and c can be computed by least squares curve fitting. The resulting coefficients for $M = 3$ different video resolutions are provided in Table 4.1. Then, given a particular quality value qV^*, the fair-share bitrates r_i for resolution type $i = 1, \ldots, M$ can be calculated from

$$r_i = (\frac{qV^* - c_i}{a_i})^{\frac{1}{b_i}}, i = 1, \ldots, M \tag{4.1}$$

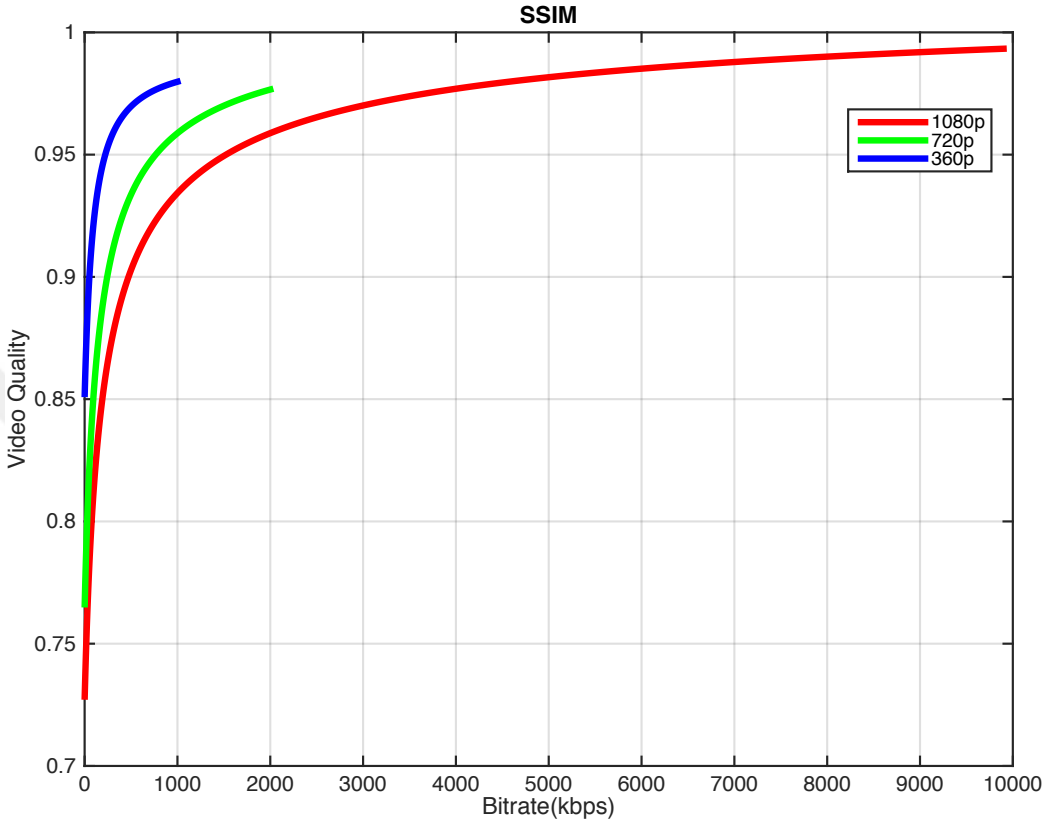
where a_i, b_i, and c_i denote coefficients for resolution type i shown in Table 4.1.

Tablo 4.1: Coefficients of the video quality function [18].

	Power Series Model			Goodness of Fit	
Device Type	a	b	c	Adjusted R^2	RMSE
1080p	-3.035	-0.5061	1.022	0.9959	0.006011
720p	-4.85	-0.647	1.011	0.9983	0.002923
360p	-17.53	-1.048	0.9912	0.9982	0.002097

4.2.2 Sort the Slice List According to Congestion Ratio

Steps 4-8 of Algorithm 1 computes the congestion ratio for each slice. The congestion ratio CR_j for a slice j is defined as the total requested traffic, given by the sum of fair-share bitrates of clients over slice j, divided by the capacity C_j of slice j. Hence,



Şekil 4.2: Mapping between video quality and bit-rate according to three types of display resolution [18]

it is given by $CR_j = \frac{\sum_{i=1}^{M} r_i \cdot N_i^j}{C_j}$, where the fair-share bitrates r_i are computed from Eqn. 4.1 by setting qV to a reasonable value. VSP performs slice management starting from the highest congested slice since it is more likely to create bottleneck along the path. Hence, the VSP sorts slices based on CR in decreasing order (Line 9).

4.2.3 Computation of Fair-Share Bitrates

In the centralized architecture, VSP computes the fair-share bitrate of each client after traversing over all slices in the sorted link set L starting from the highest congested link. Using the current capacity C_j of slice j and the number N_i^j of clients with resolution type i over slice j as inputs to Algorithm 2 (described in Section 4.3), VSP computes the fair-share bitrates B_j of all clients on slice j (Line 13). After processing each slice j, the algorithm checks if the fair-share bitrate br_k^j of client k is less than

its present value br_k. Since the e2e bitrate of each client is determined by the smallest rate over all slices, the VSP notifies client k about its new fair-share bitrate if br_k^j is smaller than the present value. (Line 14-17). We note that our experiments indicate that the smallest bitrate is always the one computed over the highest congested slice. Since the network state and/or the number of DASH clients vary over time, the per-client fair-share bitrate calculation in the centralized architecture must be dynamic. Hence, the procedure needs to be periodically repeated at every Δt seconds, i.e., the VSP periodically triggers slice management using the updated link informations and clients (Line 25-27).

4.3 Fair-Share Bitrate Algorithm

In the centralized collaboration architecture, the fair-share bitrates for each client is computed by the VSP Management Server as described in Algorithm 2. The inputs to the algorithm are the number of clients N_i for each video resolution type $i = 1, \ldots, M$ and the slice capacity C. The algorithm aims to find the maximum video quality qV such that the sum of the fair-share bitrates $r_i, i = 1, \ldots, M$ corresponding to this qV for all M clients over the slice is equal to the capacity C. This is achieved by means of an iterative search procedure in Lines 7-14 of Algorithm 2. We set the parameters convergence margin ϵ and search step size ΔqV in Lines 3-4. The variables total bitrate *total* and video quality qV are initialized in Lines 5-6. The initial qV, that is, the initial common video quality value for all clients is set to a low value so that the resulting initial total bitrate does not exceed the slice capacity. The bitrates r_i for different video resolutions corresponding to a given qV at current iteration are obtained by Eqn. 4.1, where a_i, b_i, and c_i for video resolution type i are provided in Table 4.1 (Line 8). The total bitrate *total* for all clients given the current qV value is computed in Line 9. If the total bitrate does not exceed slice capacity C, we increment the value of qV by the stepsize ΔqV. Otherwise, stepsize ΔqV is halved, and qV is

decremented by the new ΔqV. Iterations continue until the total bitrate converges to within ϵ neighborhood of C. Once the fair-share bitrates r_i are computed, the VSP informs clients about their fair-share bitrates, so that each client updates its TCP receive-window size *rwnd* and DASH video adaptation rate as described in Section 4.6.

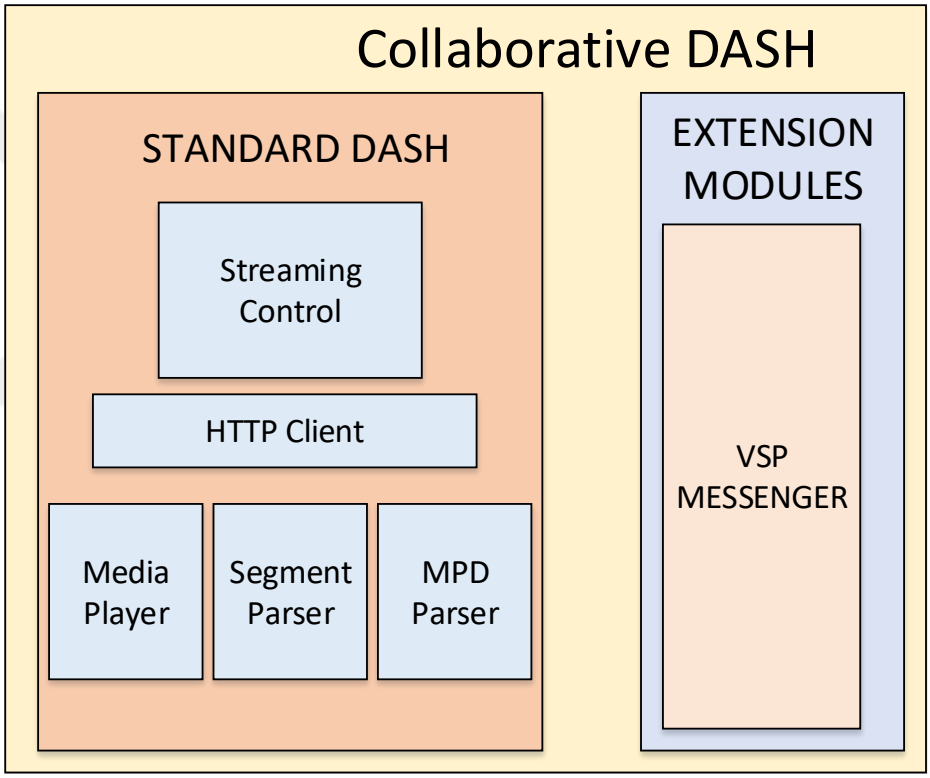
Input: C, $N_i, i = 1, \ldots, M$
Output: $r_i, i = 1, \ldots, M$
$\epsilon = 100$;
$\Delta qV = 0.01$;
$qV = 0.8$;
$total = 0$;
while $|C - total| > \epsilon$ **do**
  $r_i = (\frac{qV - c_i}{a_i})^{\frac{1}{b_i}}$, $i = 1, \ldots, M$;
  $total = \sum_{i=1}^{M} N_i \cdot r_i$;
  if $total < C$ **then**
    $qV = qV + \Delta qV$;
  else
    $\Delta qV = \frac{\Delta qV}{2}$;
    $qV = qV - \Delta qV$;
  end
end

Algorithm 2: VSP - Centralized Fair Share Algorithm

4.4 Collaborative DASH Implementation of Centralized Video Service

Collaborative DASH of Centralized Video Service is an extensive dynamic adaptive video streaming java client application. In this client application, the standart DASH modules are streaming control, HTTP client, media player, segment parser, and MPD parser [34]. The streaming control module is responsible for scheduling next segment in accordance with some adaptation algorithms such as [35] [36]. The HTTP client establishes a connection between DASH client and content server to request scheduled segments. Media player obtains the requested chunk and plays this chunk. Segment parser module is in charge of initialization of segments. Finally, MPD parser is used to analyze MPD file obtained from content server, and to inform adaptation logic about

available adaptation level sets. In addition to these modules,there exist one additional module, VSP messenger that provides communication between collaborative DASH client and VSP. VSP messenger are implemented with java server-client socket programming. The collaborative DASH implementation architecture of centralized video service is demonstrated in Fig 4.3.



Şekil 4.3: Collaborative DASH architecture

Furthermore, VSP messenger is a member of both VSP-centric centralized and distributed architectures. Each centralized collaborative DASH client gather the information about its fair-share bitrate from VSP, and transmit this value to the streaming control unit to schedule the adaptation level of next segment.

4.5 VSP-Client Collaboration

VSP messenger module provides the collaboration between VSP and DASH client. The required parameter that guide the DASH client to perform suitable adaptation

is sent via a json message format seen in 4.4. In this json message, client is informed about the suggested bitrate to adjust its adaptation bitrate and TCP receive window size.

```
{
  “suggestedBitrate”: ”4.5317”
}
```

Şekil 4.4: VSP to DASH client message

4.6 TCP Rate Control by Clients

It is important to note that resource reservation by queue management alone is not sufficient to control per-flow QoE variability. This is because the standard TCP congestion control is not aware of the reserved network resources and continues to increase its rate until experiencing losses, which results in large fluctuations in the per-flow instantaneous throughput. The throughput fluctuations result in client QoE variations as a consequence of application layer adaptation mechanism. This is especially noticeable when the number of flows assigned to a particular queue is close to U_{max}, i.e., queue utilization approaches its maximum level.

The QoE variability can be avoided by applying rate control to each TCP flow by ACK pacing when the TCP rate reaches the rate reserved for that flow. Once clients are informed by VSP-M about their optimal reserved bitrates based on the current network utilization, they can dynamically manage their TCP windows sizes. Each client periodically estimates the round-trip-time (RTT) to VSP content server (VSP-CS) so that *rwnd* is set to the optimal value $bandwidth \cdot RTT$ and advertised in TCP ACKs.

Chapter 5

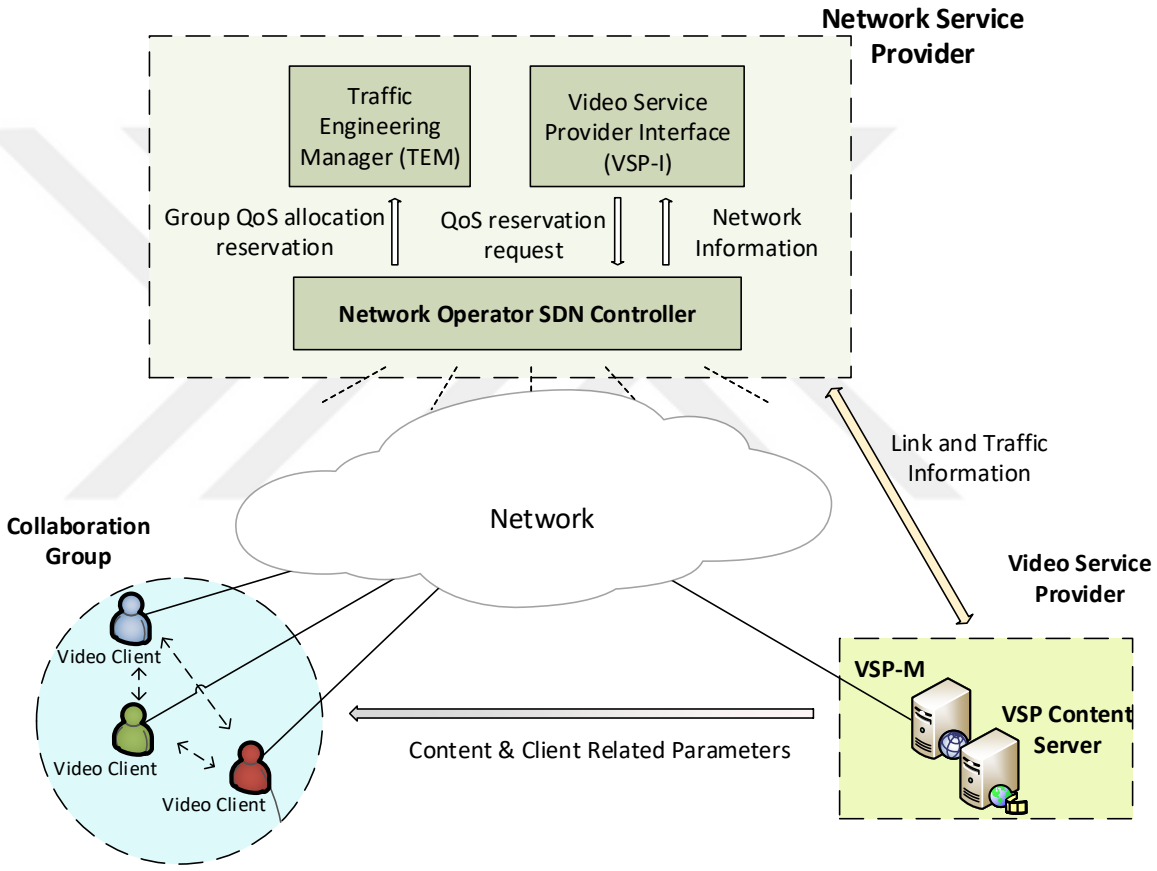
VSP-MANAGED DISTRIBUTED VIDEO SERVICE

The VSP-Managed distributed video service archtitecture aims to achieve QoE fairness among heterogeneous HAS clients without the fair-share bitrate allocation for each client from a central traffic engineering unit. In the proposed system, the clients that share a reserved network slice communicate with each other to determine their own fair-share bitrates with the assistance of the VSP Management Server rather than competing with each other. The architecture of proposed system is presented in section 5.1.

5.1 System Architecture

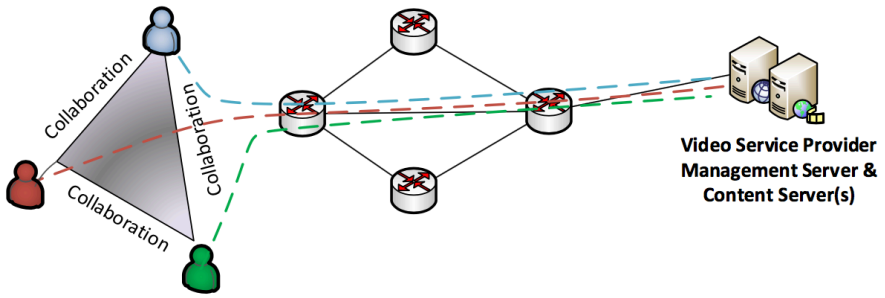
The information flow for VSP-Managed distributed video service consist of the following steps as VSP-Managed centralized video service: 1) The client requests video service from VSP through the VSP management server (VSP-M). 2) VSP passes source and destination IP to the NSP. 3) NSP-TEM computes shortest-path route for client. 4) NSP passes an updated list of slices on all links with a list of all VSP clients on each slice. 5) VSP re-computes the bitrate it requests from the NSP for slices over all links (knowing the bitrate requirements of all of its clients), 6) NSP updates reserved bitrate for the slice allocated to the VSP over each link and informs the VSP. The capacity of reserved slices are determined by the NSP according to how many other non-VSP users are on each link. It is important that the VSP employs a reserved slice in order to isolate VSP clients performing TCP receive window management from other non-VSP users. 7) VSP manages these slices (see Algorithm 3 and its description in Section 5.2), in the distributed collaboration architecture, the VSP forms groups of clients and lets them collaborate among each other to determine

their fair-share bitrates (see Section 5.3), 8) Clients set the TCP receive-window size and DASH video adaptation rate based on this information.



Şekil 5.1: System architecture for distributed collaboration.

The idea of user collaboration groups consisting of clients sharing a network slice, where three DASH clients whose traffic are indicated by the dotted lines sharing a slice, is illustrated in Figure 5.2. In the standard DASH scenario, these clients are unaware of each other and they compete with each other relying only on TCP congestion control, which results in per-client QoE variability and overall underutilization of link capacity. In the proposed system, these three users form a collaboration group over the link they share.



Şekil 5.2: Grouping

5.2 Slice Management by VSP

In VSP-Managed distributed video service, the slice management by VSP has two steps that are determining Fair-share Bitrates for different resolutions, and sorting the slice list according to congestion ratio as explained in 4.2.1 and 4.2.2 . In contrast to VSP-Managed Centralized Video Service, VSP-Managed Distributed Video Service do not perform the computation of fair-share bitrates for each client, and let each VSP client to compute its fair-share bitrate with respect to its group using algorithm 4.As mentioned in VSP-Managed Centralized Video Service,an end-to-end path from a video server to a client consists of multiple links/slices and the VSP need to manage its clients over each slice.In step 6 given in system architecture, the VSP gathers reservation information for each VSP client group from NSP. Moreover, VSP has the information of each DASH client to determine the fair-share bitrates for different video resolutions to determine a particular quality value using equation 4.1, then VSP can sort the slice list according to congestion ratio by using this particular quality value. The procedure for VSP-Managed Distributed Video Service slice management is described by means of Algorithm 3, which is run by the VSP-M. The input to Algorithm 3 is a list of slices $L = \{l_1, \ldots, l_N\}$ at each link i, such that $l_i = (C_i, D_i)$, where C_i is the reserved capacity for slice i and D_i represents the set of VSP clients on slice i. We denote the list of all current VSP clients by $D = \bigcup_{i=1}^{N} D_i$.

Input: L
$qV = 0.96$;
$r_i = (\frac{qV - c_i}{a_i})^{\frac{1}{b_i}}, i = 1, \ldots, M$;
do
  foreach *slice* l_j *in* L **do**
    Count N_i^j clients on l_j with class $i = 1, \ldots, M$;
    Compute congestion ratio $CR_j = \frac{\sum_{i=1}^{M} r_i \cdot N_i^j}{C_j}$;
  end
  Sort links in L based on CR;
  Create $D = \bigcup_{i=1}^{N} D_i$;
  Initialize br_k for each client with id k in D;
  Initialize an empty set D^*;
  foreach *slice* l_j *in* L **do**
    Remove client ids in D^* from D_j;
    Adjust C_j considering the removed clients;
    Notify clients in D_j about $k's$ in D_j and C_j;
    Add client ids in D_j to D^*;
  end
  Sleep for Δt seconds;
  Update link capacities in L;
while *true*;

Algorithm 3: VSP - Slice Management Algorithm for Distributed Video Service

5.3 Interclient Collaboration and Fair-Share Bitrate Algorithm

Implementation of inter-client collaboration requires modification of the DASH client as depicted in Figure 5.3. The collaboration functionality is provided by (i) a new messenger module, which manages communication of a DASH client with other members of the collaboration group and the VSP-M for receiving/sending messages, and (ii) buffer state aware fair-share computation (Algorithm 4) module.

Input: C, d_{ij}
Output: r_i, α_{ij}
$\epsilon = 100$;
$\Delta qV = 0.01$;
$qV = 0.8$;
$total = 0$;
$\overline{d}_i = \frac{\sum_{j=1}^{N_i} d_{ij}}{N_i}, i = 1, \ldots, M$;
while $|C - total| > \epsilon$ **do**
  $r_i = (\frac{qV - c_i}{a_i})^{\frac{1}{b_i}}, i = 1, \ldots, M$;
  $\alpha_{ij} = 2 - \frac{d_{ij}}{\overline{d}_i}, i = 1, \ldots, M$ and $j = 1, \ldots, N_i$;
  $W_i = \sum_{j=1}^{N_i} \alpha_{ij}, i = 1, \ldots, M$;
  $total = \sum_{i=1}^{M} W_i \cdot r_i$;
  if $total < C$ **then**
    $qV = qV + \Delta qV$;
  else
    $\Delta qV = \frac{\Delta qV}{2}$;
    $qV = qV - \Delta qV$;
  end
end

Algorithm 4: Clients - Distributed Fair Share Algorithm

Given that each DASH client has access to the video quality coefficients of all group members, the reserved capacity C for the group, and most recent buffer durations d_{ij} for client j with video resolution i, each client can compute its own bitrate using Algorithm 4 that is similar to Algorithm 2 but additionally uses playback buffer information of clients in the collaboration group. The idea is to scale the bitrates to be allocated for each client according to the respective playback buffer conditions. DASH clients with buffer status lower than the average buffer fullness $\overline{d}_i$ of VSP-clients with

a particular resolution type i on a given group are more likely to experience a stall event. Therefore, DASH clients with buffer status higher than this average level should be allocated a proportionally smaller bitrate than their fair-share bitrate in order to allow a higher share of the slice capacity to clients with buffer status lower than the average. To this effect, we compute the average buffer fullness measure $\overline{d}_i = \frac{\sum_{j=1}^{N_i} d_{ij}}{N_i}$ (Line 7), where N_i denotes the number of clients with resolution type i in the group and d_{ij} is the normalized buffer duration such that $0 \leq d_{ij} \leq 1$ for all ij. Deviation factor $\frac{d_{ij}}{\overline{d}_i}$ represents a measure of relative buffer state of client j among the client with resolution type i, hence, the buffer scale factor α_{ij} is computed as $2 - \frac{d_{ij}}{\overline{d}_i}$ (Line 10). In this way, total weight W_i for resolution type i is computed and buffer status aware total bitrate equation is formed (Line 10-11). Similar to Algorithm 2, Algorithm 4 iteratively searches for the highest possible fair-share bitrates r_i. As a result, client j with resolution type i computes its buffer-aware fair-share bitrate as $r_i \cdot \alpha_{ij}$ and updates its TCP receive-window size and DASH video adaptation rate accordingly.

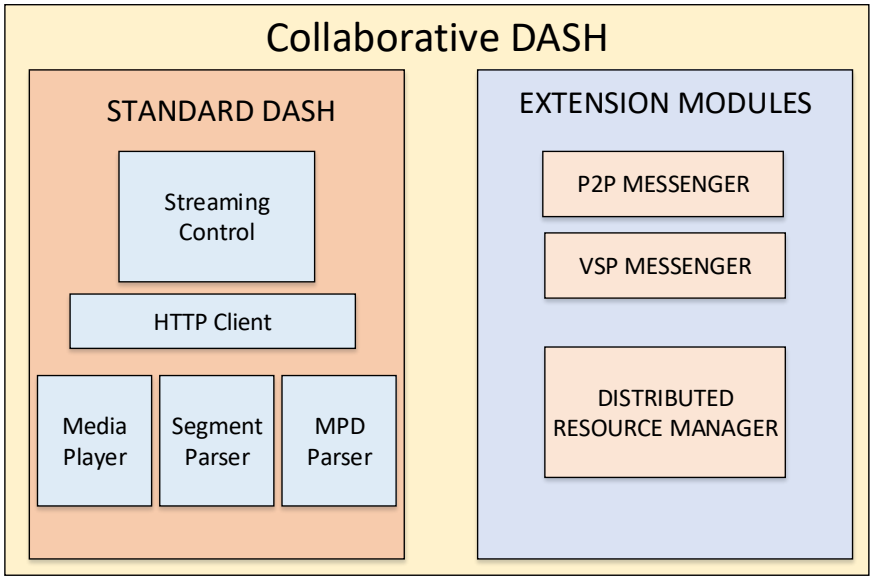
Accurate collaboration among DASH clients requires periodic communication, which is possible by soft synchronization. For synchronization, each client receives the current time-stamp from the VSP Management Server only once when they first connect to the server, and use it (after adjusting by the estimated RTT) as their reference wall-clock. Thus, the communication overhead for collaboration synchronization is negligible. Periodicity of collaboration is essential since the state of client buffers vary with time, and DASH clients should inform other group members about their buffer status to adjust their fair-share according to the new conditions. For effective collaboration, clients need recent playback buffer information of other clients in the group. To this end, clients periodically send unicast messages to notify each other about their present playback buffer status so that they can collaboratively set their TCP receive window sizes and decide next segment's adaptation level based on group QoE fairness. For a group of N collaborating clients, there exist $N \cdot (N-1)$ messages at each collaboration cycle. Since each message carries only a single number (representing playback buffer duration) of 4 bytes and the collaboration period is typically

5-10 seconds, the communication overhead due to inter-client information exchange is also very small. The structure of periodic messages, which consist of the identity of client and its buffer status, is as follows:

"$NeighborInformation$" :
"$Client1$" : "Id" : "10.0.0.2", "$BufferStatus$" : "0.5"
"$Client2$" : "Id" : "10.0.0.3", "$BufferStatus$" : "1.5"
"$Client3$" : "Id" : "10.0.0.5", "$BufferStatus$" : "0.8"

5.4 Collaborative DASH Implementation of Distributed Video Service

Collaborative DASH of Distributed Video Service is an extensive dynamic adaptive video streaming java client application. In this client application, the standart DASH modules as explained in 4.4 are streaming control, HTTP client, media player, segment parser, and MPD parser [34]. In addition to these modules,there exist three additional modules that are VSP messenger, P2P messenger and distributed resource manager. VSP messenger provides communication between collaborative DASH client and VSP. VSP messenger are implemented with java server-client socket programming.



Şekil 5.3: Collaborative DASH architecture

P2P messenger is in charge of the communication among clients in each VSP group, and it is implemented with java server-client socket programming. Distributed resource manager is a simple iterative java method and is a member of VSP-Managed distributed video service architecture. VSP messenger feeds P2P messenger with the information of neighbor list in order to establish a connection from one client to another client. When P2P messenger establishes each connection for its VSP group, then this P2P messenger starts to share buffer status parameter to inform other DASH clients to run their distributed resource manager with given buffer status parameters.

5.5 VSP-Client Collaboration

VSP messenger module provides the collaboration between VSP and DASH client. The required parameters that guide the DASH client is sent via a json message format seen in 5.4. In this json message, client is informed about the group members and reserved capacity used to calculate fair-share bitrate in inter-client collaboration.

```
{
  “neighborList”: {
     “neighbor1”:{
      “a”: “-3.035”,
      “b”: “-0.5061”,
      “c”: “1.022”
  },
     “neighbor2”:{
      “a”: “-4.85”,
      “b”: “-0.647”,
      “c”: “1.011
  }
  },
  “reservedCapacity”: “10”
}
```

Şekil 5.4: VSP to DASH client message

5.6 TCP Rate Control by Clients

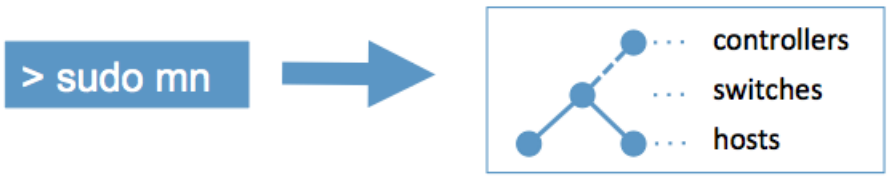
As mentioned in section 4.6 of VSP-Managed Centralized Video Service, resource reservation by queue management alone is not sufficient to control per-flow QoE variability due to the congestion control of the standard TCP since the TCP congestion control is not aware of the reserved network resources. Therefore, TCP congestion control acts in a way that no resource reservation, and increase its rate until a loss occurs, which results in large fluctuations in the per-flow instantaneous throughput. This throughput fluctuations in transport layer causes QoE variations for clients in application layer adaptation mechanism. Furthermore, these QoE variations are more noticeable when the number of flows assigned to a particular queue is close to the maximum queue utilization. Hence, the QoE variablity can be avoided by applying the rate control for each TCP flow. Once VSP-Managed Distributed Video Service client computes its fair-share bitrate value with distributed resource manager, this client can manage its TCP receive window size. Each client periodically estimates the round-trip-time (RTT) to VSP content server (VSP-CS) so that *rwnd* is set to the optimal value $bandwidth \cdot RTT$.

Chapter 6

DYNAMIC ADAPTIVE VIDEO STREAMING EVALUATIONS

6.1 Evaluation Environment

Evaluation environment to deploy DASH clients, simple HTTP server, and VSP is chosen as Mininet. Mininet is a tool that establishes a virtual network in which there exist real kernel, switch and application related codes. This virtual network can be on a single machine that can be virtual machine, cloud or native. In general, a network can be established by two main methods that are built-in topology scripts illustrated in Fig 6.1 or custom topologies created by users demonstrated in Fig 6.2.

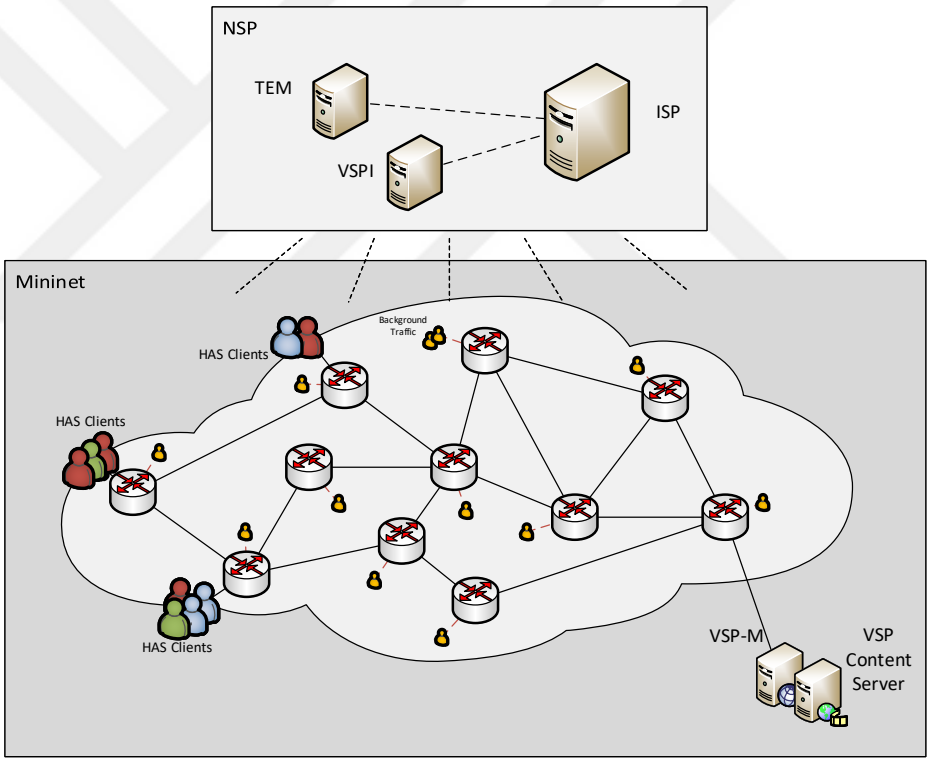


Şekil 6.1: The sample built-in Mininet topology

In custom topologies, each link and switch configuration can be set by users. The connection configuration among switches can be determined, and the delay, capacity and loss of each link can be adjusted. Mininet is suitable for teaching, research and development. For research and development, Mininet is a great environment to experiment with OpenFlow and Software-Defined Networking. Hence, the overlay network in DASH Video streaming evaluations is emulated in Mininet.

6.2 Overlay Network Description

We evaluate the performance of both centralized and distributed collaboration systems over a SDN, whose data plane consists of a partial mesh network generated by Mininet [40] using OVS switches [39], which are connected to a Floodlight [41] controller. Network topology contains 11 switches, where each switch has at least one client as shown in Figure 6.2.



Şekil 6.2: Exemplary network used for evaluations.

In our implementation, TEM and VSP-I units of the NSP expose resource allocation/routing and NSP-VSP collaboration functionalities, respectively, as web services employing JSON based APIs. In the centralized system, TEM uses GAMS Java API [42] to run its optimization engine as described in our previous work [43].

The VSP is represented by one of the hosts that runs a management and a content server using HTTP 1.1 with persistent connections. Content server stores the media presentation description (MPD) files and encoded segments. The first test video is

Tablo 6.1: Resolutions and encoding parameters for test video content.

Content	Resolution	Bitrates (Mbps)	segment(sec)
Tears of Steel	1080p	1.5, 2.4, 3, 4, 6, 10	4, 6, 10
	720p	0.50, 0.80, 1.5, 2.4	
	360p	0.50, 0.80	
Big Buck Bunny	1080p	1, 1.2, 1.5, 2.1, 2.4, 2.9, 3.3, 3.6, 3.9	6
	720p	0.50, 0.57, 0.78, 1, 1.2, 1.5	
	360p	0.17, 0.22, 0.25, 0.32, 0.37, 0.50, 0.57	

Tears of Steel with duration 12 minutes. The second video is animated movie *Big Buck Bunny* with duration 10 minutes. Both videos are variable bitrate (VBR) encoded. The available resolutions and encoding parameters for these two videos are shown in Table 6.1. Since we run our experiments on network slices with reserved link capacity in order to investigate the interaction between QoS and TCP congestion control, the only source of cross-traffic is other competing or collaborating DASH clients using the same network slice. Hence, we use DASH segment durations of 4sec., 6 sec. and 10 sec. only.

Our DASH client uses Java sockets for communication with the VSP-M and the group members (in the distributed system), and also to set calculated receive buffer sizes. We estimate *rtt* as the time difference between the instant when a request is sent to the server and the instant when the first byte of the response is received. In the experiments, the same application layer DASH rate adaptation algorithm is triggered when comparing the competitive and collaborating clients.

6.3 Evaluation of DASH over Collaborative Architectures

6.3.1 Results

We perform five different sets of experiments. The first set of results compare the performance of the proposed collaborative architectures (VSP-managed centralized, VSP-managed distributed) with those of SDN-based QoE-aware DASH streaming (available in the literature) and basic competitive DASH streaming over a fully utili-

zed network slice. The second set of results show the performance of both centralized and distributed VSP-managed collaborative streaming when clients enter and exit at random times. The third set of results show the robustness of VSP-managed distributed collaboration in a pessimistic scenario with a large number of clients in a group and limited available reserved slice capacity. In the fourth set of results, we investigate the performance of the VSP-managed distributed collaborative DASH and competitive DASH under varying slice capacity due to the dynamic change of NSP resource allocation for the VSP. Finally, the fifth set of results study the effects of DASH segment size and network slice capacity on the performance of competitive vs. collaborative streaming.

In particular, each experiment is repeated 10 times and we report the mean and variance, in the case of video quality qV, of the results of these runs, as well as mean number of stalls (MNS) and the mean duration of stalls (MDS) in seconds with different number of video resolutions and clients, varying DASH segment durations, and varying reserved link capacity.

In the first group of results, we compare the performances of i) Competitive: Basic DASH streaming (with no resource reservation and no TCP receive-window (rwnd) control), ii) SDN-DASH: DASH streaming with SDN-based centralized fair-share bitrate calculation per-client (but no client-side rwnd control), where clients only use this bitrate in their application layer DASH rate adaptation heuristic ([18]), iii) VSP-managed (Centralized): DASH streaming with VSP-based centralized fair-share bitrate calculation for a group of DASH clients over a slice, and iv) VSP-managed (Distributed): A group of clients over a reserved slice communicate with each other to compute their own fair-share bitrates. We first study the case of homogeneous clients, where all clients receive 1080p content, and then the case of heterogeneous clients, where clients receive 1080p, 720p, or 360p video.

In the homogeneous clients scenario, we have four 1080p DASH clients over a 20 Mbps slice so that the fair share bitrate for each client is 5 Mbps. Each DASH client streams *Tears Of Steel* with 4 sec. segments. In the cases of competitive DASH

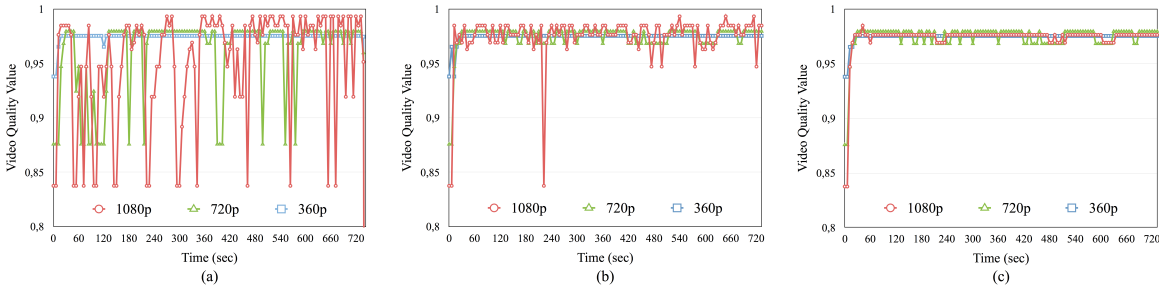
Tablo 6.2: Comparison for four homogeneous 1080p clients.

Method	Mean QV	Var QV	MNS	MDS
Competitive	0,9752	0,00006	1	3,5
SDN-DASH	0,9765	0,00006	1	1,2
VSP-Managed (Centralized)	0,9776	0,00004	0	0
VSP-Managed (Distributed)	0,9777	0,00003	0	0

and SDN-DASH streaming, application-level DASH adaptation bitrates for all clients fluctuate frequently between 2.4 Mbps (lowest) and 10 Mbps (highest) adaptation levels. In contrast, in all types of collaborative architectures, we observe that bitrate adaptation for all DASH clients only vary between 4 and 6 Mbps, where the fair-share bitrate for each client is 5 Mbps. We also observe from Table 6.2 that both NSP-managed and VSP-managed (centralized and distributed) collaborative frameworks provide better results (for all measures) than competitive and SDN-DASH streaming. In particular, the MDS and MQP values for competitive DASH clients are significantly higher than those of both types of collaboration. We also note that the VSP-managed distributed collaborative framework achieves the highest mean video quality (and smallest variance) and does not experience any stalls since all clients in a group are aware of buffer status of each other.

In the heterogeneous clients scenario, first we present results for three clients with video resolutions 1080p, 720p and 360p, respectively, streaming the video *Tears Of Steel* with 4 sec. segments over a reserved network slice with capacity 7 Mbps. In VSP-managed centralized collaborative streaming, the fair share bitrates for these clients can be computed as 4.02 Mbps for 1080p, 2.09 Mbps for 720p and 0.86 Mbps for 360p clients, which correspond to the fair video quality value $qV = 0.9765$ (see Algorithm 3). Clearly, how close we can achieve this qV value depends on how close the determined bitrate values are to the DASH adaptation bitrate sets of a particular content (available at the content server), since each DASH client starts streaming

with the nearest adaptation level to its fair-share bitrate. Table 6.3 shows that the ensemble mean of quality values of different resolution video clients over 10 runs are closer to the computed qV value in collaborative streaming (compared to competitive streaming), which is an indicator that we can achieve fairness among heterogeneous DASH clients. In the case of 360p client, we see that the competitive streaming is as good as collaborative streaming since the 360p client gets an unfair share of the bitrate at the expense of 720p and 1080p clients due to the nature of TCP congestion control. A single realization of quality qV vs. time for three different clients in the cases of competitive, VSP-centralized, and VSP-distributed are shown in Figures 6.3a, 6.3b and 6.3c, respectively. In general, the variance of quality is smaller in collaborative streaming compared to that of the competitive streaming with the exception of 360p resolution (discussed above). We also observe that DASH clients with 720p and 1080p resolutions in the competitive systems experience more stall events. All collaborative systems show comparable results for DASH clients with 360p and 720p resolutions. Our experiments with other video indicate that these observations are independent of the content and bitrates.



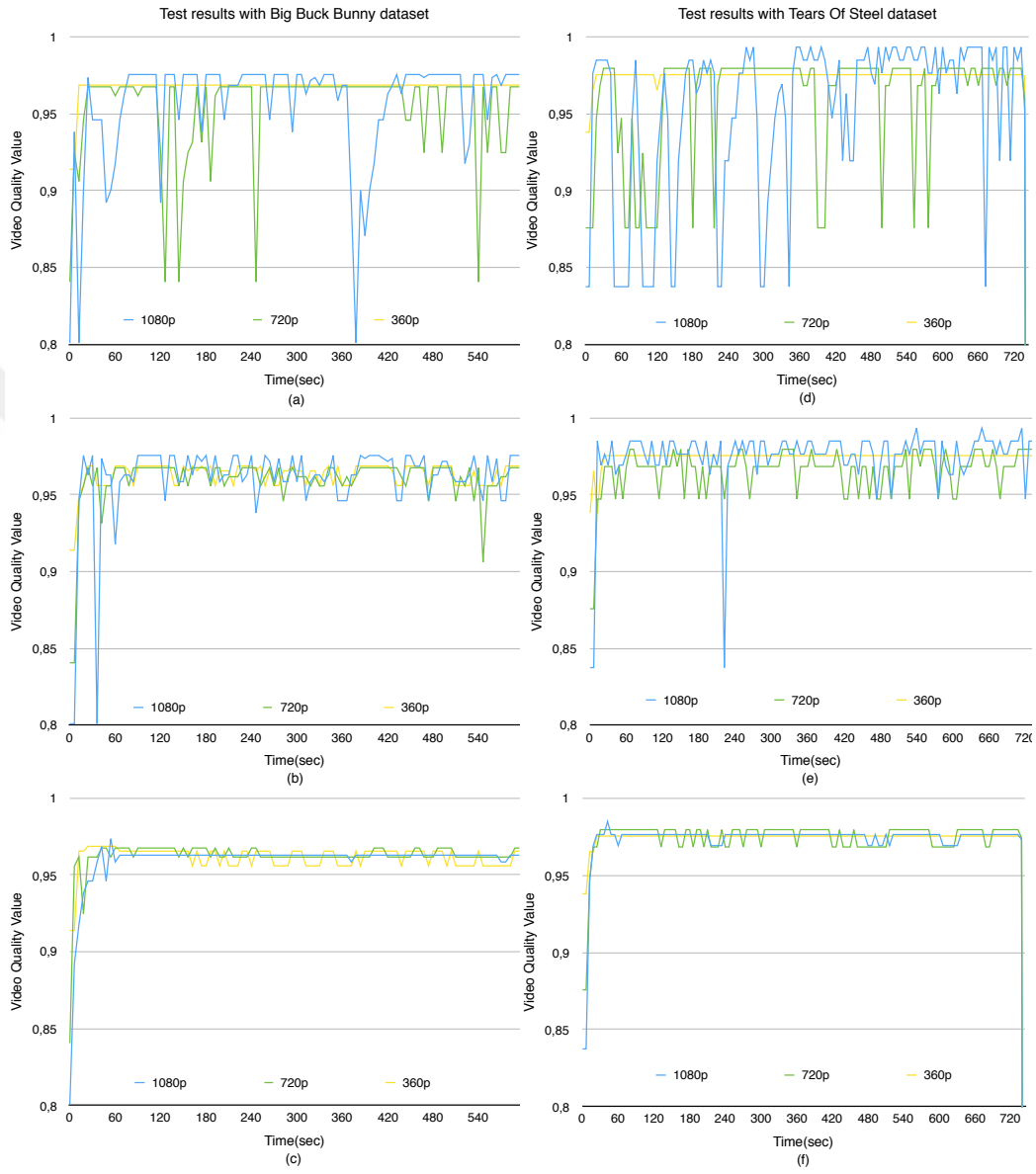
Şekil 6.3: Single realizations of competitive vs. collaborative streaming results for three heterogeneous clients using Tears of Steel: a) competitive, b) VSP-managed centralized collaboration, c) VSP-managed distributed collaboration.

Second, we present results for VSP-managed centralized and distributed systems versus competitive DASH using three heteregeneous clients with video resolutions 1080p, 720p and 360p, respectively, streaming *Big Buck Bunny* with 6 sec. segments over a reserved network slice. We stream over a slice with reserved capacity of 5 Mbps.

Tablo 6.3: Comparison for three heterogeneous clients using 'Tears of Steel' with 4 sec. segments.

Method	Resolution	Mean QV	Var QV	MNS	MDS
Competitive	1080p	0,9503	0,0005	8	3,5
	720p	0,9642	0,0003	4	2,3
	360p	0,9746	0,00001	0	0
SDN-DASH	1080p	0,9637	0,0001	5	3
	720p	0,9668	0,0002	4	1,7
	360p	0,9746	0,00001	0	0
VSP-Managed (Centralized)	1080p	0,9703	0,00002	1	1,6
	720p	0,9721	0,00006	0	0
	360p	0,9746	0,00001	0	0
VSP-Managed (Distributed)	1080p	0,9715	0,00002	1	1,5
	720p	0,9735	0,00005	0	0
	360p	0,9746	0,00001	0	0

In the VSP-managed centralized collaborative system, the fair share bitrates for the three clients are determined by the TEM unit of the NSP as 3,04 Mbps for 1080p, 1.3 Mbps for 720p and 0.65 Mbps for 360p clients. Clearly, how close we can achieve this qV value depends on how close are the determined fair share bitrate values to the available adaptation level bitrate sets of the particular content, since each DASH client starts streaming with the nearest adaptation level to its fair-share bitrate. Table 6.4 shows that the mean of quality values are closer to each other in VSP-Managed collaborative frameworks shown in Figures 6.4b and 6.4c which indicates they achieve fairness. In general, the variance of quality values of collaborative systems are less than that of the competitive system shown in 6.4a with the exception of 360p resolution, since in the case of competitive DASH clients, the 360p user gains an unfair advantage over the 720p and 1080p clients due to the aggressive and selfish nature of TCP congestion control. As a result, unlike both collaborative systems, DASH clients with 720p and 1080p resolutions in the competitive system experience stall events seen in table 6.4.



Şekil 6.4: Comparison of competitive, centralized-collaborative, and distributed-collaborative systems for three heteregeneous clients: a) competitive, b) VSP-managed centralized collaboration, c) VSP-managed distributed collaboration clients streaming Big Buck Bunny; d) competitive, e) VSP-managed centralized collaboration, f) VSP-managed distributed collaboration clients streaming Tears of Steel.

VSP-managed centralized collaborative system provides a better mean qV for the 1080p DASH client, but experiences stall. Both VSP-Managed collaborative systems show comparable results for DASH clients with 360p and 720p resolutions.

Tablo 6.4: Comparison for three heterogeneous clients using 'Big Buck Bunny' with 6 sec. segments.

Method	Resolution	Mean QV	Var QV	MNS	MDS
Competitive	1080p	0,9561	0,0013	4,5	3,7
	720p	0,9596	0,0007	2	2,1
	360p	0,9675	0,0005	0	0
VSP-Managed (Centralized)	1080p	0,9630	0,0009	1,5	1,9
	720p	0,9636	0,0003	0	0
	360p	0,9656	0,0008	0	0
VSP-Managed (Distributed)	1080p	0,9626	0,0003	0	0
	720p	0,9746	0,0003	0	0
	360p	0,9755	0,00006	0	0

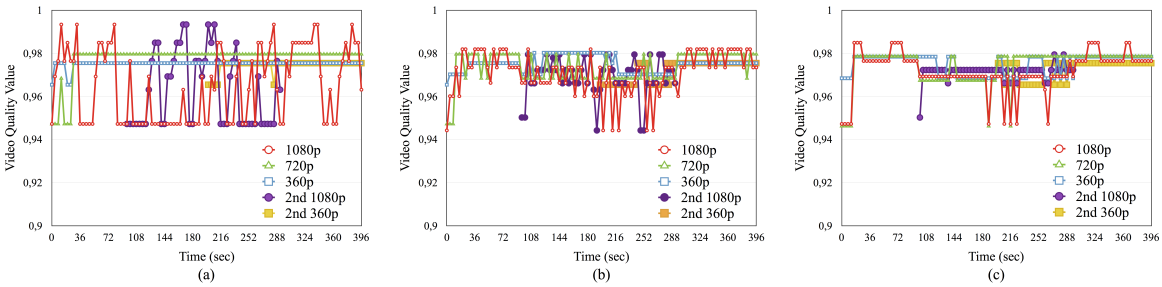
Third, we stream *Tears Of Steel* with 6sec chunks over a slice with capacity 8.5Mbps with the same three clients. In the VSP-managed centralized system, the fair-share bitrate values for the clients are 5.3 Mbps, 2.3 Mbps and 0.8 Mbps, respectively. We see that the distributed collaborative system achieves a mean qV value that is close to the computed value shown in table 6.5, which is an indicator of QoE fairness among the heterogeneous DASH clients. The quality values for the collaborative systems shown in Figure 6.4e and 6.4f both have lower variance compared to that of the competitive system shown in Figure 6.4d and the competitive clients experience high quality fluctations in case of 720p and 1080p clients shown in table 6.5. In the case of 360p client, the competitive system is as good as the collaborative systems since the 360p client gets an unfair share of the bitrate at the expense of 720p and 1080p clients. We see that these observations are independent of the content and bitrates.

In the second set of results, first we compare the performance of competitive vs. VSP-managed collaborative DASH clients in a dynamic scenario where clients enter/exit at random times. In this scenario, the reserved slice capacity for DASH flows is 10 Mbps, and we initially have 3 heterogeneous DASH clients that stream *Tears Of Steel* with 4 sec segments. In the VSP-Managed centralized collaboration, the fair-share bitrates (without considering client buffer state) are computed by the VSP as

Tablo 6.5: Comparison for three heterogeneous clients using 'Tears Of Steel' with 6 sec. segments.

Method	Resolution	Mean QV	Var QV	MNS	MDS
Competitive	1080p	0,9518	0,003	4,8	3,6
	720p	0,9597	0,0014	2	1,17
	360p	0,9746	0,00002	0	0
VSP-Managed (Centralized)	1080p	0,9765	0,00055	1,2	1,5
	720p	0,9741	0,00024	0	0
	360p	0,9746	0,00002	0	0
VSP-Managed (Distributed)	1080p	0,9754	0,0003	0	0
	720p	0,9759	0,0001	0	0
	360p	0,9746	0,00002	0	0

5.5 Mbps for 1080p, 2.9 Mbps for 720p and 1.5 Mbps for 360p to achieve a common quality $qV = 0.9832$. In the both centralized and distributed collaboration, when a new client enters or exits the new fair share bitrates are recomputed centrally or by group collaboration, respectively. After 100 sec, a new DASH client starts streaming the same content at 1080p resolution with 10 sec segments which decreases qV by 0.009 for all clients over the same slice. After 200 sec, another client starts streaming 360p resolution, which causes a further decrease in qV by 0.002, and the fair-share bitrates become 3.4 Mbps for 1080p, 1.8 Mbps for 720p, and 0.7 Mbps for 360p. One of the 1080p clients quits streaming at time 300 sec., which results in updating fair-share bitrates to 4.9 Mbps for 1080p, 2.6 Mbps for 720p, and 1.2 Mbps for 360p with a common $qV = 0.981$. We see from Figure 6.5(a) that competitive DASH clients experience qv fluctuations throughout a session since they are not aware of the dynamically entering/exiting DASH clients, while Figure 6.5(b-c) show that both central and distributed collaboration results in more stable qv values in time as they keep up with computed qV values throughout a session. In addition, we see from Table 6.6 that competitive 1080p clients suffer considerably from stall events. Furthermore, competitive 1080p clients experience undesired high video quality fluctuations, while collaborative 1080p clients provide much better stall results and smaller qv variance.



Şekil 6.5: Single realizations of competitive vs. collaborative streaming results when clients enter/exit randomly: a) competitive, b) VSP-managed central collaboration, c) VSP-managed distributed collaboration.

Tablo 6.6: Comparison when clients enter/exit randomly.

Method	Resolution	Mean QV	Var QV	MNS	MDS
Competitive	1080p	0,9651	0,0002	28	111,6
	720p	0,9778	0,00005	4	12,2
	360p	0,9752	0,00002	0	0
VSP-Managed (Centralized)	1080p	0,9709	0,00009	6	11,6
	720p	0,9733	0,0001	3	4,2
	360p	0,9749	0,00001	0	0
VSP-Managed (Distributed)	1080p	0,9719	0,00008	5	8,5
	720p	0,9742	0,00006	2	2,8
	360p	0,9750	0,00002	0	0

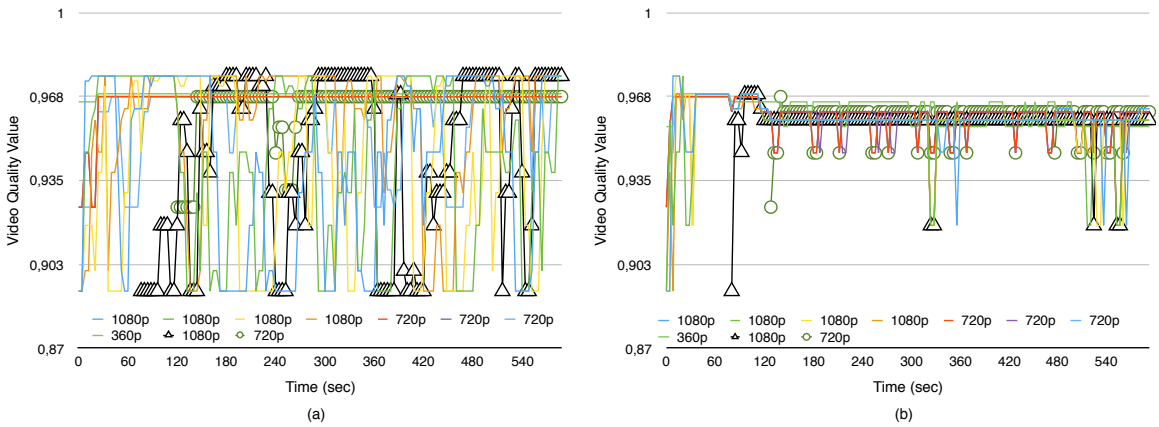
Second, since VSP-managed systems provides comparable results, we investigate the performance of VSP-managed distributed streaming and competitive DASH with large number of clients and varying network state using the content *Big Buck Bunny* with 4 second segments . In this experiment, we have a 17 Mbps network slice that is shared by eight DASH clients where four clients are 1080p DASH clients, three clients are 720p DASH clients, and one client is 360p DASH clients. All clients stream *Big Buck Bunny* where initial fair-share bitrates are 2.9 Mpbs for 1080p, 1.5 Mbps for 720p and 0.57 Mbps for 360p. An additional 1080p DASH client starts to stream same content after 90 seconds, and fair-share bitrates become 2.5 Mbps for 1080p, 1.3 Mbps for 720p and 0.48 Mbps for 360p. Finally, an additional 720p DASH client starts to stream same content after 135 seconds when eight DASH clients stream initially.

Tablo 6.7: The Result Of 10 Heterogeneous Clients Sharing a 17 Mbps Network Slice with streaming 'Big Buck Bunny'

Method	Resolution	Mean QV	Var QV	MNS	MDS
Competitive	1080p	0,9459	0,001	6,6	20,46
	720p	0,9630	0,00013	1,8	4,5
	360p	0,9684	0,0000004	0,8	0,4
VSP-Managed (Distributed)	1080p	0,9574	0,00016	2,2	3,52
	720p	0,9598	0,00005	0	0
	360p	0,9603	0,00002	0	0

The fair-share bitrates up-dated as 2.3 Mbps for 1080p, 1.2 Mbps for 720p, and 0.45 Mbps for 360p. In distributed collaboration framework, VSP-managed DASH client group is up-dated twice so that initial clients are aware of the new clients that starts to stream.

The fair quality value calculated by Fair-Share algorithm is equal to 0.9610. In Table 6.7, it is clearly seen that the mean qV values of VSP-managed distributed DASH clients are more closer to fair quality value. Default DASH with 720p and 360p clients has larger mean qV values because the equal partition of the shared link for these resolutions is larger than fair-share bitrate values in collaborative system.

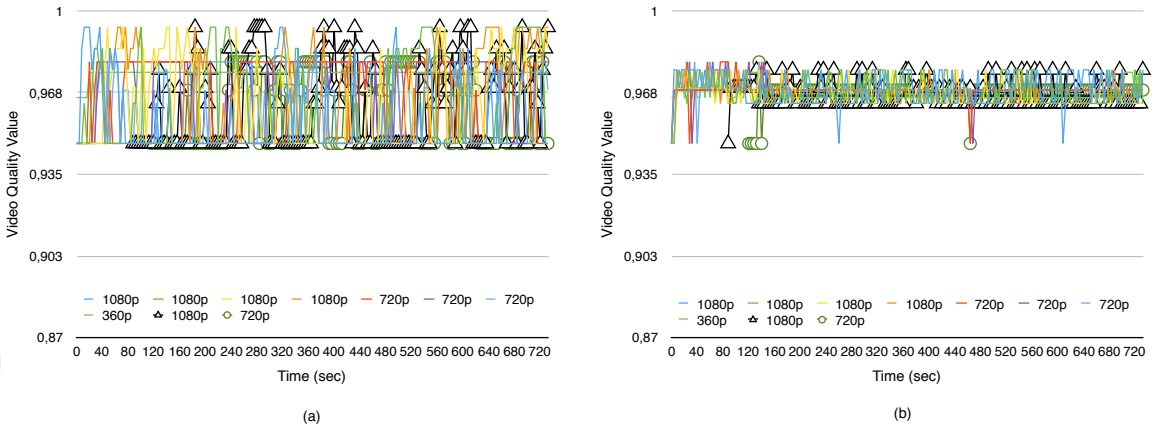


Şekil 6.6: Comparison of competitive and collaborative systems for ten heterogeneous clients: a) competitive, b) VSP-managed distributed collaboration clients streaming Big Buck Bunny

Tablo 6.8: The Result Of 10 Heterogeneous Clients Sharing a 21 Mbps Network Slice with streaming 'Tears of Steel'

Method	Resolution	Mean QV	Var QV	MNS	MDS
Competitive	1080p	0,9612	0,0003	11,2	40,32
	720p	0,9670	0,0002	4,2	11,34
	360p	0,9750	0,000003	2	1
VSP-Managed (Distributed)	1080p	0,9674	0,00003	3	4,8
	720p	0,9684	0,00001	2	0,8
	360p	0,9670	0,00001	0	0

However, our main concern is the QoE fairness among heterogeneous competing DASH clients and improvement of undesired events due to this competition, we can easily state that collaborative system improves QoE fairness related results and decrease the high quality fluctuations seen in Figure 6.6. Finally, we see from table 6.7 that collaborative framework outperforms competitive framework in stall related results. For this set of results, third we investigate the performance of VSP-managed distributed streaming and competitive DASH with large number of clients and varying network state using the content *Tears Of Steel* with 4 sec segments . In this experiment, we have eight DASH clients where four clients are 1080p DASH clients, three clients are 720p DASH clients, and one client is 360p DASH clients over network slice of 21 Mbps. All clients stream *Tears Of Steel* where fair share bitrates are equal to 3.6 Mpbs for 1080p, 1.8 Mbps for 720p and 0.75 Mbps for 360p. After 90 seconds, a new 1080p DASH client starts to stream same content, and fair share bitrates changes to 3.1 Mbps for 1080p, 1.6 Mbps for 720p and 0.61 Mbps for 360p. Finally, a new 720p DASH client starts to stream same content after 135 seconds when eight DASH clients stream initially. The fair share bitrates become 2.8 Mbps for 1080p, 1.5 Mbps for 720p, and 0.56 Mbps for 360p. VSP-managed distributed DASH client group is up-dated twice again so that initial clients adjust their TCP receive window and fair share bitrates.

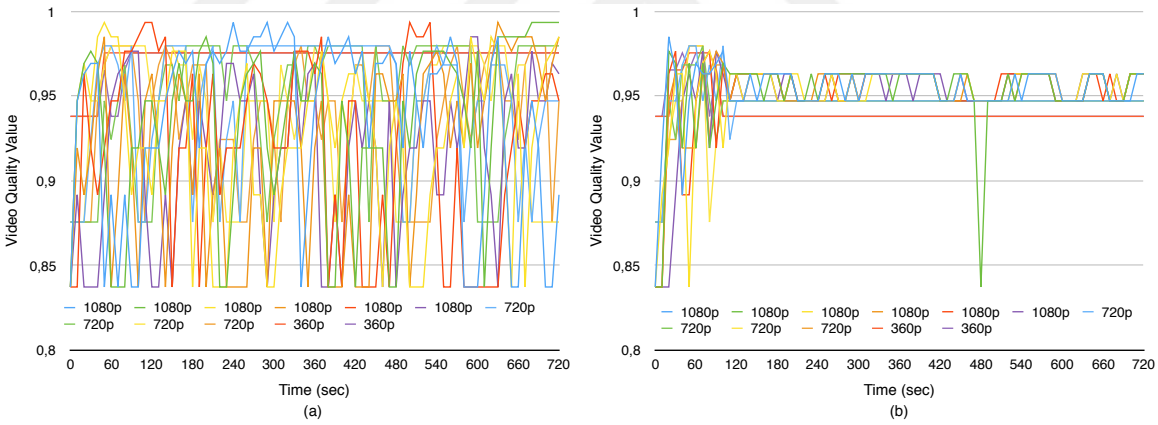


Şekil 6.7: Comparison of competitive and collaborative systems for ten heterogeneous clients: a) competitive, b) VSP-managed distributed collaboration clients streaming Tears Of Steel

The fair quality value calculated by Fair Share algorithm is equal to 0.9741 when there exist four 1080p DASH clients, three 720p DASH clients, and one 360p DASH clients . At the end of the streaming session, the fair quality value calculated by Fair-Share algorithm is equal to 0.9681. In Table 6.8, it is clearly seen that the mean qV values of VSP-managed DASH clients are more closer to fair quality value. Competitive DASH with 360p client has larger mean qV values because the equal partition of the shared link for these resolutions is larger than fair-share bitrate values in collaborative system. However, our main concern is the QoE fairness among heterogeneous competing DASH clients and improvement of undesired events due to this competition, we can easily state that collaborative framework improves QoE fairness related results and decrease the high quality fluctuations seen in Figure 6.7. Finally, we see from table 6.7 that collaborative framework outperforms competitive framework in stall related results.

In the third set of results, we consider “a pessimistic scenario" for VSP-managed distributed streaming with 12 heterogeneous clients, where 6 clients are 1080p, 4 clients are 720p, and two clients are 360p, in a group that share a low capacity network slice, where the slice capacity is set equal to the sum of the lowest available adaptation rates required to serve 12 users, i.e., the slice capacity is set equal to

21.5 Mbps. All clients stream *Tears Of Steel* with 10 second segments. Graph of a single realization of video quality vs. time for competitive vs. distributed-collaborative streaming clients is shown in Figure 6.8. We see even in this resource limited scenario that we achieve fairness, that is, there does not appear any critical difference between the mean quality values of different resolution DASH clients. Furthermore, in the competitive framework shown in Figure 6.8 (a), the variance of video quality of each DASH client is very large compared to the case of collaborative clients, and the mean video quality values given in Table 6.9 for different resolution flows are unacceptably low for 1080p DASH clients. In general, collaborative streaming achieves outstanding results compared to competitive streaming.



Şekil 6.8: Single realizations of competitive vs. collaborative streaming results for 12 heterogeneous clients: a) competitive, b) VSP-managed distributed collaboration.

In collaborative streaming, we observe adverse effect of limited slice resources in the case of 360p clients, where the fair-share bitrate is computed as 0.45 Mbps while the lowest available video adaptation level bitrate is 0.5 Mbps. As a result, it can be seen from Figure 6.8(b) that 360p clients experience lower overall mean qV values compared to 720p and 1080p clients. Competitive streaming achieves better quality values for 360p clients at the expense of 720p and 1080p clients as expected because of almost equal sharing of the slice capacity among heterogeneous clients.

In the fourth set of experiments, We perform two experiments, where in each experiment we have one 1080p client, one 720p client, and one 360p client. In the

Tablo 6.9: Comparison for 12 heterogeneous clients sharing a 21.5 Mpbs network slice

Method	Resolution	Mean QV	Var QV	MNS	MDS
Competitive	1080p	0,9255	0,0028	5,7	3,6
	720p	0,943	0,0014	3,4	2,24
	360p	0,972	0,0001	1,9	1,54
VSP-Managed (Distributed)	1080p	0,9530	0,0004	0,45	1,2
	720p	0,9467	0,0002	0	0
	360p	0,9411	0,00009	0	0

first experiment, all clients start streaming *Big Buck Bunny* with 4 sec. segments on a reserved slice with initial capacity of 5 Mbps, and where the fair-share bitrates are 2.8 Mbps for 1080p, 1.4 Mbps for 720p and 0.56 Mbps for 360p clients, respectively. After 120 seconds, the reserved slice capacity decreases to 4 Mbps, and the fair-share bitrates are recalculated as 2.2 Mbps for 1080p, 1.2 Mbps for 720p, and 0.43 Mbps for 360p clients, respectively. Then after 300 seconds, the NSP informs the VSP that the reserved slice capacity has increased to 6 Mbps. With 6 Mbps capacity, the fair-share bitrates become 3.4 Mbps for 1080p, 1.7 Mbps for 720p and 0.69 Mbps for 360p clients, respectively. Since the VSP informs its clients about the changes in the slice capacity, clients are able to adjust their fair-share bitrates and TCP receive-window size accordingly. The fair quality value calculated by Algorithm 2 is equal to 0.9613 when the capacity is equal to 4 Mbps, and it becomes 0.9727 when the capacity is equal to 6 Mbps. It can be seen from Table 6.10 and Figure 6.9b that collaborative DASH clients provide better QoE-fair mean qv values which are closer to each other, and in the distributed-collaborative system 1080p client does not suffer from lower mean qv value as the 1080p competitive DASH client suffers. Competitive DASH with 720p and 360p clients has larger mean qV values because the equal partition of the shared link for these resolutions is larger than fair share bitrate values in VSP-managed distributed method, but there is not any significant mean qv difference seen in Table 6.10 between 720p DASH clients in both method. However, 1080p competitive DASH client suffers high quality fluctuations which can be seen in Figure 6.9a. In general, VSP-managed distributed method does not experience any stall events seen in Table 6.10 except 1080p resolution which provides up to 35 times better the results when

Tablo 6.10: Three heterogeneous clients streaming Big Buck Bunny with 4 sec. segments over a variable capacity slice

Method	Resolution	Mean QV	Var QV	MNS	MDS
Competitive	1080p	0,9558	0,0005	7	17,5
	720p	0,9664	0,00005	1	0,75
	360p	0,9684	0,000004	0	0
VSP-Managed (Distributed)	1080p	0,9646	0,0001	2	0,5
	720p	0,9662	0,00002	0	0
	360p	0,9669	0,00001	0	0

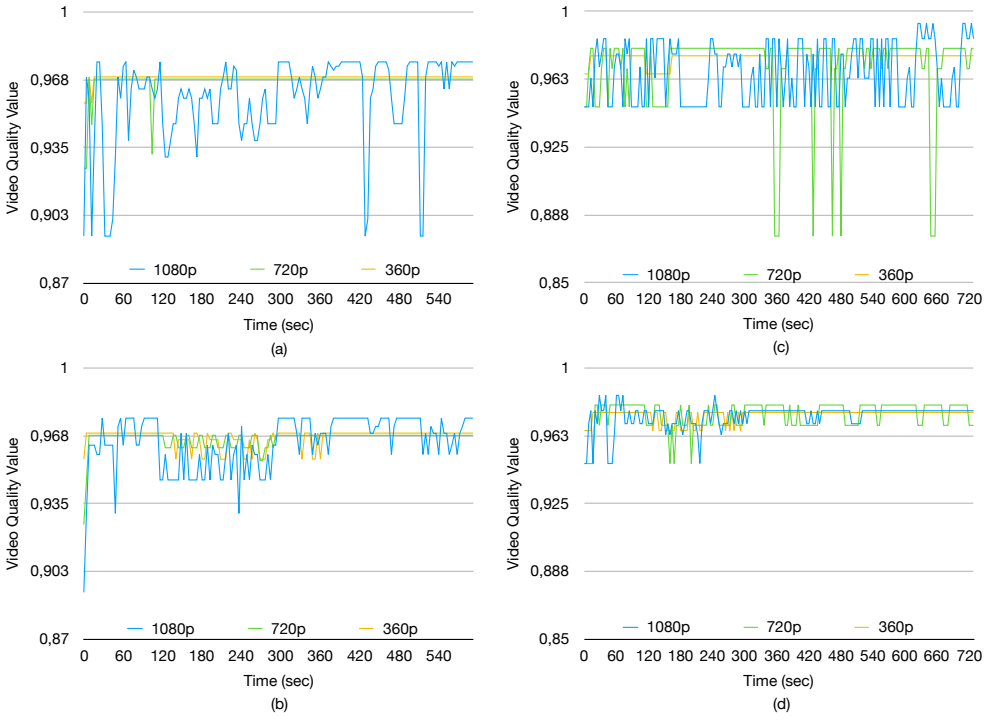
Tablo 6.11: Three heterogeneous clients streaming Tears Of Steel with 4 sec. segments over a variable capacity slice

Method	Resolution	Mean QV	Var QV	MNS	MDS
Competitive	1080p	0,9652	0,0002	12	46,92
	720p	0,9699	0,0004	8	18,48
	360p	0,9744	0,000009	0	0
VSP-Managed (Distributed)	1080p	0,9735	0,00004	4	3
	720p	0,9739	0,00006	2	0,8
	360p	0,9741	0,00001	0	0

MDS is considered. Moreover, the MNS value for 1080p resolution in competitive framework is 3.5 times higher than the VSP-managed distributed framework.

In the second experiment, all clients stream *Tears Of Steel* with 4 sec. segments on a reserved capacity of 7.5Mbps, and where fair share bitrates are 4.2 Mbps for 1080p, 2.2 Mbps for 720p and 0.98 Mbps for 360p. After 120 seconds, the reserved capacity for these clients is updated as 6.5 Mbps, and the fair-share bitrates become 3.7 Mbps for 1080p, 1.9Mbps for 720p, and 0.77 Mbps for 360p. Finally, there exists one final reserved capacity update which provides 8.5 Mbps for VSP clients after 300 seconds. In the final condition, the fair-share bitrates are 4.7 Mbps for 1080p, 2.5 Mbps for 720p and 1.1 Mbps for 360p. In the VSP-managed DASH, clients are informed about the variation in the network slice so that clients can modify their TCP receive window, and fair-share bitrates. During the change in the reserved capacity, the fair quality value calculated by Fair-Share algorithm is equal to 0.9747 when reserved capacity is

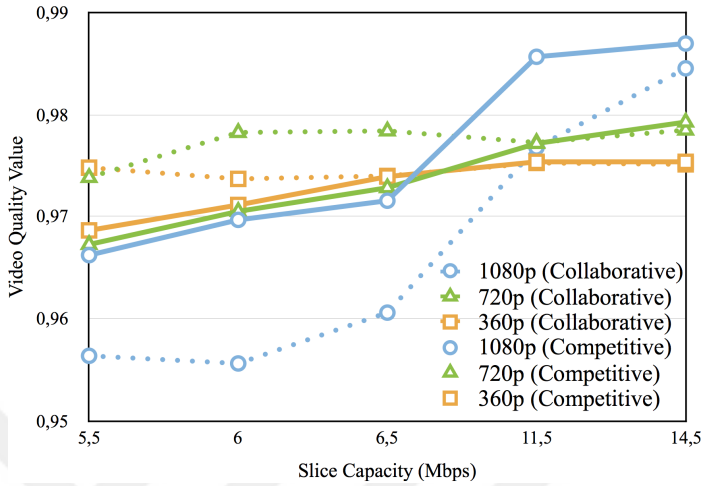
equal to 6.5Mbps, and this fair quality value becomes 0.9803 when reserved capacity is equal to 8.5Mbps. It is clearly seen in Table 6.11 that VSP-managed DASH clients has a mean qv more closer to fair share bitrate and among themselves. It can be seen from Table 6.11 that distributed-collaborative DASH clients with 720p resolution provide up to 22 times better results when MDS is considered, and competitive DASH clients experience 4 times higher MNS results which are shown in Table 6.11. Furthermore, it can be seen that competitive DASH clients with 1080p resolution experience 15 times higher MDS, and collaborative DASH clients with 1080p resolution provide up to 3 times better results when MNS is considered. Moreover, collaborative DASH clients outperform competitive DASH clients in the mean and variance of video quality values which can be seen in Table 6.11, and collaborative DASH achieves smoother quality value as can be seen in Figure 6.9d than competitive DASH results shown in Figure 6.9c.



Şekil 6.9: Comparison of competitive and collaborative DASH over a variable capacity slice: a) competitive and b) collaborative clients streaming Big Buck Bunny; c) competitive and d) collaborative clients streaming Tears Of Steel

In the fifth set of experiments, we consider two limiting cases: i) what happens as we increase the capacity of the reserved slice and ii) what happens when the DASH segment duration increases in competitive vs. collaborative streaming. To answer the first question: Inspection of Table II shows that we need at least 13.2 Mbps slice capacity to stream video to a 1080p, a 720p and 360p client at the highest quality level. In order to analyze the effect of increasing slice capacity on the comparison of competitive vs. collaborative heterogeneous streaming, we gradually increase the slice capacity from 5.5 Mbps up to 14.5 Mbps while keeping the number of the DASH clients fixed at three. Figure 6.10 shows achievable qv values for different resolution competitive (dotted lines) and collaborative (solid lines) clients. We see that in competitive streaming 360p client operates above its fair-share bitrate until the slice capacity reaches 6.5 Mbps and 720p client operates above its fair-share until the slice capacity reaches 11.5 Mbps at the expense of the 1080p client. We also observe in collaborative streaming that qv values for all three clients remain close until the slice capacity reaches 6.5 Mbps, showing that we can achieve fairness among clients when we have a limited capacity slice. We next observe that when we have sufficient slice capacity (exceeding 6.5 Mbps) to serve 360p and 720p clients at their highest quality levels (2.4 Mbps and 0.8 Mbps, respectively), then 1080p client starts receiving video at gradually higher bitrates (and hence quality). Finally, as the slice capacity increases such that we have unconstrained capacity, the quality levels achieved by competitive streaming converges to those of collaborative streaming. To answer the second question: we observe from Table 6.12 that as the segment duration increases from 2 sec. to 10 sec. there is considerable improvement in mean number and duration of stalls in competitive streaming; however, the results obtained by collaborative streaming are still significantly better than those of competitive streaming in both cases.

Overall, we see that the proposed collaborative architectures provide significant improvements in the mean and variance of video quality values as well as the number and duration of stall events compared to competing heterogeneous DASH clients. Although the fair quality value, hence fair-share bitrates for each client, is directly



Şekil 6.10: Convergence of video quality in competitive vs. collaborative streaming as the slice capacity increases.

Tablo 6.12: Analysis of the effect of DASH segment duration.

Method	**MNS**		**MDS**	
	Segment Duration		**Segment Duration**	
	2 sec.	**10 sec.**	**2 sec.**	**10 sec.**
Competitive	28.2	4	18	3
VSP-Managed (Centralized)	4.5	0	0.8	0.

related to the reserved slice capacity and the number of heterogeneous DASH clients in a reserved slice, we can see from the results that it is also important that the encoding rates of the available adaptation levels match the fair-share bitrates for heterogeneous clients in order to achieve the maximum possible fair quality value for all clients. Furthermore, the adjustment of the TCP receive window size is a key factor to prevent rate instability per client and maximize utilization of reserved resources. Finally, comparing the cases of 2 sec. and 10 sec. segments, we observe that QoE variability increases when shorter duration segments are employed in the competitive (default TCP) clients.

Chapter 7

CONCLUSION

In this thesis, we introduced the three way interaction between QoS (throughput reservation) at network level and TCP congestion control at transport level and DASH quality level adaptation at the application level. Firstly, we have analyzed the interaction between resource reservation and TCP. Secondly, we have presented the VSP-Centric architectures that are centralized and distributed collaboration frameworks. Thirdly, we compare the performance of proposed architectures with standard DASH. The main contributions and findings of this thesis can be summarized as follows:

- The proposed architecture is practical and feasible since future networks deploying SDN can be successfully reconfiguring the network resources dynamically and performing traffic engineering.

- Most of the studies in literature proposes QoS provisioning for DASH services over future networks deploying SDN without consideration of TCP rate control. Due to the mismatch between TCP and DASH, this QoS provisioning is not sufficient to i) minimize QoE variability per client, ii) maximize QoE fairness among clients, and iii) maximize utilization of the reserved slice capacity. To this end, we implement SDN-assisted centralized and distributed architectures for collaboration between NSP, VSP, and DASH clients for managing QoE variability by TCP ACK pacing in order to regulate the TCP rate about the fair-share rate, cutting unnecessary packet losses and TCP rate fluctuations.

- In literature, most of the studies have been proposed similar architectures for centralized collaboration between the NSP, VSP and clients. The concept of

inter-client (distributed) collaboration is a novel approach to achieve QoE fairness between heteregenous DASH clients. Furthermore, the proposed dynamic grouping of clients provides robustness of collaboration against changing number of clients sharing the same link as well as changing network conditions. Experiments show that both types of collaboration provide smooth QoE with less quality variations in smaller amounts and almost zero freezes compared to the case of competitive DASH clients.

BIBLIOGRAPHY

[1] "Information technology - MPEG systems technologies - part 6: Dynamic adaptive streaming over http (dash)," ISO/IEC, MPEG Draft International Standard, 2011.

[2] S. Akhshabi, L. Anantakrishnan, A.C. Begen, and C. Dovrolis, "What happens when http adaptive streaming players compete for bandwidth?" in ACM Proc. of Int. Workshop on Network and Operating System Support for Digital Audio and Video, 2012, pp. 9-14.

[3] R. Kuschnig, I. Kofler, and H. Hellwagner, "An evaluation of TCP-based rate-control algorithms for adaptive Internet streaming of H.264/SVC," in Proc. Annual ACM Conf. MMSys, Phoenix, AZ, USA, 2010, pp. 157–168.

[4] R. Mok, E. Chan, and R. Chang, "Measuring the quality of experience of HTTP video streaming," in Proc. IEEE IFIP IM (pre-conf.), 2011.

[5] M. Seufert, S. Egger, M. Slanina, T. Zinner, T. Hobfeld, and P. Tran-Gia, "A survey on quality of experience of HTTP adaptive streaming," IEEE Communication Surveys and Tutorials, vol. 17, no. 1, pp. 469 - 492, Sep. 2014.

[6] O. Oyman and S. Singh, "Quality of Experience for HTTP Adaptive Streaming Services", IEEE Communications Magazine, no. 4, 2012.

[7] C. Timmerer, M. Maiero, B. Rainer, S. Petscharing, D. Weinberger, C. Mueller, and S. Lederer, "Quality of experience of adaptive HTTP streaming in real-world environments," IEEE Comsoc MTC E-Letter, vol. 10, no. 3, May 2015.

[8] Z. Wang, L. Lu, and A. C. Bovik, "Video quality assessment based on structural

distortion measurement," Signal Processing: Image Communication,special issue on objective video quality metrics, vol. 19, Jan. 2004.

[9] M. Barbera, A. Lombardo, C. Penarello, and G, Schembra, "Active window management, an efficient gateway mechanism for TCP traffic control", IEEE Conf. on Communications (ICC), 2007.

[10] J. Jiang, V. Sekar, and H. Zhang, "Improving fairness, efficiency, and stability in HTTP-based adaptive video streaming with FESTIVE," in Int. Conf. on Emerging Networking Experiments and Technologies, France, 2012, pp. 97-108.

[11] Y. Sun, C.C. Lee, R. Berry, and A. Haddad, "A load-adaptive ACK pacer for TCP traffic control," Proc. of 40th Allerton Conf. on Communication, Control, and Computing, 2002

[12] A. Sideris, et. al., "QoE fairness, HTTP adaptive streaming and IDVB-T: The AWM approach," Int. Conf. on Telecommunications and Multimedia (TEMU), 2014.

[13] A. Mansy, M. Fayed, and M. Ammar, "Network-layer fairness for adaptive video streams", in IFIP Networking Conf., 2015.

[14] R. Houdaille and S. Gouache, "Shaping HTTP adaptive streams for a better user experience," in Proc. 3rd Ann. ACM Conf. MMSys, Chapel Hill, NC, USA, 2012, pp. 1–9.

[15] "Software-defined networking: The new norm for networks," Open Networking Fundation (ONF) White Paper, 2012.

[16] M. Karakus and A. Duresi, "Quality of service in software defined networking: A survey," Jour. of Network and Computer Applications, 2016.

[17] D. Palma, J. Gonçalves, B. Sousa, L. Cordeiro, P. Simoes, S. Sharma, and D. Staessens, “The QueuePusher: Enabling queue management in OpenFlow," European Workshop on Software Defined Networks (EWSDN), 2014.

[18] P. Georgopoulos, Y. Elkhatib, M. Broadbent, M. Mu, and N. Race, “Towards network-wide QoE fairness using openflow-assisted adaptive video streaming," in ACM SIGCOMM, New York, 2013, pp. 15-20.

[19] H. Nam, K. Kim, J. Kim and H. Schulzrinne, “Towards QoE-aware video streaming using SDN," in Global Communications Conf., 2014.

[20] E. Liotou, G. Tseliou, K. Samdanis, D. Tsolkas, F. Adelantado and C. Verikoukis, “An SDN QoE-service for dynamically enhancing the performance of OTT applications", in Quality of Multimedia Experience, 2015.

[21] S. Ramakrishnan, X. Zhu, F. Chan, and K. Kambhatla, “SDN-based QoE optimization for HTTP-based adaptive video streaming", in Int. Symp. on Multimedia, 2015.

[22] S. Petrangeli, J. Famaey, M. Claeys, S. Latre, and F. de Turck, “QoE-driven rate adaptation heuristic for fair adaptive video streaming," ACM Trans. Multimedia Comput. Comm. App., vol. 12, no. 2, Oct. 2015.

[23] J. Kleinrouweler, S. Cabrero, and P. Cesar, “Delivering stable high-quality video: an SDN architecture with DASH assisting network elements," in Int. Conf. on Multimedia Systems, Austria, 2016.

[24] M. Mu, M. Broadbent, A. Farshad, N. Hart, D. Hutchison, Q. Ni, and N. Race, “A scalable user fairness model for adaptive video streaming over SDN-assisted future networks", IEEE Jour. on Selected Areas in Communications, vol. 34, no. 8, pp. 2168-2184, 2016.

[25] "Software-Defined Networking (SDN) Definition - Open Networking Foundation". Opennetworking.org. Web.

[26] D. Kreutz et al., "Software-defined networking: A comprehensive survey," Proc. IEEE, vol. 103, no. 1, pp. 14–76, Jan. 2015.

[27] F. Hu, Q. Hao, and K. Bao, "A Survey on software-defined network and OpenFlow: From concept to implementation," IEEE Commun. Surveys Tuts., vol. 16, no. 4, pp. 2181–2206, 4th Quart. 2014.

[28] Mueller, Christopher. "MPEG-DASH Dynamic Adaptive Streaming Over HTTP". Bitmovin. N.p., 2017. Web. 21 Mar. 2017.

[29] Stockhammer, Thomas. "MPEG's Dynamic Adaptive Streaming Over HTTP (DASH) - An Enabling Standard For Internet TV". Presentation.

[30] "DASH Streaming | Encoding.Com". Encoding.com. N.p., 2017.

[31] "ITEC – Dynamic Adaptive Streaming Over HTTP". www-itec.aau.at. Web.

[32] Aweya, James et al. "Enhancing Network Performance With TCP Rate Control". Global Telecommunications Conference, 2000. GLOBECOM '00. IEEE. IEEE, 2000. 1712-1718.

[33] Kurose, James F and Keith W Ross. Computer Networking. 1st ed. Harlow: Pearson Education, 2012.

[34] Mueller. "ITEC DASH". Slideshare.net. N.p., 2012. Web. 19 Mar. 2012.

[35] Yin, Xiaoqi et al. "A Control-Theoretic Approach For Dynamic Adaptive Video Streaming Over HTTP". SIGCOMM. 2015. Print.

[36] Kumar, M. Venkata Phani, K. C. Ravi, and Sudipta Mahapatra. "Design And Implementation Of A Dynamic Adaptive Video Streaming System With A Buffer Aware Rate Selection Algorithm". ICIAP 2015. 2015. Print.

[37] "OF-Config 1.2." [Online]. Available: https://www.opennetworking.org/images/stories/downloads/sdn- resources/onf- specifications/ openflow-config/of- config- 1.2.pdf

[38] B. Pfaff and B. Davie, "The open vswitch database management protocol," 2013.

[39] "Open vSwitch: An Open Virtual Switch." [Online]. Available:http://openvswitch.org

[40] "Mininet Overview." [Online]. Available: http://mininet.org/overview

[41] "Floodlight Controller." [Online]. Available: http://www.projectfloodlight.org/floodlight

[42] "G. D. Corporation, "General Algebraic Modeling System (GAMS)" [Online]. Available: http://www.gams.com

[43] K. T. Bagci, K. E. Sahin, and A. M. Tekalp, "Queue-allocation optimization for adaptive video streaming over software defined networks with multiple service-levels," in IEEE Int. Conf. on Image Proce. (ICIP), 2016, pp. 1519-1523.

VITA

KEMAL EMRECAN SAHIN was born in Corlu, Turkey on 20th August. He received his B.Sc degree in Electrical and Electronics Engineering and Physics from Koc University, Istanbul, Turkey in 2015. He joined M.Sc program in Electrical and Electronics Engineering at Koc University in 2015 as a research and teaching assistant. During his graduate studies he worked on adaptive video streaming, quality of experience, and software defined networking. He has co-authored two conference papers in IEEE Communications and Electronics(ICCE), 2016 and IEEE Internation Conference on Image Processing(ICIP), 2016. He also has a conference paper and journal paper under submission.