

T.C.
EGE ÜNİVERSİTESİ
Fen Bilimleri Enstitüsü

YAZILIM TANIMLI AĞLARDA UYARLAMALI MEDYA
DAĞITIMI İÇİN METODLAR

Mustafa TÜRK

DOKTORA TEZİ

Danışman: Doç. Dr. Müge SAYIT

Uluslararası Bilgisayar Anabilim Dalı
Bilgi Teknolojileri

İzmir
2024

Mustafa Türk tarafından doktora tezi olarak sunulan “Yazılım Tanımlı Ağlarda Uyarlamalı Medya Dağıtımı İçin Metodlar” başlıklı bu çalışma EÜ Lisansüstü Eğitim ve Öğretim Yönetmeliği ile EÜ Fen Bilimleri Enstitüsü Eğitim ve Öğretim Yönergesi’nin ilgili hükümleri uyarınca tarafımızdan değerlendirilerek savunmaya değer bulunmuş ve 24 Aralık 2024 tarihinde yapılan tez savunma sınavında aday oybirliği/oyçokluğu ile başarılı bulunmuştur.

Jüri üyeleri

İmza

Jüri Başkanı : Doç. Dr. Müge SAYIT

Raportör Üye : Dr. Öğr. Üyesi Gül BOZTOK ALGIN

Üye : Prof. Dr. Mehmet Emin DALKILIÇ

Üye : Prof. Dr. Ali Cengiz BEĞEN

Üye : Dr. Öğr. Üyesi Kübra Kalkan ÇAKMAKÇI

EGE ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ

ETİK KURALLARA UYGUNLUK BEYANI

EÜ Lisansüstü Eğitim ve Öğretim Yönetmeliğinin ilgili hükümleri uyarınca Doktora Tezi olarak sunduğum “Yazılım Tanımlı Ağlarda Uyarlamalı Medya Dağıtımı İçin Metodlar” başlıklı bu tezin kendi çalışmam olduğunu, sunduğum tüm sonuç, doküman, bilgi ve belgeleri bizzat ve bu tez çalışması kapsamında elde ettiğimi, bu tez çalışmasıyla elde edilmeyen bütün bilgi ve yorumlara atıf yaptığımı ve bunları kaynaklar listesinde usulüne uygun olarak verdiğimi, tez çalışması ve yazımı sırasında patent ve telif haklarını ihlal edici bir davranışımın olmadığını, bu tezin herhangi bir bölümünü bu üniversite veya diğer bir üniversitede başka bir tez çalışması içinde sunmadığımı, bu tezin planlanmasından yazımına kadar bütün safhalarda bilimsel etik kurallarına uygun olarak davrandığımı ve aksinin ortaya çıkması durumunda her türlü yasal sonucu kabul edeceğimi beyan ederim.

24/12/2024

Mustafa Türk

ÖZET

YAZILIM TANIMLI AĞLARDA UYARLAMALI MEDYA
DAĞITIMI İÇİN METODLAR

TÜRK, Mustafa

Doktora Tezi, Uluslararası Bilgisayar Anabilim Dalı

Tez Danışmanı: Doç. Dr. Müge SAYIT

24 Aralık 2024, 94 sayfa

Bu tezde yazılım tanımlı ağ ortamında uyarlamalı video dağıtımı için optimizasyon yöntemleri geliştirilmiştir. Geliştirilen yöntemler ile video kalitesi artırılırken kalite değişim sayısı/derinliği ve duraksamalar azaltılarak optimuma yakın kullanıcı deneyim kalitesi sağlanmıştır.

Optimizasyon probleminin modellenmesinde, ardışık olarak çalıştırılan üç optimizasyon modeli ve istisnai durumlar için kullanılan bir optimizasyon modeli geliştirilmiştir. Bu optimizasyon modellerinde Doğrusal Programlama, Karma Tamsayılı Doğrusal Programlama ve Tamsayılı Programlama yöntemleri kullanılmıştır.

Geliştirilen yöntemleri destekleyecek şekilde istemcilerden tampon bellek (buffer) durumu alınarak olası duraksamaların önüne geçilmiş ve kullanıcılar arasındaki kalite deneyiminin daha adil ve daha az kalite değişimi ile düzgün bir şekilde dağıtılması sağlanmıştır. Benzer şekilde istemciler için belirlenen sunucu ve video kalite bilgisi istemcilere iletilmektedir.

Anahtar sözcükler: Uyarlamalı video dağıtımı (HAS), deneyim kalitesi (QoE), yazılım tanımlı ağ (SDN), doğrusal programlama (LP), karma tamsayılı doğrusal programlama (MILP), tamsayılı programlama (IP), Ortak Medya İstemci Verisi (CMCD), Ortak Medya Sunucu Verisi (CMSD), Ölçeklenebilir Video Kodlama (SVC).

ABSTRACT

METHODS FOR ADAPTIVE MEDIA DELIVERY FOR SDN

TÜRK, Mustafa

Ph.D. in International Computer Department

Supervisor: Assoc. Prof. Müge SAYIT

24 December 2024 , 94 pages

In this thesis, optimization methods for adaptive video distribution in a software-defined networking environment have been developed. With the developed methods, while video quality is increased, the number and depth of quality changes and stalls are reduced, providing a user experience quality close to optimal.

In the modeling of the optimization problem, three optimization models executed sequentially and an additional optimization model used for exceptional cases have been developed. These optimization models employ Linear Programming, Mixed Integer Linear Programming, and Integer Programming methods.

In support of the developed methods, buffer status is taken from clients to prevent possible stalls, and the quality experience among users has been distributed more fairly and smoothly with less quality change. Similarly, the determined server and video quality information for the clients is communicated to them.

Keywords: HTTP-based Adaptive Streaming (HAS), Quality of Experience (QoE), Software-Defined Network (SDN), Linear Programming (LP), Mixed-Integer Linear Programming (MILP), Integer Programming (IP), Common Media Client Data (CMCD), Common Media Server Data (CMSD), Scalable Video Coding (SVC).

ÖNSÖZ

Bu tez çalışması, benim için zorlu fakat bir o kadar da ufkumu açan faydalı bir deneyim oldu. Tam yeterliliği bitirip tez konusu belirleme aşamasında çalıştığım kurumda tayinimin çıkması benim için müthiş bir hayal kırıklığı oldu. Bu süreçteki adaptasyon sorunlarım ve tez konusu araştırmaya odaklanamamam nedeniyle tez danışmanı ve konusunu değiştirmek zorunda kaldım. Bu zor süreç ardından kendimi daha yatkın hissettiğim bilgisayar ağları alanında çalışma fırsatı buldum. Ancak burada da daha önce adımı dahi duymadığım yeni konularla tanıştım. Tabii bu süreç epey zorlu geçti. O zamanlar incelediğim taslak çalışmanın ilk sayfasını dahi anlamamıştım. Ancak sabırla ve azimle çalışarak bu süreci de atlatmayı başardım. Sadece tez çalışmasıyla değil hayatta da bir çok sıkıntılarla karşılaştım bu süreçte ve bu durum insanlığın ne kadar önemli olduğunu bana bir kez daha gösterdi. Tabii hayatta sadece zorluklar yok. Hayatımızı cennete çevirebilecek gelişmeler de yaşanabiliyor. Bu süreçte dünya tatlısı iki tane çocuğumun dünyaya gelmesiyle yaşadığım mutluluğu anlatmam mümkün değil. Bu yeni durumun getirdiği sorumluluk ve iş yükünün itici gücüyle ve inancım sayesinde, tükenmişlik hissine kapılmadan, yılmadan usanmadan çalışmaya devam ederek tez çalışmamı tamamlayabildim. Bu çalışmayla bilim dünyasına doğru küçük bir adım attığımı düşünüyorum ve emekleyerek hatta sürünerek başladığım bu serüveni uçarak tamamlamak istiyorum. Dünya çapında önemli çalışmalara imza atma ümidiyle akademik yolculuğumu keyifli bir şekilde sürdürmeyi planlıyorum. Ümitlerin hep yeşil kaldığı bir dünya temennisiyle...

İZMİR

24/12/2024

Mustafa TÜRK

İÇİNDEKİLER

Sayfa

İÇ KAPAK	ii
KABUL VE ONAY SAYFASI	iii
ETİK KURALLARA UYGUNLUK BEYANI	v
ÖZET	vii
ABSTRACT	ix
ÖNSÖZ	xi
İÇİNDEKİLER	xiii
ŞEKİLLER DİZİNİ	xvi
TABLolar DİZİNİ	xix
SİMGELER VE KISALTMALAR DİZİNİ	xxi
1 GİRİŞ	1
2 TEKNOLOJİ VE YÖNTEMLER	4
2.1 Ölçeklenebilir Video Kodlama (Scalable Video Coding - SVC)	4
2.2 Yazılım Tanımlı Ağ (Software Defined Network – SDN) ve ONOS	5
2.3 MiniNet	8
2.4 Ortak Medya İstemci/Sunucu Verisi (Common Media Client/Server Data (CMC/SD))	9
2.5 IBM ILOG CPLEX Optimizer	10
2.6 Segment Boyutunu Kullanarak Dağıtım Uyarlaması	10
2.7 Doğrusal Programlama (Linear Programming - LP)	13
2.8 Tam Sayılı Programlama (Integer Programming)	15
2.9 İkili Programlama (Binary Programming - BP)	17
2.10 Karma Tamsayılı Doğrusal Programlama (MILP)	19
2.11 Bender'in Ayrıştırma Yöntemi(Bender's Decomposition)	20
2.12 Γ Sağlam (Γ Robust) Optimizasyon Yöntemi	22
3 LİTERATÜR ARAŞTIRMASI	23
3.1 Ağ Destekli Uyarlamalı Video Dağıtım Yöntemi	25
3.2 Ağ Destekli Uyarlamalı Katmanlı Video Dağıtım Yöntemi	28

3.3	Ortak Medya İstemci/Sunucu Verisi (Common Media Client/Server Data - CMC/SD) Uyumlu Çalışmalar	31
4	OPTİMİZASYON PROBLEMİNİN MODELLENMESİ	32
4.1	İlk Modelleme	32
4.2	Optimizasyon Probleminin Etkin Bir Şekilde Modellenmesi ve Çözümlemesi	36
4.2.1	Ağ topolojisi ve bant genişliğinin etkin kullanımı	39
4.2.2	Optimizasyon modelleri	43
4.2.3	Optimizasyon Algoritması	58
5	PERFORMANS DEĞERLENDİRMESİ	62
5.1	FROG'un Değerlendirilmesi	63
5.2	FROG ve Diğer İstemcilerin Karşılaştırılması	70
6	SONUÇ	84
	KAYNAKLAR DİZİNİ	86
	TEŞEKKÜR	93
	ÖZGEÇMİŞ	94

ŞEKİLLER DİZİNİ

<u>Şekil</u>	<u>Sayfa</u>
2.1 SDN Mimarisi (Open Networking Foundation (2011))	7
2.2 MiniNet Sanal Ağ Oluşturma Komutu (Mininet Team (2024)) . .	9
2.3 Ethernet Frame (Wikipedia Contributors (2024))	13
4.1 Zaman dilimleri	38
4.2 Zaman dilimleri ve etkin bant genişliği kullanımı	39
4.3 Ağ Topolojisi	41
4.4 Hat kapasitesinin etkin kullanımı	43
5.1 FROG’da istemcilerin ortalama video kalitelerinin ortalaması	64
5.2 FROG’da istemcilerin toplam duraksama sürelerinin ortalaması . .	66
5.3 FROG’da istemcilerin toplam video kalite değişim derinliğinin ortalaması	68
5.4 FROG’un ortalama kalite deneyimi (QoE) değerleri	68
5.5 FROG’un ortalama çalışma süreleri	69
5.6 FROG’da çalışma süresi aşımalarının toplamı	70
5.7 Ortalama video başlatma süresi	71
5.8 1x1y’nin BBB video seti için QoE karşılaştırması	72
5.9 1x1y’nin ED video seti için QoE karşılaştırması	73
5.10 1x1y yapılandırmasında BBB video seti ile yapılan testte istemci- lerin toplam duraksama sürelerinin ortalaması	74
5.11 1x1y yapılandırmasında ED video seti ile yapılan testte istemcilerin toplam duraksama sürelerinin ortalaması	74
5.12 Optimizasyon araçlarının çalışma süresi aşımalarının toplamı	75
5.13 50 istemci ile yapılan teslerde optimizasyon aracı süre aşımı ve istemci duraksama ilişkisi	75
5.14 30 istemci ile yapılan teslerde optimizasyon aracı süre aşımı ve istemci duraksama ilişkisi	76
5.15 20 istemci ile yapılan teslerde optimizasyon aracı süre aşımı ve istemci duraksama ilişkisi	76

5.16 1x1y'de BBB SVC seti için istemcilerin ortalama video kalitelerinin ortalaması	78
5.17 1x1y'de ED SVC seti için istemcilerin ortalama video kalitelerinin ortalaması	79
5.18 1x1y'de BBB SVC seti için istemcilerin toplam video kalite değişim derinliğinin ortalaması	79
5.19 1x1y'de ED SVC seti için istemcilerin toplam video kalite değişim derinliğinin ortalaması	80
5.20 Optimizasyon araçlarının ortalama çalışma süreleri	81
5.21 4x1y'de istemcilerin QoE değerinin ortalaması	82
5.22 4x1y'de istemcilerin ortalama video kalitelerinin ortalaması	83
5.23 4x1y'de istemcilerin toplam video kalite değişim derinliğinin ortalaması	83
5.24 4x1y'de istemcilerin toplam duraksama sürelerinin ortalaması	84

TABLOLAR DİZİNİ

<u>Çizelge</u>	<u>Sayfa</u>
2.1 Karşılaştırmalı Dosya Büyüklükleri	12
4.1 Notasyonlar	46
5.1 Video setlerinin ortalama dosya büyüklükleri (Mb)	67
5.2 <i>FROG</i> ve S^2VC' 'nin QoE Performans Karşılaştırması	77
5.3 <i>FROG</i> ve S^2VC' Çalışma Sürelerinin Karşılaştırması	81

SİMGELER VE KISALTMALAR DİZİNİ

<u>Simgeler</u>	<u>Açıklama</u>
<i>SDN</i>	Yazılım Tanımlı Ağ - Software Defined Network
<i>QoE</i>	Deneyim Kalitesi - Quality of Experience
<i>HAS</i>	Uyarlamalı Video Dağıtımı - HTTP Adaptive Streaming
<i>LP</i>	Doğrusal Programlama - Linear Programming
<i>MILP</i>	Karma Tamsayılı Doğrusal Programlama
<i>IP</i>	Tamsayılı Programlama - Integer Programming
<i>CMCD</i>	Ortak Medya İstemci Verisi - Common Media Client Data
<i>SVC</i>	Ölçeklenebilir Video Kodlama - Scalable Video Coding
<i>AVC</i>	Gelişmiş Video Kodlama - Advanced Video Coding
<i>CDN</i>	İçerik Dağıtım Ağı - Content Delivery Network

1 GİRİŞ

Video akışlandırma uygulamaları, İnternet uygulamaları arasında en yüksek trafiği yaratan, en popüler ve yaygın uygulamalardan birisidir (Global Internet Performance Report (2024)). Bu sebeple, video akışlandırma uygulamalarının performansının geliştirilmesi ve kullanıcı memnuniyetinin maksimize edilmesi amacıyla hem akademik literatürde hem de ticari alanda bir çok uygulama ve çalışma gerçekleştirilmiştir. Tez çalışmasının amacı, Yazılım Tanımlı Ağ (Software Defined Network - SDN) teknolojisinden yararlanarak, HTTP uyarlamalı video dağıtımının (HTTP Adaptive Streaming - HAS) kullanıcı deneyim kalitesinin iyileştirilmesidir. Bu amaçla, ağın yönetimi için yeni metotlar geliştirilmiş, uygulanmış ve geniş kapsamlı testler yapılmıştır.

HTTP uyarlamalı video dağıtım yönteminde, video dosyalarının farklı kalitelerdeki versiyonları medya sunucuda tutulmaktadır. Bu kalite versiyonlarına gösterim (representation) adı verilmekte ve bu gösterimler segment adı verilen, uzunlukları genellikle 2-10 saniye aralığında belirlenebilen ve tek başına kodu çözülebilen (decode edilebilen) küçük parçalara ayrılarak kodlanmaktadır (encode). Her bir gösterim, belirli bir bit hızında ve çözünürlükte kodlanmıştır. Kodlanmış video gösterimlerinin ortalama veri aktarım hızları, çözünürlükleri, oynatma süreleri ve diğer meta verileri, Media Presentation Description (MPD) adı verilen bir manifesto dosyasında yer almaktadır. İstemciler, MPD dosyasında listelenen video dosyaları için HTTP talebi göndererek video parçalarını indirip belleğinde depolamakta ve sırası gelen segmenti video oynatıcısında çözerek (decode ederek) oynatmaktadır. MPD dosyasında listelenen video dosyalarından hangi gösterimin indirileceğine, ağdaki kullanılabilir bant genişliğine, ağdaki gecikme durumuna, istemcinin bellek doluluk durumuna, istemcinin ihtiyaç duyduğu çözünürlüğe (Örn.: 4K TV ile cep telefonunun ihtiyaç duyduğu çözünürlükler farklıdır.) vb. koşullara bakılarak istemci üzerindeki ABR (Adaptive Bit Rate) algoritmasının çalıştırılmasıyla karar verilmektedir. Bu adaptasyon işlemi, aynı ağdaki farklı video oynatıcıları arasında herhangi bir koordinasyon olmadan yapılmaktadır (MPEG Streaming (2010); Bentaleb

et al. (2018)). Bu nedenle ve her bir istemci bant genişliğinden en yüksek düzeyde faydalanmak istediği için, adil olmayan bir kullanıcı deneyimi sorunu ortaya çıkmaktadır. Çünkü büyük bir televizyon ekranı ile cep telefonunun çözünürlük ihtiyacı aynı değildir ama TCP algoritması her ikisine de eşit davranmaktadır. Buradaki problem bununla da sınırlı değildir. TCP ile ilgili bu sorun olmasa bile veya istemcilerin hepsi aynı ekran çözünürlüğüne sahip olsa bile yine de bir adaletsizlik yaşanacaktır. Çünkü istemcilerin hepsi farklı zamanlarda ağa dahil olmakta ve tampon bellek dolulukları farklı olabilmektedir veya ağ bağlantı noktaları farklı olduğu için farklı akış yollarından farklı ağ kapasiteleri kullanabilmektedirler. Ayrıca ağ bilgisinden yoksun istemciler ağ kapasitesinden yeterince faydalanamayacaklardır. Bu problem, SDN'nin sağladığı merkezi kontrol yardımıyla giderilebilmektedir.

SDN'de kontrol düzlemi ve veri düzlemi bir birinden ayırarak ağ yönetimi merkezi bir denetleyici üzerinden gerçekleştirilmektedir. Bu mimari, ağ kaynaklarının dinamik ve programlanabilir bir şekilde yönetilmesine olanak tanımaktadır. SDN, ağ trafiği optimizasyonu, daha iyi bant genişliği tahsisi ve ağ direncini artırma gibi avantajlar sağlamaktadır. Bulut bilişim ve veri merkezleri gibi modern ağlarda yaygın olarak kullanılmaktadır (Kreutz et al. (2014)).

Tez çalışması kapsamında, SDN ortamda bulunan HAS istemcilerin QoE değerlerinin, adil bir yaklaşımla optimal bir şekilde artırılması için, ağ kaynakları ile ilgili kısıtlamaları göz önünde bulunduran optimizasyon tabanlı bir yaklaşım geliştirilmiştir. Geliştirilen bu yaklaşım, 3. parti bir uygulama olarak SDN denetleyicisi ile iletişim halinde olmak üzere, uç ağda ya da bulutta çalıştırılabilmektedir. Bu optimizasyon çalışmasının yapılabilmesi için girdi olarak; istemcilerin talep ettikleri video parçaları (segment), talep edilen video parçalarının gösterimlerinin büyüklükleri, istemcilerin bellek durumu, ağdaki kullanılabilir bant genişliği ve ağ topolojisine ihtiyaç duyulmaktadır. İlk üç girdi medya sunuculardan ve istemcilerden alınabilmektedir. Ağ topolojisinin ve ağdaki bant genişliğinin gerçek zamanlı öğrenilmesi için ise Yazılım Tanımlı Ağ ortamındaki denetleyici kullanılmaktadır. Medya sunucu, istemci ve SDN

denetleyiciden veriler toplanmakta ve birbirine bağımlı olarak çalışan dört farklı optimizasyon yöntemine verilmektedir. Bu optimizasyon yöntemleri kullanılarak uyarlamalı video dağıtımında kullanıcı deneyim kalitesi (QoE) arttırılmaktadır. Burada değerlendirilen QoE metrikleri; başlangıç gecikmesi, duraklama süresi, video kalitesi, kalite değişimi (segmentler arasında kalite değişimi olup olmadığı), ve kalite değişimindeki derinliktir. Kalite değişim derinliği, ardışık segmentler arasında kalite değişiminin ne kadar olduğudur. Örneğin; segment-1'in video kalitesi 1 olsun ve segment-2'nin video kalitesi 4 olsun. Buradaki kalite değişim derinliği $|1 - 4| = 3$ 'tür.

Optimizasyon modelleme yöntemleri olarak Doğrusal Programlama, Kar-Tamsayılı Doğrusal Programlama ve Tam sayılı (İkili) Programlama yöntemleri kullanılmaktadır. İstemciler arasındaki adaletin sağlanabilmesi için belirli bir süre istemcilerden istekler toplanmakta ardından optimizasyon işlemi yapılmaktadır. Optimizasyon sonucu belirlenen akış yolları ağ denetleyicisine gönderilerek yazılım tanımlı ağlarda kullanılan OpenFlow anahtarlama cihazlarının ayarlanması sağlanmaktadır. Bunu takiben istemcilere hangi video kalitesini hangi sunucudan isteyecekleri bilgisi gönderilmektedir. İstemcilerden veri toplanmasında ve istemcilere veri gönderilmesinde CMCD/CMSD (Consumer Technology Association (2020), Consumer Technology Association (2022)) standartlarına paralel bir yol izlenmiştir.

Tez çalışması ile yapılan katkılar Bölüm 3'te "Literatür Araştırması" başlığı altında verilmiştir.

Aynı ağ ortamındaki bütün istemcileri kapsayan merkezi bir optimizasyon işlemi sonucunda tüm istemcilerin QoE değeri benzer seviyede iyileştirilmektedir. Optimizasyon modeli ve geliştirilen mimarinin ayrıntılarının verildiği tezin ilerleyen bölümleri şu şekildedir: Tez kapsamında kullanılan teknoloji ve yöntemler Bölüm 2'de, literatürde bulunan konuyla ilgili çalışmalar ise Bölüm 3'te verilmiştir. Optimizasyon modellemeleri Bölüm 4'te yapılmış ve detaylı bir şekilde anlatılmıştır. Bölüm 5'te performans değerlendirmesi yapılmış ve Sonuç Bölümü (Bölüm 6) ile tez sonlandırılmıştır.

2 TEKNOLOJİ VE YÖNTEMLER

Bu bölümde, tezin okunabilirliğini kolaylaştırmak amacıyla gerekli olan temel kavramlar, teknoloji ve yöntemler verilmektedir. Günümüzde, Güvenlik Duvarı (Firewall - FW) ve Ağ Adresi Çevirisinden (Network Address Translation - NAT) rahatça geçebilmesi, güvenli veri transferi sunması (Transmission Control Protocol - TCP), gibi nedenlerle video trafiği için HTTP Uyarlamalı Medya Dağıtımı (HTTP Adaptive Streaming - HAS) kullanılmaktadır. Çalışmamızda Motion Picture Experts Group (MPEG) tarafından standart hale getirilen DASH (Dynamic Adaptive Streaming over HTTP) (Sodagar (2011)) tekniği kullanılmaktadır. Video kodlama yöntemi olarak da Scalable Video Coding (SVC) kullanılmaktadır. Geliştirme ve test ortamı olarak Yazılım Tanımlı Ağ (Software Defined Network – SDN) teknolojisi kullanılmaktadır. Bu mimaride ağın gözlem, kontrol ve ayarları için SDN denetleyicisi olarak Open Network Operating System (ONOS) (ON.Lab (2024)), ağ ortamının simüle edilmesi için de MiniNet (Mininet Team (2024)) kullanılmaktadır. Optimizasyon aracı olarak IBM ILOG CPLEX Optimizer (IBM (2024)), optimizasyon programlama modeli olarak Doğrusal Programlama (Linear Programming - LP), İkili Programlama (Binary Programming - BP) ve Karma Tamsayı Doğrusal Programlama (Mixed Integer Linear Programming - MILP) kullanılmaktadır. Optimizasyon yöntemi olarak Bender'in Ayrıştırma Yöntemi (Naoum-Sawaya and Elhedhli (2010)) ve Γ Sağlam Optimizasyon Yönteminden (Bertsimas and Sim (2003)) faydalanılmaktadır.

2.1 Ölçeklenebilir Video Kodlama (Scalable Video Coding - SVC)

H.264/AVC standardının bir uzantısı olan SVC, bir temel katman (base layer) ve bir veya daha fazla genişletme katmanından (enhancement layer) oluşan bir video kodlama yöntemidir. Temel katman kullanıcıya minimum seviyede video kalitesi sunmayı garanti etmektedir. Video kalitesinin iyileştirilmesi için genişletme katmanlarının da izlenen videoya eklenmesi gerekmektedir

(Schwarz et al. (2007)). SVC ile kodlanmış bir video dosyası içerisinde bulunan birden fazla katman ile kalite alternatifleri oluşturmaktadır. DASH istemcileri SVC ile kodlanmış bir video dosyasının farklı kalitelerini oluşturan katmanları talep ederek kalite adaptasyonu yapabilmektedir. AVC (Wiegand et al. (2003)) ile kodlanmış videolarda ise her bir kalite seviyesi için ayrı video dosyası oluşturulmaktadır.

2.2 Yazılım Tanımlı Ağ (Software Defined Network – SDN) ve ONOS

Yüksek kaliteli multimedia içerik dağıtımına sürekli olarak artan talep, HTTP Uyarlamalı Medya Dağıtım (HTTP Adaptive Streaming - HAS) tekniklerinin yaygın bir şekilde benimsenmesine yol açmıştır. HAS, video içeriğinin internet üzerinden verimli bir şekilde dağıtılmasını sağlayarak, ağ koşullarındaki ve cihaz kapasitesindeki değişimlere uyum sağlayabilmektedir. Bu uyum yeteneği, özellikle ağ tıkanıklığının duraksamalara veya video kalitesinin düşmesine neden olabileceği yoğun kullanım zamanlarında kullanıcılar için tatmin edici bir deneyim kalitesi (QoE) sağlaması açısından kritik öneme sahiptir. HAS için gerçek zamanlı ağ desteği sağlayabilen Yazılım Tanımlı Ağ (Software Defined Network - SDN) teknolojisi ile kullanıcı deneyim kalitesi katlanarak arttırılabilmektedir. SDN ağ kaynakları ve trafik akışları üzerinde esnek kontrol sağlayan merkezi ve programlanabilir bir ağ mimarisi sunmaktadır. Bu yaklaşım, video dağıtım performansını iyileştirmenin yanı sıra bant genişliği kullanımını optimize etmekte, böylece daha iyi kaynak kullanımı ve maliyet tasarrufu sağlamaktadır (Kalan and Dulger (2024), Bentaleb et al. (2018)).

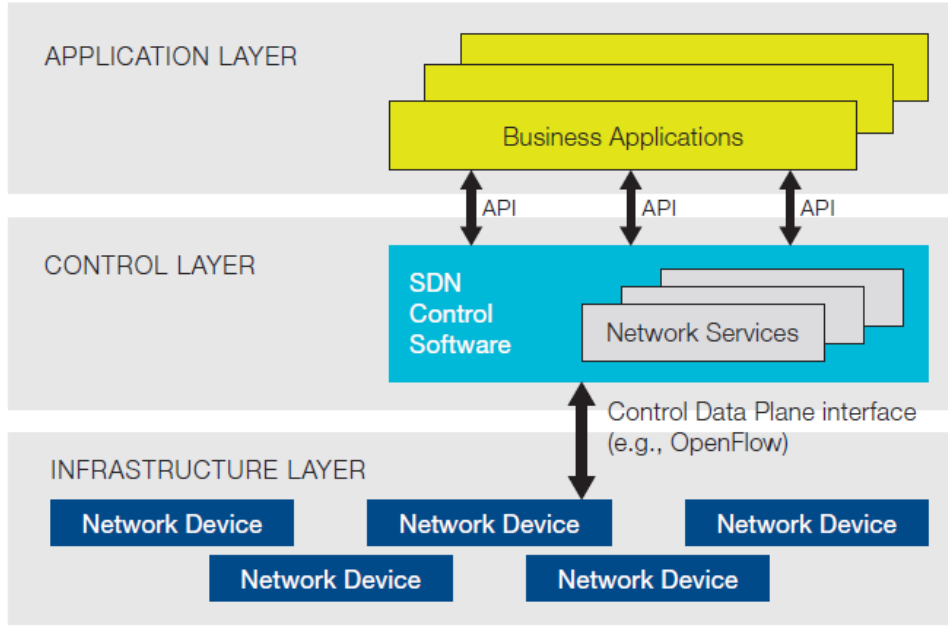
Geleneksel ağ mimarileri, günümüz işletmelerinin, ağ operatörlerinin ve son kullanıcıların ihtiyaçlarını karşılamak için yetersiz kalmaktadır. Geleneksel ağ mimarisinin yönetiminin zor olması, hata eğilimli olması, merkezi ve otonom yönetimden mahrum olması gibi nedenlerle, gelişen, değişen ve artan ağ servisi ihtiyacını karşılayamayacağı anlaşılmıştır. Bu yetersizliğe neden olan geleneksel

ağ mimarisinin sorunlardan bazıları aşağıdaki gibidir:

- Karmaşık yapısının sanal mimarinin dinamik yapısını yeterince destekleyememesi,
- Uygulanan ilke (policy) konfigürasyonlarının mimarideki karmaşıklık ve cihaz seviyesi manuel işlemlerden dolayı tutarsızlığa sebep olması,
- Dinamik bir ölçeklenebilirliğe sahip olmaması,
- Firma bağımlı olması.

Bu sorunların giderilmesi için ve bulut bilişim trafiği, cihazdan cihaza (Machine-to-Machine - M2M) ağ trafiği, HD/UHD video trafiği vb. nedenlerle artan ağ servisi ihtiyacının karşılanabilmesi için ağ mimarisinin, yazılım tanımlı (software defined) olarak yeniden dizayn edilmesi ihtiyacı doğmuştur. Yeni oluşan ihtiyacın nedeni, ağ trafik deseninin istemci-sunucu yapısından uzaklaşıp cihazlar, veri tabanları vb. ağ bileşenleri arasında yatay bir şekil almasıdır. Yazılım Tanımlı Ağlar geleceğin İnternet'i için en umut verici çözümlerden biri olarak görülmektedir ve 5G mimarisinin önemli bir bileşenidir. SDN mimarisinde kontrol (control) ve veri düzlemi (data plane) birbirinden ayrılmış, ağın yapısı ve durumuna ait bilgiler mantıksal olarak merkezileştirilmiş ve ağ altyapısı uygulamalardan soyutlanmıştır. Bu sayede programlanabilirlik, otomasyon, ağ yönetimi ve kontrol kolaylığı kazandırılmıştır (Open Networking Foundation (2011)).

Şekil 2.1'de SDN'in mantıksal mimarisi görülmektedir. SDN mimarisinde ağa ait bilgiler merkezi olarak kontrol katmanında (control layer) toplanmakta ve ağın tamamı bu merkezden global olarak görülebilmektedir. Kontrol katmanında toplanan bu veriler API'ler vasıtasıyla uygulama katmanına iletelebilmekte ve uygulama katmanından alınan komutlar da yine API'ler vasıtasıyla kontrol katmanına, oradan da ağ cihazlarına iletilmektedir. Sonuç olarak ağın tamamı, uygulamalar ve ilke motorları (policy engine) tarafından sadece anahtarlama cihazları olarak görülmektedir. Bunların sonucu olarak da SDN mimarisi sayesinde ağ yönetimi, firma bağımsız, tüm ağı tek bir noktadan kontrol edebilen, yönetim ve dizaynı kolay bir yapıya kavuşmaktadır.



Şekil 2.1: SDN Mimarisi (Open Networking Foundation (2011))

SDN ile ağ yönetim ve konfigürasyonu kolaylıkla programsal olarak yapılabilmektedir. Böylelikle çok sayıdaki konfigürasyon komutlarının manuel olarak tek tek girilmesi önlenmekte, dolayısıyla hatalar ve iş yükü azaltılmaktadır. Kontrol katmanının merkezileştirilmesiyle birlikte;

- Ağ davranışları gerçek zamanlı olarak değiştirilebilmekte ve yeni ağ uygulama ve servislerin kullanıma alınma süresi saatler ve günler seviyesine indirilebilmektedir.
- SDN ile esnek ve merkezi bir ağ yönetimi sağlanmaktadır. Ağ kaynaklarının yönetim, konfigürasyon, güvenlik ve optimizasyonu dinamik otonom programlar tarafından yapılabilmektedir. Dahası bu programlar ağ yöneticileri/geliştiriciler tarafından da kodlanarak kullanıma sunulabilmektedir.

SDN mimarisi, ağın soyutlanmasının yanında, akış, çoklu yayın (multicast), güvenlik, erişim kontrolü, bant genişliği yönetimi, trafik mühendisliği, servis kalitesi, işlem ve depolama optimizasyonu, güç kullanımı ve tüm ilke yönetim işlemlerinin yapılabilmesi için bir API kümesi sunmaktadır. Böylelikle, kontrol katmanı ve uygulama katmanı arasındaki iletişimi sağlayan açık kaynak

kodlu API'ler sayesinde uygulamalar, ağ konfigürasyonu ile ilgili detaylı ve zor yapıdan kurtularak soyutlanmış ağ üzerinde çalışabilmekte ve yüksek kalitede ağ servisi verilebilmektedir Open Networking Foundation (2011). API'ler vasıtasıyla alınan komutların anahtarlama cihazlarına iletilmesinde kullanılan OpenFlow protokolü kontrol katmanı ile altyapı (veri) katmanının haberleşmesi için yaygın olarak kullanılan protokollerden birisidir. OpenFlow protokolü ONOS ve MiniNet tarafından desteklendiği için çalışmamızda kullanılmıştır.

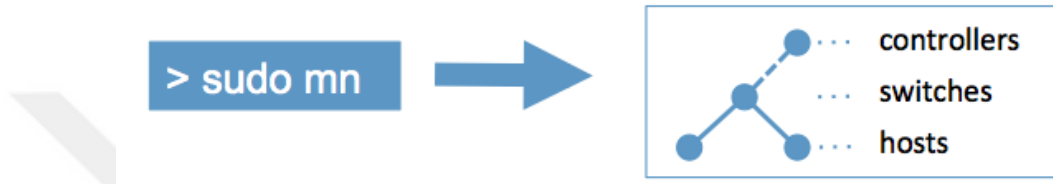
Çalışmamızda kontrol katmanı olarak kullandığımız ONOS, Open Network Operating System'in kısaltmasıdır. ONOS, yazılım tanımlı ağda (SDN) kontrol düzlemini sağlamak ve ağ bileşenlerini, örneğin anahtarlama cihazları ve bağlantılarını yönetmektedir. Ayrıca, uç sistemler ve komşu ağlar için iletişim hizmetleri sağlamak üzere yazılım programlarını veya modüllerini çalıştırmaktadır.

ONOS, API'ler, soyutlamalar, kaynak tahsisi ve izinler gibi işlevler sunmaktadır. Sunduğu API'lerden biri olan REST API çalışmamızda kullanılmıştır. HTTP kullanılarak REST API aracılığıyla uygulama katmanında (Application Layer) çalışan optimizasyon aracımız ONOS ile haberleştirilmiştir. Ayrıca ONOS'un içerisine gömülebilen uygulamalar da geliştirilebilmektedir. Komut Satırı Arayüzü (Command Line Interface - CLI) ve Grafiksel Kullanıcı Arayüzü (Graphical User Interface - GUI) ile yönetim imkanı bulunmaktadır. Geleneksel anahtarlama cihazlarının işletim sistemleri kullanılarak tek bir cihaz yönetilebilirken ONOS tek bir cihaz yerine tüm ağı yönetebilmektedir. Bu da yeni yazılım, donanım ve hizmetlerin yönetimini, yapılandırmasını ve dağıtımını büyük ölçüde basitleştirmektedir. ONOS platformu ve uygulamaları, genişletilebilir, modüler ve dağıtık bir SDN denetleyicisi olarak çalıştırılabilmektedir (ONOS Project (2024)).

2.3 MiniNet

MiniNet, Python dilinde yazılmış, kişisel bilgisayarlar üzerinde çalıştırılabilen bir ağ simülatörüdür ve çalışmamızda Altyapı Katmanı (Infrastructure

Layer) olarak kullanılmıştır. MiniNet ile komut satırı kullanılarak sanal ağlar oluşturulabilmektedir. Ayrıca Python dosyası kullanarak daha özel ve esnek topolojiler oluşturulabilmektedir. Bu topolojilerde bant genişlikleri istenilen şekilde belirlenebilmekte, bağlantılar belirli bir süre sonra kapatılabilmekte, hat kapasitesi belirli bir süre sonra arttırılabilmekte/düşürülebilmekte vb. bir çok esneklik sağlanmaktadır. MiniNet üzerinde çalıştığı işletim sistemine erişim sağlayabilmekte ve uygulama kodlarını çalıştırılabilmektedir. Sanal ağlar saniyeler içerisinde sadece bir komutla (Bkz. Şekil 2.2) oluşturulabilmektedir Mininet Team (2024).



Şekil 2.2: MiniNet Sanal Ağ Oluşturma Komutu (Mininet Team (2024))

2.4 Ortak Medya İstemci/Sunucu Verisi (Common Media Client/Server Data (CMC/SD))

Uyarlamalı medya dağıtımında deneyim kalitesini artırmanın önemli bir yolu, Ortak Medya İstemci Verisi (Common Media Client Data - CMCD) ve Ortak Medya Sunucu Verisi (Common Media Server Data - CMSD) Consumer Technology Association (2020), Consumer Technology Association (2022) standartlarının kullanılmasıdır. CMCD ile istemcisinin, medya (örneğin; segment türü/süresi/formatı, içerik/oturum kimlikleri) ve oynatma (örneğin; mevcut tampon bellek doluluğu ve duraksama) ile ilgili bilgileri medya sunucusuna nasıl ileteceği standartlaştırılmaktadır. CMSD ise sunucu tarafından istemciye bilgi iletmek için kullanılmakta ve bu bilgiler sunucu tarafı bant genişliği tahminleri, oynatma bit hızı için limitler, yönlendirme önerileri, ön bellekleme işaretleri, izleme verisi ve sunucu/ağ yük sinyalleri gibi unsurlar içerebilmektedir (Begen (2021)). Çalışmamızda, CMCD’de olduğu

gibi istemcilerin video tampon bellek bilgisini almakta ve CMSD’de olduğu gibi yönlendirme önerisini ve video kalite bilgisini istemcilere göndermekteyiz.

2.5 IBM ILOG CPLEX Optimizer

CPLEX ile doğrusal programlama, tam sayı programlama, karma tam-sayı doğrusal programlama, karesel programlama ve karesel sınırlı programlama için esnek ve yüksek performanslı matematiksel programlama çözüm araçları sağlanmaktadır. CPLEX, C, C++, Java, .NET ve Python dilleri için API desteği sunmaktadır (IBM (2024)). Çalışmamızda C++ API’si kullanılmıştır. CPLEX ile çözülmek istenen problemin modellenmesi gerekmektedir. Optimizasyon modelleri, organizasyonların ve bireylerin, belirli kısıtlamalara uyarak bir hedef fonksiyonu maksimize veya minimize ederek bilinçli kararlar almasına yardımcı olmak için tasarlanmıştır. Hedef fonksiyonlar, maksimize edilmek istenen (örneğin kâr, gelir, verimlilik) veya minimize edilmek istenen (örneğin maliyet, atık, zaman) şeyleri tanımlayan matematiksel ifadelerdir. Hedef fonksiyonu, optimizasyon probleminin temelidir. Karar değişkenleri, sonucu etkilemek için kontrol edilebilen veya ayarlanabilen değişkenlerdir, genellikle sembollerle temsil edilir ve belirli kısıtlamalara tabidir. Bu kısıtlamalar, karar değişkenleri arasındaki değerleri veya ilişkileri sınırlayan matematiksel ifadelerdir. Kısıtlamalar, kaynak mevcudiyeti, kapasite sınırları veya düzenleyici gereklilikler gibi gerçek dünya sınırlamalarını temsil eder. Oluşturulan modeller CPLEX ile çözümlenerek bir sonuca ulaşılmaktadır.

2.6 Segment Boyutunu Kullanarak Dağıtım Uyarlaması

Çoğu video akış yöntemi, geleneksel olarak dosya boyutları ve bant genişliği için MPD dosyasından elde edilen ortalama metrikleri kullanmaktadır. Ancak bu yöntem, verimsiz kaynak kullanımı ve oynatma sırasında duraksama sorunlarına yol açabilmektedir. Çalışmamızda Segment Aware Rate Adaptation (SARA) Juluri et al. (2015) tarafından da kullanılan,

video dosyalarının gerçek boyutlarının kullanılması yöntemi kullanılmıştır. Bu yöntem ile gerçek zamanlı yapılan optimizasyon çıktılarının doğruluğu artırılmıştır. Bu hassasiyet, kaynakların daha verimli tahsis edilmesini, dinamik olarak değişen video taleplerine uygun optimizasyon yapmayı ve neticesinde gerçekçi ağ kapasitesi tahsisini mümkün kılmaktadır.

Ortalama değer kullanmak yerine doğrudan dosya boyutlarını kullanmanın ne kadar fark oluşturabileceği Tablo 2.1’de örnek olarak sunulan değerler karşılaştırıldığında açık bir şekilde görülmektedir. MPD dosyasından elde edilen ”bandwidth” değeri, ilgili gösterimdeki (representation) dosyanın iletilebilmesi için saniyede ihtiyaç duyulan bitrate’i göstermektedir. Dosyada iki saniye uzunluğunda bir video bulunduğu ve çalışmamızda bir video parçasının (segment) sunucudan istemciye iletim süresinin video uzunluğuyla aynı uzunlukta olması gerektiği kabul edildiğinden, MPD dosyasından alınan bandwidth değeri iki ile çarpılmış ve ardından Megabit’e çevrilmiştir (”MPD Ortalama Büyüklüğü”). Bu örnekte, MPD’deki ortalama değer ile gerçek dosya boyutları arasındaki fark açıkça görülmektedir.

Gerçek değerın daha yüksek olduğu durumda (örneğin ED-I-1080p.seg0-L3.svc) aradaki farkın $11,20 - 7,73 = 3,47$ Megabit’e kadar çıktığı gözlemlenmektedir. Ortalama değer kullanılırsa, bu fark kullanıcı sayısının yoğun olduğu ve ağ kapasitesinin sınırlı olduğu durumlarda duraksamalara neden olacaktır. Örneğin, 30 istemciye sahip bir ağda ve 100 Mbps kapasite sınırı altında, $3,47 \times 30 = 104,1$ Mbps’lık kapasite ihtiyacının olmadığı varsayılarak optimizasyon yapılacaktır; bu da kullanıcı tarafında duraksamalara neden olacağı için çalışmanın başarısını büyük ölçüde sınırlayacaktır. Ortalama değerın daha yüksek olduğu durumda ise ağ kapasitesi yeterince verimli kullanılamayacağından dolayı video kalitesi düşecek ve yine kullanıcı deneyim kalitesi olumsuz etkilenecektir.

Çalışma esnasında yaptığımız testlerde optimizasyon işlemine girdi olarak verilen ağ kapasitesi değerinin çok az da olsa yanlış girilmesi sonucu ağ kapasitesinin yoğun olarak kullanıldığı durumlarda duraksamalara neden olduğu görülmüştür. Bu gözlemin de etkisiyle ağ kapasitesinin doğru kullanımına

Dosya Adı	Gerçek Dosya Büyüküğü (Mb)	MPD Ortalama Büyüküğü (Mb)
BBB-I-1080p.seg0-L0.svc	8.0	9.06
BBB-I-1080p.seg0-L1.svc	2.2	3.13
BBB-I-1080p.seg0-L2.svc	1.5	3.20
BBB-I-1080p.seg0-L3.svc	3.7	8.94
ED-I-1080p.seg0-L0.svc	8.00	8.04
ED-I-1080p.seg0-L1.svc	5.60	3.16
ED-I-1080p.seg0-L2.svc	3.24	3.10
ED-I-1080p.seg0-L3.svc	11.20	7.73

Tablo 2.1: Karşılaştırmalı Dosya Büyükükleri

olumlu etki yapacak olan ağdaki iletim protokollerinin getirdiği ek yük dosya büyüküğüne eklenerek optimizasyon işlemine girdi olarak verilmiştir. Şekil 2.3’de ethernet frame’in $7 + 1 + 6 + 6 + 2 = 22$ Byte başlık kısmında 4 Byte da kuyruk kısmında olmak üzere toplamda 26 Byte ekstra trafik oluşturduğu görülmektedir. Ayrıca Interpacket gap ile 12 Byte daha trafik yükü getirilmektedir. Payload kısmında ise IP paket başlığı ve TCP segment başlığı 20’şer Byte ekstra trafik yükü oluşturmaktadır (IETF (1981a), IETF (1981b)). Toplamda ise $26 + 12 + 20 + 20 = 78$ Byte ekstra trafik yükü oluşmaktadır. Her bir frame’de payload kısmının tam dolu olduğu düşünüldüğünde 1460 Byte veri gönderilebilmektedir. Bu durumda 1460 Byte veri gönderebilmek için $1460 + 78 = 1538$ Byte trafik oluşturmak gerekmektedir. Dolayısıyla $1538/1460 = 1.053$, yani gönderilecek olan dosyanın ağda oluşturacağı trafik kendi büyüküğünün binde 53 kadar fazlası olmaktadır. Ayrıca her bir dosyanın bir de HTTP başlık kısmı olacaktır. Bu hususların hepsi düşünülerek optimizasyon işleminin girdileri hassas bir şekilde belirlenmiştir.

Layer	Preamble	Start frame delimiter (SFD)	MAC destination	MAC source	802.1Q tag (optional)	Ethertype (Ethernet II) or length (IEEE 802.3)	Payload	Frame check sequence (32-bit CRC)	Interpacket gap (IPG)
Length (octets)	7	1	6	6	(4)	2	42–1500 ^[c]	4	12
Layer 2 Ethernet frame	(not part of the frame)		← 64–1522 octets →						(not part of the frame)
Layer 1 Ethernet packet & IPG	← 72–1530 octets →								← 12 octets →

Şekil 2.3: Ethernet Frame (Wikipedia Contributors (2024))

2.7 Doğrusal Programlama (Linear Programming - LP)

Doğrusal Programlama (LP), optimizasyon problemlerinde belirli bir hedef fonksiyonu maksimize etmek ya da minimize etmek için kullanılan bir matematiksel yöntemdir. LP problemleri, doğrusal bir hedef fonksiyonu ve doğrusal kısıtlamalar içerir ve geniş bir uygulama alanına sahiptir. Doğrusal hedef fonksiyon ve doğrusal kısıtlamalardan kasıt bu fonksiyonlarda kullanılan değişkenlerin üssel değerinin 1 olmasıdır. Genellikle üretim planlaması, kaynak tahsisi, ulaşım ve lojistik gibi alanlarda yaygın olarak kullanılmaktadır. Doğrusal programlama modelinin temel bileşenleri şunlardır:

- **Hedef Fonksiyonu:** Maksimize veya minimize edilmek istenen doğrusal bir fonksiyondur. Hedef fonksiyonu genellikle maliyeti minimize etmek veya kârı maksimize etmek gibi bir amaçla tanımlanır.
- **Karar Değişkenleri:** Problemi etkileyen ve optimizasyon sürecinde değerleri belirlenen değişkenlerdir. Bu değişkenler genellikle $x_1, x_2, \dots, x_n$ şeklinde gösterilir.
- **Kısıtlamalar:** Karar değişkenlerinin alabileceği değerleri sınırlayan doğrusal eşitsizlikler veya eşitliklerdir. Bu kısıtlamalar, belirli kaynakların sınırlı olması gibi sınırlamalarını temsil eder.

Örnek: Ürün A ve Ürün B üreten bir fabrikanın, sınırlı miktarda ham madde ve iş gücü kullanarak kârını maksimize etmesini hedeflenmektedir.

Ürünlerin üretimi için verilen sayısal bilgiler doğrultusunda bu problem, Doğrusal Programlama yöntemi kullanılarak modellenenmektedir.

Problem Tanımı: Fabrika, Ürün A'dan bir birim ürettiğinde 50 TL, Ürün B'den bir birim ürettiğinde ise 40 TL kâr etmektedir. Üretim sürecinde iki kısıt bulunmaktadır:

- **Ham madde kısıtı:** Ürün A'nın üretimi 3 birim, Ürün B'nin üretimi ise 2 birim ham madde gerektiriyor. Toplam ham madde miktarı 120 birimle sınırlıdır.
- **İş gücü kısıtı:** Ürün A'nın üretimi 2 saat, Ürün B'nin üretimi ise 1 saat iş gücü gerektiriyor. Toplam iş gücü saati ise 80 saattir.

Karar Değişkenleri:

- x : Ürün A'dan üretilcek miktar
- y : Ürün B'den üretilcek miktar

Hedef Fonksiyon: Toplam kârı maksimize etmek.

$$\text{Maksimize : } Z = 50x + 40y \quad (2.1)$$

Kısıtlamalar: Ham madde miktarı ve İş gücü saati sınırlıdır.

$$3x + 2y \leq 120 \quad (\text{Ham madde kısıtı}) \quad (2.2)$$

$$2x + y \leq 80 \quad (\text{İş gücü kısıtı}) \quad (2.3)$$

$$x, y \geq 0 \quad (\text{Pozitiflik kısıtı}) \quad (2.4)$$

Bu durumda, üretilcek Ürün A ve Ürün B miktarları, maksimum kârı sağlayacak şekilde belirlenir ve her iki ürünü üretmek için gereken ham madde ve iş gücü sınırlamalarına uyulması sağlanır. Doğrusal programlama bu tür kaynak sınırlamaları altında en iyi çözümü bulmak için ideal bir yöntemdir.

Çözüm: Bu problem, bir Doğrusal Programlama çözüm yöntemi kullanılarak çözülebilmektedir. Bu yöntemlerden birisi Simplex Algoritmasının kullanılması, bir diğeri IBM CPLEX Optimizer veya başka bir optimizasyon yazılımı olabilir.

Sonuçlar: Örneğin, çözüm sonucunda $x = 20$ ve $y = 30$ değerleri bulunursa:

- Fabrika, 20 birim Ürün A ve 30 birim Ürün B üreterek toplam kârını maksimize edecektir.
- Maksimum kâr:

$$Z = 50 \times 20 + 40 \times 30 = 1000 + 1200 = 2200 \text{ TL} \quad (2.5)$$

Bu örnek ile doğrusal programlamanın, sınırlı kaynaklarla en yüksek kârın nasıl sağlanacağını belirlemek için nasıl kullanılabileceği görülmektedir. Çalışmamızda, Doğrusal Programlama yöntemi, yazılım tanımlı ağda uçtan uca kullanılabilecek olan maksimum bant genişliğini bulmak için kullanılmıştır.

2.8 Tam Sayılı Programlama (Integer Programming)

Tam Sayılı Programlama (IP), optimizasyon problemlerinde karar değişkenlerinin sadece tam sayılar olmasına izin veren bir matematiksel programlama türüdür. Bu tür problemler, belirli kaynakların (örneğin, üretim hatları, araçlar veya personel) sayısal olarak belirli bir değerde tahsis edilmesi gerektiği durumlarda yaygın olarak kullanılır. IP, genellikle lojistik, üretim planlaması, tedarik zinciri yönetimi ve finansal planlama gibi alanlarda kullanılmaktadır. Tam sayılı programlama modeli, doğrusal hedef fonksiyonlarını ve doğrusal kısıtlamaları kullanarak belirli bir problemi çözmeye çalışmaktır. Ancak, karar değişkenlerinin sadece tam sayılar olması gerekmektedir. Bu durum problemi daha karmaşık hale getirmektedir. Model, bir hedef fonksiyonu (Z) ve bu hedefi optimize etmek için karar değişkenlerini (x) içermektedir. Ayrıca, kısıtlamalar ile ($Ax \leq b$) hedef fonksiyonun sınırlarını belirlenmektedir.

Örnek: Bir lojistik şirketinin minimum maliyetle depolar ve mağazalar arasında nakliye planlaması yapması hedeflenmektedir. Nakliye maliyetlerini minimize etmek için tüm depolardan belirli miktarda ürün taşınması gerekmektedir. Her bir taşıma işlemi tam sayı olarak ifade edilmiştir.

Problem Tanımı: Şirketin 2 deposu (D_1 ve D_2) ve 3 mağazası (M_1 , M_2 , M_3) bulunmaktadır. Depoların her biri belirli bir depolana kapasitesine

sahip olup, her mağazanın belirli bir talebi vardır. Aşağıda nakliye maliyetleri ve kapasite/talep değerleri verilmiştir;

Nakliye Maliyetleri (TL):

	M_1	M_2	M_3
D_1	4	6	8
D_2	5	4	3

Depo Kapasiteleri ve Mağaza Talepleri:

- D_1 kapasitesi: 70 birim
- D_2 kapasitesi: 50 birim
- M_1 talebi: 40 birim
- M_2 talebi: 50 birim
- M_3 talebi: 30 birim

Karar Değişkenleri:

- x_{ij} : D_i deposundan M_j mağazasına taşınacak ürün miktarı, tamsayı ve pozitif olmak zorundadır.

Hedef Fonksiyon: Toplam nakliye maliyetini minimize etmek istiyoruz.

$$\text{Minimize : } Z = 4x_{11} + 6x_{12} + 8x_{13} + 5x_{21} + 4x_{22} + 3x_{23} \quad (2.6)$$

Kısıtlamalar:

$$x_{11} + x_{12} + x_{13} \leq 70 \quad (\text{Depo } D_1 \text{ kapasite kısıtı}) \quad (2.7)$$

$$x_{21} + x_{22} + x_{23} \leq 50 \quad (\text{Depo } D_2 \text{ kapasite kısıtı}) \quad (2.8)$$

$$x_{11} + x_{21} = 40 \quad (\text{Mağaza } M_1 \text{ talep kısıtı}) \quad (2.9)$$

$$x_{12} + x_{22} = 50 \quad (\text{Mağaza } M_2 \text{ talep kısıtı}) \quad (2.10)$$

$$x_{13} + x_{23} = 30 \quad (\text{Mağaza } M_3 \text{ talep kısıtı}) \quad (2.11)$$

$$x_{ij} \geq 0 \quad \text{ve tamsayıdır (Pozitiflik ve tam sayı kısıtı)} \quad (2.12)$$

Çözüm: Bu tam sayılı programlama problemi, Branch-and-Bound algoritması, IBM CPLEX, Gurob veya başka bir optimizasyon yazılımı kullanılarak çözülebilmektedir.

Sonuçlar: Örneğin, çözüm sonucunda $x_{11} = 20$, $x_{12} = 30$, $x_{13} = 20$, $x_{21} = 20$, $x_{22} = 20$, ve $x_{23} = 10$ değerleri bulunursa:

- Optimum taşıma planı, minimum maliyetle depoların kapasitesine ve mağazaların taleplerine uygun şekilde belirlenmiş olacaktır.
- Toplam minimum maliyet:

$$\begin{aligned} Z &= 4 \times 20 + 6 \times 30 + 8 \times 20 + 5 \times 20 + 4 \times 20 + 3 \times 10 \\ &= 80 + 180 + 160 + 100 + 80 + 30 = 630 \text{ TL} \end{aligned} \quad (2.13)$$

Bu örnek ile tam sayılı programlamanın, belirli kapasite ve talep kısıtlamalarına uyarken maliyetleri nasıl optimize edebileceğini gösterilmiştir. Bu yöntem çalışmamızda kullandığımız tam sayılı programlamanın özel bir türü olan ikili programlamaya (binary programming) giriş amacıyla verilmiştir.

2.9 İkili Programlama (Binary Programming - BP)

İkili programlama, karar değişkenlerinin yalnızca 0 veya 1 değerleri alabildiği özel bir tam sayılı programlama türüdür. Bu tür bir modelleme, özellikle kararların iki seçenek arasında olduğu durumlarda kullanışlıdır; örneğin, bir işin yapılması ya da yapılmaması, bir sistemin açık veya kapalı olması gibi. İkili programlama, operasyon araştırmaları, lojistik, finans ve proje seçimi gibi alanlarda yaygın olarak kullanılmaktadır. İkili programlama, karar vericilere belirli bir bütçe veya kaynak sınırlaması dahilinde maksimum getiri elde etmek için en uygun seçenekleri belirleme imkanı sağlamaktadır. Örneğin, bir yatırımcının belirli projeler arasından en kârlı olanları seçmesi veya bir lojistik firmasının, sınırlı kapasite ile hangi rotaları tercih etmesi gerektiğine karar vermesi gibi durumlarda ikili programlama kullanılabilmektedir.

Bir şirketin sınırlı bütçeyle hangi projeleri üstleneceğine karar vermesi gerektiğini düşünelim. Her projenin belirli bir maliyeti ve beklenen bir getirisi var ve şirket, bütçeyi aşmadan toplam karını maksimize etmek istemektedir.

- Problem Tanımı:

$x_1, x_2, \dots, x_n$ ikili karar değişkenleri olsun. Her bir x_i değişkeni:

$$x_i = \begin{cases} 1, & \text{eğer } i \text{ projesi seçildiyse} \\ 0, & \text{eğer } i \text{ projesi seçilmediyse} \end{cases}$$

- Amaç Fonksiyonu: Toplam getiriye maksimize etmek:

$$\text{Maksimize } Z = p_1x_1 + p_2x_2 + \dots + p_nx_n$$

Burada p_i , i projesinin getirisi.

- Kısıtlar: Toplam maliyetin mevcut bütçeyi aşmamasını sağlamak için:

$$c_1x_1 + c_2x_2 + \dots + c_nx_n \leq B$$

Burada c_i , i projesinin maliyeti ve B bütçe miktarını göstermektedir.

Örnek: Bir şirketin üç projesi olduğunu varsayalım ve bu projelerin maliyetleri ve getirileri aşağıdaki gibidir:

- Proje 1: Maliyet = 10, Getiri = 20
- Proje 2: Maliyet = 15, Getiri = 25
- Proje 3: Maliyet = 20, Getiri = 30

Şirketin $B = 25$ olan bir bütçesi vardır ve bu bütçeyi aşmadan karını maksimize etmek istemektedir.

Her bir proje için x_i ikili karar değişkeni olsun:

$$x_i = \begin{cases} 1, & \text{eğer } i \text{ projesi seçildiyse} \\ 0, & \text{eğer } i \text{ projesi seçilmediyse} \end{cases}$$

Amaç Fonksiyonu:

$$\text{Maksimize et } Z = 20x_1 + 25x_2 + 30x_3$$

Bütçe Kısıtı:

$$10x_1 + 15x_2 + 20x_3 \leq 25$$

İkili Kısıtlar:

$$x_1, x_2, x_3 \in \{0, 1\}$$

Bu problemin çözümü, bütçe dahilinde karı maksimize etmek için hangi projelerin seçilmesi gerektiğini belirlemektedir. Çalışmamızda kullanılan ikili programlama yöntemi ile ağda en düşük kalitede dahi video segmenti gönderecek kadar yeterli bant genişliği olmadığı durumda, varsayılan sunucudan istemcilere, en az bant genişliği kapasitesi aşımı yaparak, hangi yollar üzerinden veri iletimi yapılacağı problemi modellenmiş ve IBM CPLEX Optimizer kullanılarak belirlenen model çözümlenmiştir.

2.10 Karma Tamsayı Doğrusal Programlama (MILP)

Karma Tamsayı Doğrusal Programlama (MILP), karar değişkenlerinin hem kesirli (sürekli) hem de tam sayı olabildiği optimizasyon problemlerini çözmek için yaygın olarak kullanılan bir matematiksel yöntemdir. MILP, çeşitli endüstrilerde, özellikle lojistik, enerji sistemleri, ağ optimizasyonu ve video akış yönetimi gibi alanlarda çok çeşitli problemlerin çözümünde kullanılmaktadır. MILP'nin temel avantajı, karmaşık ve çok boyutlu problemlerin etkili bir şekilde modellenebilmesi ve çözülmesidir. MILP modelinde de diğerler modellerde olduğu gibi, bir hedef fonksiyon, karar değişkenleri ve kısıtlamalar bulunmaktadır ve doğrusal denklemle ifade edilmektedir. MILP modelini uyarlamalı video dağıtım optimizasyonu problemini üzerinden açıklayalım. Diyelim ki, bir ağda birden fazla istemci farklı video dosyalarını izliyor ve her bir istemcilerin belirli bir ağ kapasitesi talebi var. Bu müşterilerin akışları, ağın sınırlı bant genişliğine bağlı olarak dağıtılmalıdır. Bu durumda MILP, her bir istemci için bant genişliği tahsisi yapmak için kullanılabilir.

- **Hedef Fonksiyon:** Müşterilerin video izleme deneyim kalitesini (QoE - Quality of Experience) maksimize etmek.

- **Karar Değişkenleri:** Her bir istemciye hangi bant genişliğinin tahsis edileceği, her bir akışın yönlendirileceği ağ yolunun belirlenmesi.
- **Kısıtlamalar:**
 - Ağın toplam bant genişliği kapasitesinin aşılmaması,
 - Her bir istemcinin minimum kalite gereksinimlerinin karşılanması,
 - Her istemcinin tek bir yol üzerinden yönlendirilmesi.

Bu model, MILP kullanılarak çözülür ve en uygun bant genişliği tahsisi yapılır. MILP'nin avantajı, çeşitli kısıtlamalar altında, ağdaki kaynakların en verimli şekilde kullanılmasını sağlamaktır.

2.11 Bender'in Ayrıştırma Yöntemi(Bender's Decomposition)

Çalışmamızda Bender'in Ayrıştırma Yöntemi'ne benzer bir yöntem kullanılmaktadır. Bender'in Ayrıştırma Yöntemi'nde, çözülmek istenilen optimizasyon problemi ana problem ve alt problemler olmak üzere iki optimizasyon problem grubuna ayrılmaktadır. Ana problemde bulunan, değişkenlere ait geçerli sonuçlar alt problem(ler)e girdi olarak verilmektedir. Şayet alt problemde geçerli bir çözüme ulaşılamazsa ana probleme denenen değerler için uygulanabilirlik kısıtı (feasibility cut) eklenmektedir. Böylelikle denenmiş olan bu değerlerin ve bu değerlerin çizdiği sınırın dışında kalan değerlerin bir çözüme götürmeyeceği belirlenmektedir. Başka bir deyişle, çözülmek istenilen probleme bir üst sınır getirilmekte ve bu sınır üstünde kalan değerlere bakılarak vakit kaybedilmesi önlenmektedir. Böylelikle problem alanı daraltılmaktadır. Şayet alt problemde geçerli bir çözüme ulaşırsa bu sefer de ana probleme en iyilik kısıtı (optimality cut) eklenmektedir. Böylelikle optimum sonuca ulaşmak için kullanılacak olan değerlere bir alt sınır çizilmektedir. Bu işlemler tekrarlanarak alt sınır ve üst sınır değerleri birbirine yaklaştırılmaktadır. İkisinin eşit olduğu noktada optimum değer elde edilmektedir. Büyük problemlerde işlem yoğunluğu nedeniyle işlem süresini kısaltmak için optimum değer yerine bir

ϵ ($\epsilon > \frac{UB-LB}{UB}$, $\epsilon > 0$) değerine ulaşıldığında bu optimal altı değer yeterli görülerek optimizasyon işlemi sonlandırılabilir. Geçerli en iyilik ve uygulanabilirlik kısıtlarının kullanılması, ana problemin çözümü için yapılan tekrar sayısının azaltılmasında ve toplam hesaplama süresini düşürülmesinde etkili olduğu kanıtlanmış bir yöntemdir (Naoum-Sawaya and Elhedhli (2010)). Çalışmamızda Bender'in Ayrıştırma Yöntemi'nden esinlenerek optimizasyon problemi ana problem ve alt problem (işçi problem) olarak iki optimizasyon problemine ayrılarak probleme iki farklı yaklaşım açısı geliştirilmiştir;

1. Ana problemin sonucu alt probleme girdi olarak verilmektedir. Başka bir deyişle, Ana problemde, her bir istemciye sunulacak olan video kalitesi (ilgili segmente ait katman sayısı) her hangi bir hat kapasitesi sınırına bakılmadan belirlenmektedir. Yani, döngünün ilk tekrarında tüm istemciler için en yüksek video kalitesi belirlenmektedir. Alt problemde videoların belirlenen bu kalitelerde, mevcut ağ kaynaklarını kullanılarak, istemcilere iletilip iletilemeyeceğine bakılmaktadır. Şayet alt problemde bir sonuca ulaşılamazsa ana probleme girilen uygulanabilirlik kısıtı ile mevcut video kalite seviyesi yarıya düşürülmektedir (toplam video katmanları sayısı yarıya düşürülmektedir). Bununla bir üst sınır belirlenmekte ve toplam video katman sayısının bu sınır üzerine çıkması engellenmektedir. Alt problemde geçerli bir sonuç bulunmuşsa ve optimum noktaya ulaşılmışsa işlem sonlandırılmakta, şayet optimum noktaya ulaşılamamışsa, bir önceki üst sınır silinmekte, bir önceki olumlu ve olumsuz sonuç veren kalite düzeylerinin orta noktası, yeni üst sınır olacak şekilde ana probleme eklenmektedir. Yeni alt sınır olarak da mevcut olumlu sonuç kullanılmaktadır. Özetle, her döngü tekrarında problem üst ve/veya alt sınırlarından daraltılmakta ve optimum nokta bulunmaktadır. Bu yaklaşımda kalite düzeyinin belirlenmesinde böl fethet (divide and conquer) algoritması ve Bender'in Ayrıştırma Yöntemi benzeri olan bir yöntem birlikte kullanılmaktadır.
2. Yukarıdaki ilk yaklaşımımızda optimizasyon problemimiz en kötü durumda 8 tekrarda çözülebilmektedir. Bu yöntemle her ne kadar kabul

edilebilir bir sürede optimizasyon problemi çözülsede birden fazla sunucu olan ortamlarda kullanılamamakta, merkezi olarak istemcilerin video tampon belleğinde ne kadar arttırım yapılabileceği veya kaynakların kısıtlı olduğu durumlarda bu tampon belleklerden ne kadarının feda edilebileceği belirlenememektedir. Bu nedenlerle ikinci bir yaklaşım geliştirilmiştir. Bu yaklaşımımızda öncelikli olarak uçtan uca hat kapasitesi doğrusal bir optimizasyon modeliyle belirlenmekte ve bu bilgi kullanılarak ana problemde belirlenen video kalitelerinin üst sınırı rahatlıkla tespit edilebilmektedir. Uçtan uca kullanılabilir ağ kapasitesi verisiyle birden fazla sunucu bulunan ortamda video dağıtımı optimale yakın bir şekilde gerçekleştirilebilmektedir. Bu şekilde hat kapasitesi açısından belirlenen üst sınırla problemin çözümü genel olarak bir döngü tekrarıyla çözülebilmektedir. Bu yaklaşımımız tez çalışmamızın temelini oluşturmaktadır ve ilerleyen aşamalarda detaylı bir şekilde anlatılmaktadır.

2.12 Γ Sağlam (Γ Robust) Optimizasyon Yöntemi

Γ sağlam optimizasyon yöntemi, işleme alınan veride belirsizlik ve olası sapmalar olduğu durumlarda kullanılan bir yöntemdir (Bertsimas and Sim (2003)). Çalışmamızda bu yöntemin bir benzeri kullanılmıştır. Γ sağlam optimizasyon yöntemine ait tanımlama şu şekildedir;

Γ sağlam optimizasyon yöntemiyle katsayısı önceden bilinmeyen veya değişkenlik gösteren kısıtların bulunduğu optimizasyon problemleri çözülmek istenmektedir. Örneğin; bir A matrisi içerisinde bilinen a_{ij} girdileri ve bu bilinen girdilerin değişim gösterebileceği bir $[a_{ij}-\hat{a}_{ij}, a_{ij} + \hat{a}_{ij}]$ aralığı olduğunu varsayalım. Bu durumda Γ sağlam optimizasyon yönteminde,

$$\sum_j a_{ij}x_j \leq b_i \quad (2.14)$$

kısıtı,

$$\sum_j a_{ij}x_j \pm \max_{j \in S_i} \{\sum_{j \in S_i} \hat{a}_{ij}x_j\} \leq b_i, \quad S_i \subseteq J_i, |S_i| \leq \Gamma \quad (2.15)$$

olarak değiştirilmektedir. Burada J_i sapma olabilecek kolonları belirtmektedir. Γ ise bu sapma olabilecek olan kolonlardan hesaba katılacak olan maksimum kolon sayısını vermektedir. Böylelikle Γ kadar data hattında önceden herhangi bir rezervasyon yapmadan $\pm \hat{a}$ miktarına kadar olan değişikliklerden etkilenebilen bir model ortaya konulmaktadır. Çalışmamızda buradaki yöntemle benzer şekilde ancak Γ ile bir sınırlama getirilmeden tüm kullanılabilir hatlar işleme alınarak bir yaklaşım geliştirilmiştir. Sağlam optimizasyon yönteminde $\hat{a}_{ij}$ değişkenine sıfırdan yüksek bir alt sınır belirlenirse kullanılabilir hat kapasitesi o kadar azalmaktadır. Bu durumda ihtiyaç duyulan hat kapasitesi karşılanamayabilmektedir. Bu nedenle çalışmamızda bu alt sınır değeri sıfır olarak belirlenerek ilgili değişken hedef fonksiyonda üst sınırına kadar maksimize edilmeye çalışılmıştır. Böylelikle imkan dahilinde kaliteden taviz verilmeden sağlamlık sağlanmıştır. İmkan yoksa da kaliteden taviz verilmemiştir. Başka bir deyişle, mevcut hat kapasitesi talep edilen hat kapasitesini karşılamıyorsa boşa çıkarılabilecek bir hat kapasitesi olmadığından ilgili değişken değeri sıfır yapılmaktadır.

3 LİTERATÜR ARAŞTIRMASI

HTTP üzerinden uyarlamalı medya dağıtımı için günümüzde temel olarak tamamen istemci tabanlı adaptasyon yöntemi, ağ elemanı destekli (Örn.: vekil (proxy) sunucu) ve ağ tabanlı (SDN) adaptasyon yöntemi karşımıza çıkmaktadır. Tamamen istemci tabanlı yöntemde, bir istemci, kullanılabilir ağ kapasitesi tahminlemesi, tampon bellek doluluğu gibi yerel parametrelere dayanarak medya kalite adaptasyonu yapmaktadır. Bu yöntemle ilgili literatürde çok fazla çalışma bulunmaktadır ancak bu yöntemde istemcilerin yeterli ağ bilgisi olmadığı için video kalite uyarlaması istenilen seviyede yapılamamaktadır (Bentaleb et al. (2018)). Özellikle ağda yetersiz bant genişliği kaynaklı bir dar boğaz varsa istemcilerin performansının çok daha fazla düştüğü gözlemlenmektedir (Akhshabi et al. (2012); Huang et al. (2012)). Bahse konu bu zorlukların aşılması ve kullanıcı deneyim kalitelerinin (QoE)

yükseltilmesi için Ağ Destekli Uyarlamalı Video Dağıtım Yöntemi karşımıza umut verici bir çözüm yöntemi olarak çıkmaktadır. Bu yöntemde, tüm ağ geniş bir açıdan görülebilmektedir. Böylelikle video oynatıcılarının doğru adaptasyon kararları almasına yardımcı olunabilmektedir. Dolayısıyla kullanıcıların deneyim kaliteleri iyileştirilebilmektedir. Bununla birlikte gelişen, uç cihaz (edge computing) yöntemi ve ağ cihazlarının sanallaştırılması (Network Function Virtualization - NFV) ile uçtan uca ağ desteği daha iyi sağlanabilmektedir (Kalan and Dulger (2024); Bentaleb et al. (2018)). Bununla birlikte, Uyarlamalı medya dağıtımında deneyim kalitesini artırmanın önemli bir yolu da, CMCD/CMSD Consumer Technology Association (2020), Consumer Technology Association (2022) standartlarının kullanılarak istemci ve sunucu iletişiminin sağlanmasıdır. Bu kapsamda modern ağ destek teknoloji ve yöntemleri kullanarak uyarlamalı video dağıtımını yapan çalışmalar; *Ağ Destekli Uyarlamalı Video Dağıtım Yöntemi*, *Ağ Destekli Uyarlamalı Katmanlı Video Dağıtım Yöntemi ve Genel Meya İstemci/Sunucu Verisi (Common Media Client/Server Data - CMC/SD) Uyumlu Çalışmalar* başlıkları altında incelenmiştir. Çalışmamızı literatürdeki bu çalışmalardan ayıran özellikler, dolayısıyla yaptığımız katkılar şu şekildedir;

1. Uçtan uca kullanılabilir ağ kapasitesinin tespit edilmesi sayesinde optimizasyon probleminin daraltılmasıyla, Bender'in Ayrıştırma Yöntemi ile Γ Sağlam Optimizasyon Yönteminden faydalanılarak ve geliştirdiğimiz algoritmayla yapılan gevşetmeyle, NP-Tam olarak tanımlanan optimizasyon probleminin optimum seviyeye yakın ve hızlı bir şekilde çözümünün sağlanması,
2. Yine kullanılabilir ağ kapasitesi verisi ve istemcilerden CMCD'ye benzer şekilde alınan tampon bellek doluluk verisiyle tüm istemcilerin tampon bellek yönetimi organize bir şekilde yapılabilir. Böylelikle, hem hangi istemcinin/istemcilerin tampon bellek arttırımı yapabilecekleri belirlenmekte hem de hat kapasitesi yetersiz olduğu durumlarda hangi istemcinin/istemcilerin tampon belleğinin feda edileceğinin (hangi istemcinin ilgili zaman diliminde bekletileceği) belirlenebilmesi,

3. Ağ kapasitesinin etkin kullanımı için kullandığımız yöntem ile hem video kalitesinin önemli ölçüde artırılması hem de video kalite değişiminin azalması,
4. Akış atamalarının etkin bir şekilde yapılması ile OF cihazlardaki akış ayarlarının beklenmesinde geçen sürenin video dağıtımında kullanılabilmesi,
5. Γ Sağlam Optimizasyon Yöntemi sayesinde kaliteden ödün vermeden ve QoS uygulamadan, esnek bir şekilde ağdaki dalgalanmalardan etkilenmeyi azaltan bir yöntem geliştirilmesi,
6. Ağ kapasitesinin temel kalite seviyesini göndermeye dahi yeterli olmadığı durumlarda kaçınılmaz olan duraksamaları en aza indiren bir optimizasyon modeli kullanılması,
7. Duraksama yaşayan veya duraksama yaşama ihtimali olan istemcilerin önceliklendirilmesi.

3.1 Ağ Destekli Uyarlamalı Video Dağıtım Yöntemi

Kleinrouweler ve diğerleri, bir QoS algoritması sunmaktadır. Bu algoritma ile OpenFlow (OF) anahtarlama cihazlarında, HAS trafiği ve diğer ağ trafikleri için iki ayrı kuyruk oluşturulmaktadır. Bu QoS mekanizması, her bir istemciye eşit bant genişliği tahsisi yapmaktadır (Kleinrouweler et al. (2016)). Ancak bu durum kullanıcı deneyimini olumsuz etkileyecektir. Çünkü her istemci farklı çözünürlük, dolayısıyla farklı ağ kapasitesine ihtiyaç duyabilmektedir. Sadece QoS uygulanarak çok sınırlı bir kullanıcı deneyim kalitesi sağlanabilir.

Thomas ve diğerleri, Sunucu ve Ağ Destekli DASH (Server And Network-assisted DASH - SAND) ile ağ üzerinden istemci ve sunucu arasında bilgi alışverişi yaparak akış performansını iyileştirmektedir. Bu, ev kullanıcıları için, tampon bellek yönetimi sorunları, CDN seçimi ve mobil kullanıcılar için kenar önbellekleme gibi konuları ele almaktadır. SAND, hem standart DASH hem de SAND destekli öğelerin (DASH Farkında Ağ Öğeleri, (DASH Aware Network Elements - DANEs)) birlikte çalışmasına olanak tanımaktadır. SAND,

tanımlı bir mesaj seti sunarken, uygulaması açık bırakılmıştır (Thomas et al. (2016)). Bu da kaynak optimizasyonu için daha entegre ve programlanabilir bir yaklaşım sunan SDN'e kıyasla etkisini sınırlamaktadır.

Bentaleb ve diğerleri, QoS-QoE eşlemesine ile ağ kaynaklarını dinamik bir şekilde yöneterek QoE'yi optimize eden, SDN destekli bir HTTP video akış çerçevesi olan SDNDASH'ı sunmuşlardır. İstemci tarafındaki tampon bellek durumu ve duraksamalar gibi bilgiler DASH.js uzantısı aracılığıyla toplanırken, ağ kaynakları ile ilgili durum SDN denetleyicisinden alınmaktadır. Optimizasyon sonuçları ile istemcilere video kalite önerileri sunulurken, SDN denetleyici içerisindeki bir SDN uygulaması ile ağ kaynakları yapılandırılmaktadır (Bentaleb et al. (2016)). Yine Bentaleb ve diğerleri büyük ölçekli video dağıtımlarını desteklemeyi amaçlayan, SDNDASH'ın gelişmiş bir versiyonu olan SDNHAS'ı tanıtmışlardır. Bu geliştirme, HAS istemcileri ile SDN arasındaki iletişim yükünü azaltırken, önceki versiyonda ele alınmayan kısa ve uzun vadeli ağ dalgalanmaları arasında ağ kaynaklarını etkin bir şekilde tahsis etmektedir (Bentaleb et al. (2017)).

Lu ve diğerleri, ağ istatistiklerini elde etmek için SDN teknolojisini kullanmaktadırlar. Merkezi bir koordinatör, istemcilerden kalite ve video tampon bellek bilgilerini toplamakta ve belirlediği video kalite seviyesini istemcilere göndermektedir. Ancak, OF cihazlar üzerinde herhangi bir akış ayarlaması yapmamaktadır (Lu et al. (2017)). Bu da ağ kaynaklarının yetersiz şekilde kullanılması sonucunu doğurmaktadır.

Bhat ve diğerleri SABR ile, SDN teknolojisini kullanarak, CDN'lerde HAS kullanıcı deneyim kalitesini arttırmaktadırlar. Bunu, ağdaki önbellek içeriği ve mevcut bant genişliği bilgisini istemcilere sağlayarak ve video dağıtımları için ağ trafiğini dinamik olarak programlayarak yapmaktadır (Bhat et al. (2018)).

Erfanian ve diğerleri, Gelişmiş Video Kodlama (AVC) yöntemi için kullanıcı deneyimini artırmak amacıyla, zaman bölmeli, LP-gevşetilmiş bir optimizasyon modeli sunmaktadırlar. Ancak çalışmada sadece NP-Tam model verilmekte, LP-gevşetilmiş model verilmemektedir. Her ne kadar LP-

gevşetilmiş model verilme de ortak yazarlardan ve çalışmalar arasındaki farkın neredeyse sadece farklı video kodlamaları kullanmasından başka bir şey olmamasından dolayı bu çalışmanın Tashtarian et al. (2018) ile aynı olumsuzlukları barındırdığı açıktır. Çalışmadaki optimizasyon işlemi ile her bir istemci için video dosyasının ne kadarının hangi yoldan gideceği ve video kalitesinin ne olacağı belirlenmektedir. Başka bir algoritmayla ilgili dosyanın hangi TCP paketinin nereden gönderileceği paket üzerinde etiketleme yapılarak belirlenmektedir. Bu etiketlemelerle ilgili atamalar OF cihazlar üzerinde yapılmaktadır (Erfanian et al. (2018)).

Özçelik ve Ersoy, ağ kaynaklarını rezerve etmeden, SDN destekli bir ortamda DASH istemcileri için merkezi optimizasyonu mümkün kılmak amacıyla video parçalarının (segment) oynatma sürelerini dikkate alan CSASDN'i geliştirmişlerdir. Bu çalışma ağdaki darboğaz bağlantılarına odaklanan bir çalışmadır (Ozcelik & Ersoy (2019)).

Kalan, SDN ve SAND teknolojilerini kullanarak, istemci kalite kararlarına bir üst sınır koymakta ve en yakın CDN (veya önbellek) seçimi yoluyla ağ yükünü azaltıp kullanıcı deneyim kalitesini adil bir şekilde iyileştirmektedir. İstemcilerin video tampon bellekleri belirli bir seviyenin altındayken yalnızca en düşük katmanı alabilmektedirler (Kalan (2021)). Bu da adil kalite deneyimi iyileştirmesini olumsuz etkileyen bir yöntemdir.

Farahani ve diğerleri SDN ve uç bilişimi (edge computing) kullanan, bir segmenti en kısa getirme süresine göre bir önbellek sunucusundan ya da kaynak sunucudan seçmek için zaman dilimli bir optimizasyon modeli çalıştıran ES-HAS adlı bir çalışma sunmuşlardır (Farahani et al. (2021)). Bu çalışma, daha sonra CSDN ile geliştirilerek hibrit getirme/dönüştürme mekanizmasıyla istemcilere hizmet vermek için bir codec dönüştürücü işlevi eklenmiştir (Farahani et al. (2021)). Ardından, LEADER ile, HAS istemcilere hizmet vermek için uç sunucular (edge server) arasında iş birliği kurarak bu sunucuların boşta kalan kaynaklarını değerlendiren bir sistem tanıtmışlardır (Farahani et al. (2022)). Son olarak ARARAT ile, CSDN'i daha da geliştirerek SDN, NFV ve uç bilişimi içeren hiyerarşik bir mimari sunmuşlardır. Bu mimari

ile, ağdaki iletim maliyetini göz önüne alarak hizmet süresini minimuma düşürmeyi hedeflemektedirler. Böylelikle istemci cihazlardaki indirme süresine dayanarak çalışan Uyarlanabilir Bit Hızı (ABR) algoritmalarının daha iyi tahminleme yapmasını sağlamaktadırlar (Farahani et al. (2023)).

Farahani ve diğerleri, SDN, Hizmet Fonksiyon Zincirleri (Service Function Chains - SFC) ve uç bilişimin iş birliğini kullanarak gecikme ve/veya bant genişliği duyarlı olanlar (örneğin, canlı yayın ve isteğe bağlı video (Video on Demand - VoD, Örn.: Youtube)) gibi çeşitli uyarlanabilir video akış hizmetleri için SERENA adlı bir SFC özellikli mimariyi tanıtmışlardır. Başka bir deyişle modern ağ teknolojilerinden faydalanarak farklı uyarlamalı video tiplerinin dağıtımının etkin bir şekilde yapılmasını sağlayan bir çalışma tanıtmışlardır (Farahani et al. (2023)).

3.2 Ağ Destekli Uyarlamalı Katmanlı Video Dağıtım Yöntemi

Tashtarian ve diğerleri, Ölçeklenebilir Video Kodlama (Scalable Video Coding - SVC) ile kodlanmış videolar için kullanıcı deneyimini arttırmak amacıyla Zaman Bölmeli LP-gevşetilmiş bir optimizasyon modeli kullanan, SDN destekli bir HTTP video akış çerçevesi olan S^2VC 'yi sunmuşlardır. Yapılan optimizasyonla her bir istemci için ortalama video katman dosya büyüklüğüne dayanarak ondalık bir şekilde video kalitesi belirleniyor. Bu ondalık değer aşağı yuvarlanarak nihai video kalitesi tespit ediliyor. Ayrıca yine ortalama video kalitesine dayanarak bu istemcinin hangi yolları kullanacağı tespit ediliyor. Ancak her bir istemci için belirlenen bu yolları kullanmak yerine, video iletiminin hangi yollardan yapılacağı farklı bir algoritmayla belirlenmektedir. Çalışmada her bir video katmanının ne kadarının hangi yoldan gideceği ve bunun sonucunda bu katman için hangi yolların kullanılacağı başka bir algoritmayla belirlenmektedir. İletimin sağlanması için de her bir TCP paketi medya sunucudan çıkmadan önce etiketlenmekte ve bu etiketlemelerle ilgili atamalar OF cihazlar üzerinde yapılmaktadır. Burada üç sorun

görülmektedir; 1. Çok kısa sürelerde yapılan optimizasyon işleminde ortalama dosya büyüklüğü duraksamalara veya düşük video kalitesine sebep olabilir. 2. Her bir istemci için tespit edilen aşağı yuvarlama miktarı kapasite kullanımını, dolayısıyla video kalitesini düşürecektir. 3. Her bir TCP paketinin etiketlenmesi zaman yönetimini negatif etkileyecektir. Bu çalışmayla bir sunucu veya bir sunucu kümelemesi için çözüm bulunmakta ancak birden fazla sunucu (veya kümelemesi) desteği sunulamamaktadır. Çalışma ile optimal altı bir kullanıcı deneyim kalitesi sağlanabilmektedir (Tashtarian et al. (2018)).

Gohar ve diğerleri, mevcut ağ bant genişliğini maksimize etmek için SDN teknolojisi ve Çoklu Yönlü İletim Kontrol Protokolünü (Multipath Transmission Control Protocol - MPTCP) kullanarak Dinamik Çoklu Yol Bulucu (Dynamic Multi-Path Finder - DMPF) algoritmasını sunmuşlardır (Gohar et al. (2019)). Bu yaklaşım, Gohar and Lee (2020) ile daha da geliştirilmiştir. Ancak, yalnızca yol seçimine dayanan bu yöntem, QoE’de umut verici fakat sınırlı bir iyileşme göstermektedir.

Kalan ve Sayit’ın çalışmasında SDN denetleyicisi tarafından video codec türü, video katman sayısı (video kalitesi) ve her katman için akış yolu seçmek amacıyla bir optimizasyon modeli kullanılmaktadır. SDN Denetleyici ayrıca, video iletimi esnasında ağdaki dalgalanmalara göre video katman sayısını ve akış yollarını yeniden belirleyebilen olay tabanlı bir sezgisel algoritma çalıştırmaktadır. İstemciler, denetleyici tarafından gönderilen, optimizasyon ve sezgisel algoritmaların çıktıları doğrultusunda uyarlama algoritması çalıştırmaktadırlar (Kalan and Sayit (2021)).

Tashtarian ve Timmerer, REVISION ile canlı ve isteğe bağlı uyarlamalı video dağıtımını iyileştirmek için videonun ilk kaynağından son kullanıcıya kadar tüm bileşenleri kapsayan geniş kapsamlı bir yol haritası çıkarmışlardır. Bu yol haritasında, video akışını iyileştirmek için üç ana bileşen ele alınmıştır: Amaç, Girdi Alanı ve Eylem Alanı. QoE, gecikme ve maliyet gibi faktörleri dikkatle dengeleyerek video akışındaki zorlukları ele alan bir yaklaşım sunmaktadır. Mimarisinde üç katman bulunmaktadır: Uygulama Katmanı, Kontrol ve Yönetim Katmanı ve Kaynak Katmanı. Bu katmanlar, istenen hedeflere

belirli kısıtlar altında ulaşmak için çeşitli parametreleri ve kaynakları ayarlayarak video dağıtımının optimizasyonu sürecini kolaylaştırmak için birlikte çalışmaktadırlar (Tashtarian and Timmerer (2024)). Bu çalışma bu alanda çalışacak olan araştırmalar için geniş açılı bir görüş oluşturmaktadır. Bizim çalışmamız ise bu bakış açısını teknik detaylar ile desteklemekte ve bu detayların optimum noktaya yaklaşmada ne kadar etkili olduğunu göstermektedir.

Yeznabad ve diğerleri, canlı uyarlamalı video dağıtımında QoE'yi iyileştirmek için kablosuz ağlarda, radyo, depolama ve hesaplama kaynaklarının müşterek tahsisine odaklanmaktadırlar. Çalışmada ağ dilimleme, Bulut Radyo Erişim Ağı (Cloud Radio Access Network - C-RAN), Çok Erişimli Uç Bilişim (Multi-access Edge Computing - MEC) ve Yazılım Tanımlı Mobil Ağ (Software Defined Mobile Network - SDMN) teknolojilerinden faydalanılmıştır. Cross-Layer QoE-Driven Bitrate Allocation (CLQDBA) isimli bir algoritma ile HTTP uyarlamalı video kullanıcılarının trafik desenleri ve istenen video kalitesi hakkında alınan bilgileri kullanarak sistem kullanımını iyileştirmeyi hedeflemektedirler. Ağ dilimi gereksinimlerini, radyo kaynak sınırlamalarını, MEC sunucularının depolama ve video kod dönüştürme kapasitesini ve kullanıcıların deneyim kalitesini göz önünde bulundurarak formüle edilmiştir Karışık Tam Sayılı Doğrusal Olmayan bir programla kullanılmaktadırlar (Yeznabad et al. (2024)).

Kırmızıoğlu ve diğerleri, Ölçeklenebilir video kodlaması (SVC) kullanan çok taraflı WebRTC video konferanslarında, heterojen bant genişliği ve terminallere sahip katılımcılar arasında bağlantı sağlamaya yarayan, seçici yönlendirme birimini (Selective Forwarding Unit - SFU) SDN destekli uç bilişim noktasına (Edge Computing) konuşlandırarak yaşanan gecikmeleri azaltmışlardır. Böylelikle uçtan uca iletişimde yaşanan gecikmeler ile ağ kapasitesi kullanımı azaltılmış ve tüm uç katılımcılar için bant genişliği garantisi sağlanmıştır (Kırmızıoğlu et al. (2025)).

3.3 Ortak Medya İstemci/Sunucu Verisi (Common Media Client/Server Data - CMC/SD) Uyumlu Çalışmalar

Bentaleb ve diğerleri, CMCD tanımlamalarına dayalı bir kavram kanıtı çalışması sunmuşlardır. Bu çalışmada, ağ bant genişliğini paylaşan ve eş zamanlı dağıtım yapılan istemcilerde CMCD'nin performans iyileştirmedeki rolü incelenmiştir. Bu amaca ulaşmak için, mevcut istemci video tampon bellek seviyesi ile tahsis edilecek bant genişliği arasındaki bağıntıyı gözetken tampon bellek farkındalığına dayalı bir bant genişliği tahsis algoritması tasarlanmış ve yapılan testlerde video kalitesinden önemli bir şekilde taviz vermeden duraksamalarda azalma olduğu görülmüştür (Bentaleb et al. (2021)).

Nguyen ve diğerlerinin CADLAD isimli çalışmasında, istemci tarafında en yüksek bitrate hesaplanarak cihaz tipi ve ekran çözünürlüğü bilgisiyle birlikte CMCD yardımıyla sunucuya gönderilmektedir. Bu bilgiler doğrultusunda istemcinin QoE değerinin arttırılması ve gönderilen verinin azaltılması için, sunucuda uygun bir bitrate kademesine karar verilmektedir (Nguyen et al. (2022)).

Pham ve diğerleri, CMSD uygulamalarına genel bir bakış sunmaktadır ve CMSD'nin dash.js ile entegrasyonuna odaklanmışlardır. CMSD'nin avantajları üç senaryo ile gösterilmiştir. Bunlar; Uç sunucunun ağ kapasitesi kullanım tahminlemesinden faydalanarak başlangıçtaki bitrate seçiminin iyileştirilmesi ile düşük gecikmeli canlı yayın sağlanması, manifesto dosyası ve segmentlerin önceden alınarak başlangıç gecikmesinin düşürülmesi ve uç sunucunun bitrate önermesinin sağlanması şeklindedir Pham et al. (2023).

Bentaleb ve diğerleri çalışmalarında, daha önceden geliştirdikleri, öğrenme tabanlı, istemcilere bitrate yönlendirmesi sağlayan Ahaggar'ı geliştirmişlerdir. Ahaggar, meta pekiştirmeli öğrenme yaklaşımı ile ağ durumu, istemci statüsü, cihaz çözünürlüğü ve dağıtım yapılan içerik bilgisini girdi olarak almakta ve bitrate seviyesi belirlenmesine rehberlik etmektedir. İstemci ve sunucular arasındaki iletişim CMCD/SD ile sağlanmaktadır. Bu çalışmada; (i) farklı ağ koşulları ve cihaz çözünürlüklerine sahip Ahaggar ve Ahaggar olmayan

istemcileri içeren heterojen bir ortamda Ahaggar'ın performansı test edilmiştir. (ii) Ahaggar ile cihaz çözünürlüklerinin QoE üzerindeki etkisi incelenerek raporlanmıştır. (iii) Ahaggar tasarım seçimleri tartışılmıştır. Bentaleb et al. (2024)

Tran ve diğerleri çalışmalarında uyarlamalı video dağıtımı için, hücresel ağlar üzerinde QoE'yi iyileştirmek amacıyla hücresel ağ operatörleri tarafından sunulan ağ API'lerini kullanarak kapasite kullanımındaki ihtiyaç artışlarını önbellek sunucusuna CMCD ile iletmışlerdir (Tran et al. (2024)).

4 OPTİMİZASYON PROBLEMİNİN MODELLENMESİ

Bu bölümde uyarlamalı video dağıtımı için, optimizasyon modelleri verilmektedir. Bu modellemeler için daha önce belirtilen QoE kriterleri kullanılarak, kullanıcı deneyim kalitesinin en iyi hale getirilmesi amaçlanmaktadır. Bölüm 4.1'de problemin bütünü ortaya konulmaktadır. Ortaya konulan bu modelleme NP-Tam bir problem içermektedir. NP-Tam olan bu problemin gevşetilerek optimale yakın bir şekilde çözülebilmesi için Bölüm 4.2'de bu çalışmanın temel unsurları olan modellemeler detaylı bir şekilde verilmektedir. Bu nedenle, gereksiz bir tekrar olmaması için, Bölüm 4.1'de verilen modelleme genel bir görüş kazandırmak için yüzeysel olarak anlatılmıştır.

4.1 İlk Modelleme

Yazılım tanımlı ağlarda uyarlamalı medya dağıtımının en iyi şekilde yapılabilmesi için bir çok QoE kriterinin gözetilmesi gerekmektedir. Bu amaçla bir optimizasyon modellemesi geliştirdik. Bu modellemede kullanılan notasyonlar Tablo 4.1'de yer almaktadır. Modelin daha iyi anlaşılabilmesi için öncelikli olarak kısıtlamalar (Formüller 4.2 - 4.12), ardından da hedef fonksiyon (Formül 4.1) açıklanmıştır. Formül 4.2'de her bir istemciye en az temel katmanın sunulma zorunluluğu getirilmektedir. Formül 4.3'te bir

istemciye gönderilecek olan bir video katmanının sadece bir sunucu tarafından sunulabileceği belirtilmektedir. Formül 4.4'te bir önceki video katmanı gönderilmeden ardından gelen video katmanının gönderilemeyeceği belirtilmektedir. Formül 4.5'te her bir istemciye gönderilecek olan video katman sayısı adil bir şekilde arttırılmaya çalışılmaktadır. Bu da F değişkeninin hedef fonksiyonda minimize edilmesiyle sağlanmaktadır. Formül 4.6'da, I_c normalize edilmiş video kalite değişim derinliği değişkenidir. $\bar{l}_c$, istemci c tarafından bir önceki zaman diliminde elde edilen video kalitesini göstermektedir. $\bar{\mu}_c$ ile c istemcisinin başlangıçtan mevcut zaman dilimine kadar olan toplam video kalitesi değişim derinliği tutulmaktadır ve $I_{max} = \max\{\bar{\mu}_c + L_c - 1 \mid \forall c \in C\}$ ifadesiyle belirlenmektedir. I_c değişkeni hedef fonksiyonda minimize edilerek video değişim derinliği azaltılmaya çalışılmaktadır. Formül 4.7'de, N_c normalize edilmiş video kalite değişim sayısının belirlendiği değişkendir. $\bar{v}_c$ ile c istemcisinin başlangıçtan mevcut zaman dilimine kadar olan toplam video kalitesi değişim miktarı tutulmaktadır. v_c ikili değişkeni ile c istemcisinin mevcut zaman diliminde video kalite değişimi olup olmadığı belirlenmektedir ($v_c L_c \geq |\sum_{x \in X} \sum_{l=1:L_c} \psi_{cl}^x - \bar{l}_c|, \forall c \in C$). N_{max} ise $N_{max} = \max\{\bar{v}_c, \forall c \in C\} + 1$ ile belirlenmektedir. N_c de hedef fonksiyonda minimize edilerek kalite değişimleri azaltılmaya çalışılmaktadır. Formül 4.8'de sunucu tarafı OF anahtarlama cihazından gönderilen bir video katmanının o cihazın bağlı olduğu OF anahtarlama cihazlarına gönderilmesinin zorunlu olduğu belirtilmektedir. Formül 4.9'da sunucu ve istemci tarafında olan OF cihazlar haricindeki OF cihazların gelen tüm video katmanlarını göndermek zorunda olduğu belirtilmektedir. Formül 4.10'da sunucu tarafı OF anahtarlama cihazından gönderilen tüm video katmanlarının istemci tarafındaki OF anahtarlama cihazına, bağlı olduğu OF cihazları tarafından ulaştırılmak zorunda olduğu belirtilmektedir. Formül 4.11'de bir veri hattından (ij) gönderilecek olan toplam veri büyüklüğünün o hattın kapasitesini geçemeyeceği belirtilmektedir. Ayrıca imkan dahilinde bu hat üzerinde Γ_{ij} kadar hat kapasitesi boş bırakılmaya çalışılmaktadır. Formül 4.1'de hedef fonksiyon yer almaktadır. Hedef fonksiyonda F minimize edilerek adil bir video kalitesi sunulmaya çalışılmaktadır. α katsayısı ile F 'in

önemi arttırılarak diğer hedeflerden daha öncelikli olması sağlanmaktadır. Aksi halde N_c ve I_c değişkenleri nedeniyle istemcilere sadece temel katman sunulabilir. β_{ic} katsayıları ile I_c ve N_c 'nin önemi belirlenmektedir. ϵ ve ω katsayıları da ilgili bölümün önem derecesini belirlemek için kullanılmıştır. $\epsilon \sum_{c \in C} \sum_{l=1:L_c} \sum_{i \in O} \sum_{j \in O} P_c b_{cl} t_{ij}^{cl}$ minimize edilerek hedefe en az hat kapasitesi kullanılarak gidilmesi hedeflenmektedir. Burada P_c istemcilerin video tampon belleklerine bakılarak belirlenen bir katsayıdır. P_c sayesinde düşük tampon belleğe sahip istemciler için diğerlerinden biraz daha fazla hat kapasitesi tahsis edilmektedir. Diğer hedefler minimize edilirken Γ_{ij} , önündeki negatif işareti sayesinde maksimize edilmektedir. Çünkü ne kadar boş hat kapasitesi elde edilirse o kadar sağlam bir iletim ortamı oluşturulabilmektedir.

$$\text{minimize et } \alpha F + \frac{\sum_{c \in C} (\beta_{1c} I_c + \beta_{2c} N_c)}{\text{len}(C)} + \epsilon \sum_{c \in C} \sum_{l=1:L_c} \sum_{i \in O} \sum_{j \in O} P_c b_{cl} t_{ij}^{cl} - \omega \Gamma_{ij}$$

$$\text{Kısıtlar : (4.2) - (4.12)}$$

$$\text{Değişkenler : } F, I_c, N_c \in [0, 1], \psi_{cl}^x, v_c, t_{ij}^{cl} \in \{0, 1\}, \Gamma_{ij} \in [0, \infty] \quad (4.1)$$

$$\sum_{x \in X} \psi_{cl}^x = 1, \forall c \in C, l = 1 \quad (4.2)$$

$$\sum_{x \in X} \psi_{cl}^x \leq 1, \forall c \in C, l = 1 : L_c \quad (4.3)$$

$$\sum_{x \in X} (\psi_{c(l+1)}^x - \psi_{cl}^x) \leq 0, \forall c \in C, l = 1 : L_c - 1 \quad (4.4)$$

$$F + \left(\frac{\sum_{x \in X} \sum_{l=1:L_c} \psi_{cl}^x}{L_c} \right) \geq 1, c \in C \quad (4.5)$$

$$I_c - \frac{\bar{\mu}_c + |\sum_{x \in X} \sum_{l=1:L_c} \psi_{cl}^x - \bar{l}_c|}{I_{max}} \geq 0 \quad (4.6)$$

$$N_c - \frac{\bar{v}_c + v_c}{N_{max}} \geq 0, \forall c \in C \quad (4.7)$$

$$\sum_{j \in \{O-X\}} e_{ij} t_{ij}^{cl} - \sum_{j \in \{O-X\}} e_{ij} t_{ji}^{cl} = \psi_{cl}^i \quad (4.8)$$

$$\forall i \in X, \forall c \in C, l = 1 : L_c$$

$$\sum_{j \in O} e_{ij} t_{ij}^{cl} - \sum_{j \in O} e_{ij} t_{ji}^{cl} = 0, \quad (4.9)$$

$$\forall i \in O - \{X \cup Y\}, \forall c \in C, l = 1 : L_c$$

$$\sum_{j \in O} e_{ij} t_{ij}^{cl} - \sum_{j \in O} e_{ij} t_{ji}^{cl} = - \sum_{x \in X} \psi_{cl}^x \quad (4.10)$$

$$\forall i \in Y, \forall c \in C, l = 1 : L_c$$

$$\sum_{c \in C} \sum_{l=1:L_c} P_c b_{cl} t_{ij}^{cl} + \Gamma_{ij} \leq b_{ij}, \quad \forall i, j \in O \quad (4.11)$$

$$lb \leq \Gamma_{ij} \leq ub \quad (4.12)$$

Her ne kadar bu model ile hedeflediğimiz QoE kriterleri modellenmiş olsa da model NP-Tam olduğu için makul bir zamanda çözümü mümkün değildir. Problemin NP-Tam bir probleme polinom zamanda indirgenebildiği aşağıda gösterilmiştir;

Problemin basitleştirilmiş halini düşünelim. Toplam video kalitesini maksimize etmek için tüm sunucu tarafı OF anahtarlama cihazlarının kısıtlı bir hat kapasitesiyle, istemci tarafı anahtarlama cihazlarına direk bağlı olduğunu varsayalım. Şimdi ağırlık olarak video dosyasının/dosyalarının ihtiyaç duyduğu hat kapasitesini, değer olarak da video kalitesini belirleyelim. Hat kapasitesini de sırt çantasının taşıyabileceği ağırlık olarak belirlediğimizde problem polinom zamanda sırt çantası problemine (knapsack problem) indirgenebilmektedir.

Bu modelin benzerleri Tashtarian et al. (2018) ve Erfanian et al. (2018)'un çalışmalarında da yapılmıştır. Ancak bu çalışmalarda kullanılan doğrusal gevşetme yöntemleriyle optimal altı sonuç elde edilmektedir. Ayrıca sadece bir sunucu veya bir sunucu kümelemesi için kullanılabilir. Çalışmamızda uyguladığımız gevşetme yöntemiyle hem optimale yakın hem de birden fazla sunucunun kullanılabilirdiği bir model geliştirilmiştir. Çalışmamızda

Bender'in Ayrıştırma Yönteminden esinlenerek asıl programlama modelimizi (Formül 4.1) Ana Programlama Modeli (APM) ve İşçi Programlama Modeli (İPM) olarak iki alt programlama modeline ayırdık. Bender'in Ayrıştırma Yöntemindeki tekrar sayısını azaltmak için de Optimal Programlama Modeli (OPM) geliştirdik. Bu modelle uçtan uca kullanılabilir hat kapasitesi tespit edilebilmektedir. Ulaştığımız bu veri ile APM'ye bir üst sınır getirerek dağıtım yapılabilecek maksimum video kalitesinin ne olabileceğini belirledik. Böylelikle Bender'in Ayrıştırma Yöntemindeki döngü tekrarı sayısını genelde 1 tekrara düşürdük. Bu döngü tekrarının azalmasını sağlayan başka bir yardımcı bileşen de Bertsimas and Sim (2003)'da yer alan Γ sağlam yaklaşımına benzer şekilde, programlama modellerine dahil ettiğimiz Γ_{ij} ve Γ_{ij}^w değişkenleridir. Bu değişkenler sayesinde her bir iletim yolunda imkan dahilinde boş hat kapasitesi bırakılabilmektedir. Bu da hem APM - İPM arasındaki döngü tekrarını düşürmekte hem de ağdaki kapasite kullanımındaki değişikliklere karşı ve/veya kapasite tespitinin doğruluğundaki sapmalara karşı önlem oluşturmaktadır.

4.2 Optimizasyon Probleminin Etkin Bir Şekilde Modellemesi ve Çözümlemesi

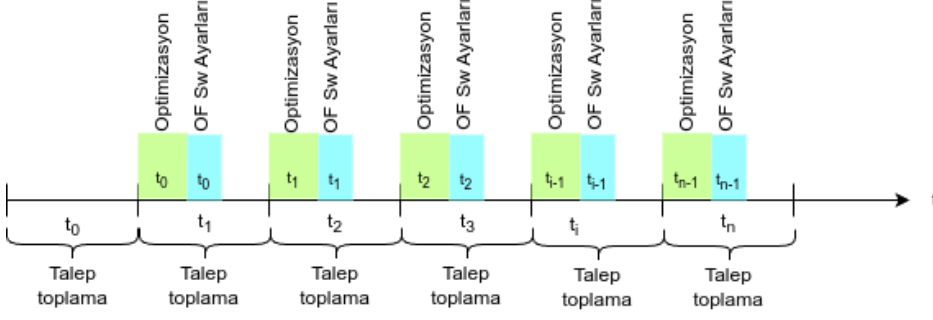
Bu çalışma, Yazılım Tanımlı Ağ (SDN) üzerinde SVC (Scalable Video Coding) yöntemi ile kodlanmış videoların dağıtımını, kullanıcı deneyim kalitesini (Quality of Experience - QoE) adil bir şekilde arttırarak, gerçek zamanlı optimize etmek için yeni bir yaklaşım sunmaktadır. Çalışmada, SDN denetleyicisi olarak ONOS, ağ simülatörü olarak MiniNet ve optimizasyon problemini modellemek ve çözmek için IBM CPLEX Optimizer kullanılmıştır. Çalışmayla önerilen optimizasyon aracı, dört ana görevi yerine getirmektedir. Bunlar;

1. Medya sunucularından, ONOS'tan ve istemcilerden gerekli olan verileri toplamak,
2. Toplanan verileri kullanarak optimizasyon problemini modellemek ve çözmek,

3. Çözüm sonuçlarını kullanarak OpenFlow (OF) anahtarlama cihazlarındaki ağ akış ayarlarını yapılandırmak,
4. İstemcilere talep edecekleri video kalite seviyelerini ve atanmış medya sunucusunu/sunucularını bildirmek.

Optimizasyon aracı ile istemciler arasında karşılıklı bilgi alışverişi, optimizasyon seviyesini önemli ölçüde artırmaktadır ve CMCD/CMSD (Consumer Technology Association (2020) Consumer Technology Association (2022)) ile benzer şekilde çalışmaktadır.

İstemciler, diğer istemcilerin farkında olmadan video kalitelerini artırmak için bant genişliği kullanımını arttırma gayreti içerisinde. TCP'nin çalışma prensibi gereği her istemci, mevcut bant genişliğinden benzer bir pay alacak şekilde eğilim göstermektedir. TCP'nin tıkanıklık kontrol mekanizmaları, TCP bağlantıları arasında eşit bant genişliği tahsisine yol açmaktadır. Örneğin, farklı çözünürlük gereksinimlerine sahip olan bir mobil telefon ve bir TV'ye benzer şekilde bant genişliği tahsisi yapılmaktadır. Ancak bu durum video dağıtımı özelinde istenilen bir durum değildir. Çünkü örneğimizde iki cihazın çözünürlük gereksinimleri farklılık göstermektedir. Dolayısıyla tahsis edilen bant genişlikleri de farklı olmak zorundadır. Bu sorunu çözmek için ağ dilimleme ve zaman dilimleme gibi çözümler kullanılabilir. Çalışmamızda, kullanıcı deneyim kalitesinin adil bir şekilde iyileştirilmesi amacıyla zaman dilimli bir yaklaşım kullanılmaktadır. Zaman iki saniyelik dilimlere (Şekil 4.1) bölünerek istemci talepleri toplanmaktadır. Her zaman diliminde, istemci istekleri toplanmakta ve zaman dilimi ile belirlenmiş sürenin sonunda (Şekil 4.1'deki "Talep toplama" kısmı), optimizasyon aracı tüm talepler için ortak bir optimizasyon algoritması çalıştırmaktadır (Şekil 4.1'deki yeşil alan). Optimizasyon işlemi sonrasında optimizasyon aracı, OpenFlow anahtarlama cihazlarını yapılandırmaktadır (Şekil 4.1'deki turkuaz alan). Son olarak istemcilere talep ettikleri segmentlerin kalite seviyeleri ve sunucu atamaları hakkında bilgi verilmektedir. Zaman dilimli yaklaşım, sistematik bir optimizasyon imkanı sağlamakta ve ağ kaynaklarının adil bir şekilde dağılımını sağlamaktadır. Çünkü istemciler arasındaki ağ kaynaklarını kullanma rekabeti belirli bir

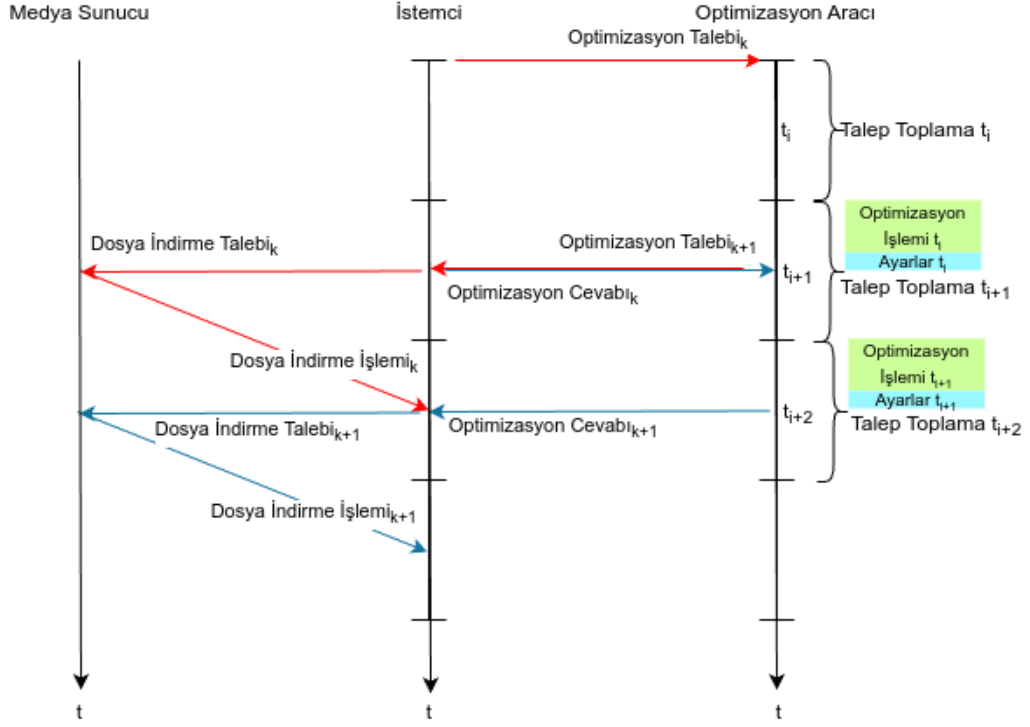


Şekil 4.1: Zaman dilimleri

süreye indirilmekte ve o sürede kullanılması gereken toplam bant genişliği belirlenmektedir. Böylelikle tüm istemciler için deneyim kalitesi adil bir şekilde arttırılmış olmaktadır.

Geliştirdiğimiz HTTP uyarlamalı video dağıtım yöntemine İngilizce’de kurbağa anlamına gelen FROG (**F**ast **R**esponse to **O**ptimization **G**oals) adı verilmiştir. FROG yakalayacağı avını bekleyen bir kurbağaya benzer şekilde çalışmaktadır. Bir kurbağanın hızlıca saldırmadan önce sabırla beklemesi gibi, FROG da talep toplanma süresi boyunca beklemekte ve doğru an geldiğinde hızla optimizasyon işlemlerini gerçekleştirmektedir, tıpkı bir kurbağanın aniden avını yakalaması gibi.

Zaman dilimleme yönteminin ve geliştirdiğimiz video dağıtım yönteminin daha iyi anlaşılması için Şekil 4.2’yi inceleyelim. Buradaki örneğimizde kolaylık sağlaması için sadece bir istemci için değerlendirme yapılmıştır. Talep toplama süresi t_i ’nin başlangıcında $OptimizasyonTalebi_k$ (segment k için yapılan optimizasyon talebi) optimizasyon aracına gönderiliyor. t_i süresinin sonunda optimizasyon işlemi ve ardından OF anahtarlama cihazlarının ağ akış ayarları yapılıyor. Sonrasında ise istemciye optimizasyon sonucu gönderiliyor. k ’nci segmentin optimizasyon talebinin cevabını alan istemci hem $k + 1$ ’inci segment için optimizasyon isteğini gönderiyor hem de k ’nci optimizasyon sonucuna göre medya sunucudan ilgili segment’in katmanını/katmanlarını talep ediyor. Medya sunucu istenen dosyaları göndermeye başlıyor. Bu indirme süreci optimizasyon aracı tarafından maksimum iki saniye sürecektir şeklinde hesaplanmaktadır. Ağda belirli bir seviyeye kadar ani bir kaynak kullanım değişikliği



Şekil 4.2: Zaman dilimleri ve etkin bant genişliği kullanımı

gerçekleşmediği sürece indirme işlemi iki saniye içerisinde sonlanacaktır. Bu esnada t_{i+1} 'inci talep toplama süreci sonlanıyor ve optimizasyon işlemine geçiliyor. Bu esnada bir önceki zaman diliminde (t_i) belirlenen optimizasyon sonuçlarına (*segment k*) göre indirme işlemi devam ediyor. Ancak bu indirme işlemi $k + 1$ 'inci segment indirilmeye başlandığında sona erecek. Bu nedenle $k + 1$ 'inci segment için t_{i+1} 'inci zaman dilimi sonunda yapılan optimizasyon esnasında k 'ıncı segment'in kullandığı ağ kaynakları kullanılmıyor olarak değerlendirilmektedir. Böylelikle ağ kaynaklarının çok daha etkin bir şekilde kullanılması sağlanmaktadır.

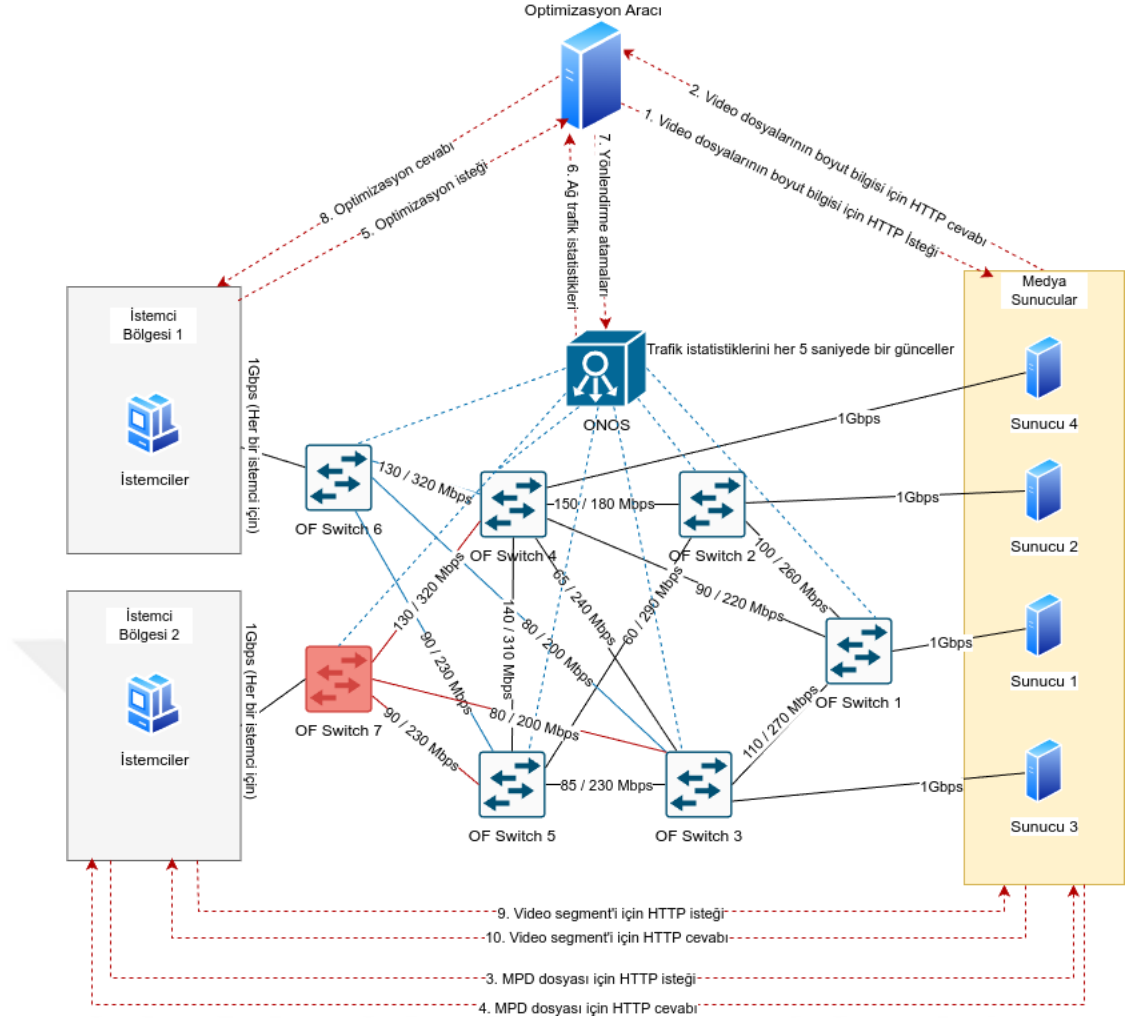
4.2.1 Ağ topolojisi ve bant genişliğinin etkin kullanımı

Çalışmamızda kullanılan ağ topolojisi Şekil 4.3'te görülmektedir. Optimizasyon sürecinin işleyişi şu şekildedir; 1. adımda optimizasyon aracı, medya sunuculardan video dosyalarının bilgilerini (dosya isimleri, boyutları, segment sayısı, MPD dosyası gibi) istemektedir. 2. adımda da bu isteğe cevap

verilmektedir. 3. ve 4. adımlarda istemciler MPD dosyalarını talep etmekte ve almaktadırlar. 5. adımda istemciler optimizasyon isteklerini optimizasyon aracına göndermektedirler. Talep toplama süresinin sonunda, 6. adımda, optimizasyon aracı HTTP REST API'yi kullanarak ONOS denetleyicisinden ağ istatistiklerini almaktadır. Daha sonra bu bilgileri kullanarak optimizasyon işlemini yapmakta ve sonucunda akış yollarını, video kalite seviyelerini ve medya sunucu bilgilerini içeren bir çözüm üretmektedir. Adım 7'de, optimizasyon aracı, akış atama isteklerini ONOS denetleyicisine göndermektedir. Adım 8'de ise istemciler için belirlenmiş olan video kalitesi ve medya sunucu/sunucuları bilgilerini göndermektedir. Akış atamaları için iki seçenek bulunmaktadır;

1. Her bir OpenFlow (OF) anahtarlama cihazı için yeni atamaların *Pending* durumundan *Added* durumuna ulaşmasını beklemek. Yapılan akış atamasının çalışır duruma gelmesi için *Added* durumuna gelmesi gerekmektedir. Ancak bu durum biraz zaman almaktadır. Optimizasyon süresinin üzerine bu bekleme süresinin eklenmesi istemcilere zamanında dönüş sağlanmamasına neden olmaktadır. Bu nedenle duraksamalar yaşanabilmektedir.
2. Her bir zaman diliminde akış önceliği (priority) artırılarak, mevcut zaman diliminde yapılan akış atamalarının bir önceki zaman diliminde yapılan akış atamalarının önüne geçmesi sağlanmaktadır. Böylelikle mevcut zaman dilimindeki atamalar etkin olana kadar (*Added* durumuna ulaşana kadar) bir önceki dönemdeki akış atamaları kullanılmaya devam edilmektedir. Bu yaklaşım, ONOS'un varsayılan akış atama uygulamasının (reactive forwarding) müdahalesini önleyerek akış atamalarının optimizasyon cihazının kontrolünden çıkmasını önlemektedir. Optimizasyon aracı, tanımlı optimizasyon hedeflerine dayalı olarak her zaman diliminde benzer yolları seçerek, akış atama stratejimizin etkinliğine katkıda bulunmaktadır. Çalışmamızda bu yöntem kullanılmıştır.

Son olarak, adım 9 ve 10'da, istemciler belirlenmiş olan video seg-

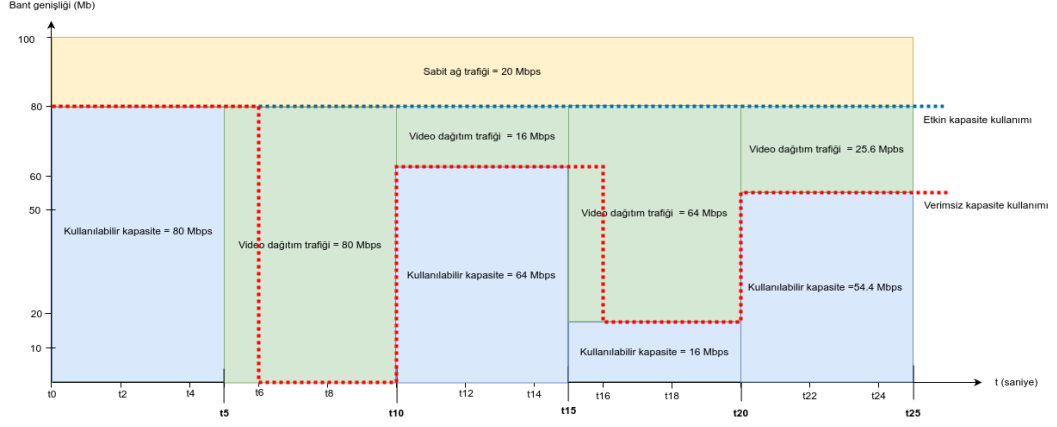


Şekil 4.3: Ağ Topolojisi

ment kalitelerine uygun olarak ilgili segmenti/segmentleri medya sunucudan/sunuculardan talep etmekte ve indirmektedirler.

Şekil 4.2 ile değindiğimiz, çalışmamızdaki önemli noktalardan birisi olan hat kapasitesinin verimli bir şekilde kullanılmasını Şekil 4.4'te detaylı olarak inceleyeceğiz. Bağlantı kapasitesinin 100 Mbps olduğunu ve anahartarlama cihazının bir portunda sabit, 20 Mbps büyüklüğünde bir trafik olduğunu varsayalım. Geriye kalan 80 Mbps'lik hat kapasitesi video dağıtımı için kullanılmaktadır. Optimizasyon aracı, her iki saniyede bir optimizasyon işlemi yapmaktadır. Kullanılabilir hat kapasitesinin hesaplanabilmesi için optimizasyon işlemine başlamadan önce mevcut hat kullanım verisine ihtiyaç duymaktadır. Bu kullanım verisini ONOS denetleyicisinden almakta

ve bu değeri bağlantı kapasitesinden (100 Mbps) çıkararak kullanılabilir hat kapasitesi bilgisine ulaşmaktadır. ONOS, anahtarlama cihazının her portu için son beş saniyede gönderilen/alınan byte sayısı bilgisini tutmakta ve talep edildiğinde bu bilgiyi göndermektedir. Optimizasyon aracı bu bilgiyi kullanarak kullanılabilir hat kapasitesi verisini güncellediğinde, kullanılabilir hat kapasitesi Şekil 4.4'te kırmızı noktalı çizgi ile gösterildiği gibi oluşmaktadır. t_0 zamanında, optimizasyon aracı ONOS denetleyicisinden son beş saniyedeki hat kullanım verisini alır ve 80 Mbps'in kullanılabileceğini görür ve bunu optimizasyon işlemine girdi olarak verir. t_2 ve t_4 zamanlarında optimizasyon aracı yeni optimizasyon işlemine başlar fakat ONOS hat kapasitesinin kullanım verisini güncellemediği için, optimizasyon aracı da optimizasyon işlemine girdi olarak verdiği hat kullanım verisi değiştirmez. t_5 zamanında, ONOS denetleyicisi kullanım bilgilerini günceller. Optimizasyon aracı t_6 noktasında yeni optimizasyon işlemine başlamadan önce ONOS denetleyicisinden çektiği hat kullanım verisinin değişmiş olduğunu görür ve optimizasyon işlemine girdi olarak verdiği hat kullanım verisini günceller. Bu noktadaki hat kullanım verisinde hat kapasitesinin %100'ünün kullanıldığı görüldüğü için video dağıtımı için kullanılabilecek bir kapasite bulunmadığını tespit eder. Ancak, video dağıtımı her iki saniyede bir başlatılmakta ve en fazla iki saniye sürmektedir. Bu süre sonunda da hat kullanımı sona ermektedir. Dolayısıyla, gerçek kullanım yalnızca sabit ağ trafiği olarak belirlediğimiz 20 Mbps'lik kısımdır. Bu durumu dikkate alarak, mevcut bant genişliğini daha doğru belirlemek için ONOS denetleyicisinden gelen kullanım verisinden bir önceki zaman diliminde video dağıtımı için kullanılan hat kapasitesi çıkarılarak mevcut kullanılabilir hat kapasitesi hesaplandığında mavi noktalı çizgi ile gösterilen kullanılabilir hat kapasitesi elde edilmektedir. İlk senaryoda, kırmızı noktalı çizginin gösterdiği şekliyle kullanılabilir hat kapasitesi belirlenirse optimizasyon aracının belirlediği video kalitesi önemli ölçüde düşecek ve çok fazla kalite değişimi olacaktır. Bu da kullanıcı deneyimini önemli ölçüde azaltacaktır. Bu nedenle çalışmamızda, etkin bir şekilde kullanılabilir hat kapasitesinin belirlendiği, mavi noktalı çizgi ile gösterilen yöntem kullanılmıştır.



Şekil 4.4: Hat kapasitesinin etkin kullanımı

4.2.2 Optimizasyon modelleri

Çalışmamızda kullanılan notasyonlar 4.1’de yer almaktadır. Notasyonlarda kullanılan *değişken* ve *sabit* terimleri programlama dillerindeki *değişken* ve *sabit*le karıştırılmamalıdır. Buradaki *sabit* aslında programlama dillerindeki *değişkene* eş değerdir. Buradaki *değişken* ise optimizasyon işlemine bir bilinmeyen değer olarak verilmekte ve optimizasyon işlemi sonucunda bir değer belirlenmesi beklenmektedir.

Optimizasyon aracında bir biriyle bağlantılı olarak çalışan dört farklı optimizasyon modeli bulunmaktadır. İlki, Optimal Programlama Modeli (OPM) olarak adlandırılan doğrusal programlama (LP) yönteminin kullanıldığı bir modeldir. İkinci ve üçüncü modeller, sırasıyla Ana Programlama Modeli (APM) ve İşçi Programlama Modeli (İPM) olarak adlandırılan modellerdir. Her iki modelde de karma tamsayılı doğrusal programlama (MILP) yöntemi kullanılmıştır. Dördüncü model, Penaltı Programlama Modeli (PPM) olarak adlandırılmakta olup, bir Tamsayılı Programlama Modelidir (IP). OPM, sunucu tarafındaki OpenFlow (OF) anahtarlama cihazlarından (X kümesine dahil cihazlar), istemci tarafındaki OF anahtarlama cihazlarına (Y kümesine dahil cihazlar) uçtan uca kullanılabilecek olan halihazırdaki hat kapasitesini hesaplamaktadır. Bu veri kullanılarak hangi anahtarlama cihazının ($x \in X$) hangi istemciye hangi video katmanını göndereceği belirlenmektedir. Buradaki yöntemde tüm sunucu tarafı OF anahtarlama cihazları en az uçtan uca hat

kapasitesine sahip cihazdan en çok hat kapasitesine sahip olan cihaza doğru sıralanmaktadır. İstemcilerin istediği videoların ilk önce birinci katmanı tüm istemcilere gönderilecek şekilde sıranın önündeki OF anahtarlama cihazından başlanarak en son cihaz hariç tüm cihazlara tahsis edilmektedir. Burada cihazların gönderim kapasitesinin üst sınırı olarak OPM ile belirlenmiş olan değer kullanılmaktadır. Bir sunucu tarafı OF cihazı gönderme kapasitesinin üst sınırına ulaşmışsa aynı işleme bir sonraki cihazdan devam edilmektedir. Bu yöntem diğer katmanlar için de tekrarlanmaktadır. Bu işlem sonucunda elimizde, istemcilere gönderim yapması için tahsis edilmemiş, en çok hat kapasitesine sahip olan son cihaz kalmıştır. Bu cihazın hangi istemciye hangi video katmanını göndereceği ise APM ile daha önceden belirlenmiş olan QoE kriterleri gözetilerek ve OPM'le hesaplanmış olan kullanılabilir hat kapasitesinden faydalanılarak belirlenmektedir. Çalışmamızda basitlik açısından, bir $x \in X$ 'e sadece bir sunucu bağlanmıştır. Odak noktamızın, sunucu kümeleme değil, ağ kullanımı olması nedeniyle bu şekilde bir mimari oluşturulmuştur.

İPM, zaman dilimiyle belirlenen süre içerisinde, APM'in bulduğu video kalite seviyesinde, istemcilere dağıtım yapıp yapılamayacağını ve bu dağıtımın hangi yolları kullanarak yapılacağını belirlemektedir. Bunu ağdaki kullanılabilir hat kapasitesine ve APM'den gönderilen video kalitelerine bakarak yapmaktadır. Geçerli bir sonuç bulunamazsa APM ve İPM arasında bir döngü başlatılmaktadır. Her bir döngüde APM tarafından bir miktar video katmanı azaltılarak optimizasyon yapılmaktadır. Bunun sonucunda İPM tarafından tekrar değerlendirilmektedir. Bu döngü bir sonuç bulunana kadar veya kabul edilebilir bir sonuç bulunamayacağına karar verilene kadar devam etmektedir. Şayet bulunan/bulunacak olan sonuç (toplam video katmanı sayısı) istemci sayısından az ise bu sonuç kabul edilebilir bir sonuç değildir.

Dördüncü ve son modelimiz olan PPM, APM'de veya APM - İPM döngüsü sonucunda kabul edilebilir bir sonuca ulaşamazsa, yani ilk video katmanı olan temel katmanı dahi gönderecek bir hat kapasitesi bulunmuyorsa kullanılmaktadır. PPM ile hat kapasitesi aşımını en aza indirecek şekilde bir modelleme ve çözüm sunulmaktadır. Böylelikle istemcilere minimum video

kalitesinin gönderimi için en iyi iletim yolları belirlenmiş olmaktadır. Bu da zorunlu olan duraksamaları en aza indirmektedir.

Optimal programlama modeli: Optimal Programlama Modeli (OPM)'nin ilk kısıtı ile (Formül 4.13) her bir sunucu tarafındaki OF anahtarlama cihazından ($\forall i \in X$) tüm istemci tarafı anahtarlama cihazlarına ($\sum_{y \in Y, y \in Y}$) uçtan uca hat kapasitesinin ilgili sunucu tarafındaki OF anahtarlama cihazının bağlantılarının toplam kapasitesini geçemeyeceği belirtilmektedir. Şayet sunucu tarafındaki OF anahtarlama cihazı başka bir sunucu tarafındaki anahtarlama cihazına bağlı ise o yöne doğru bir trafik oluşturmayaacağı için o bağlantı kullanılabilecek hat kapasitesi olarak değerlendirilmemektedir. İletim yolunda hangi sunucu tarafı OF anahtarlama cihazı hedefe daha yakınsa onun talebi karşılaması için modelleme bu şekilde yapılmıştır. Böylelikle hem hesaba katılacak olan bağlantı sayısı azaltılarak optimizasyon modellemesinin çözüm süresi azaltılmakta hem de gereksiz ağ trafiği oluşturulmasının önüne geçilmektedir.

$$\sum_{y \in Y} r_y^i \leq \sum_{j \in \{O-X\}} e_{ij} b_{ij}, \forall i \in X \quad (4.13)$$

İkinci kısıt olan Formül 4.14 ile her bir sunucu tarafı OF anahtarlama cihazından çıkıp istemci tarafı OF anahtarlama cihazına gidecek olan trafiğin kapasitesinin (r_y^i) ilgili sunucu tarafı OF anahtarlama cihazının bağlantılarından gönderilecek olan trafiğin kapasitesine ($\sum_{j \in \{O-X\}} e_{ij} f_{ij}^{iy}$) eşit olması sağlanmaktadır. Kısıttaki $\sum_{j \in \{O-X\}} e_{ij} f_{ji}^{iy}$ kısmı ile sunucu tarafı OF anahtarlama cihazına gelen toplam trafik kapasitesi belirlenmektedir. Trafik yönü istemci tarafına doğru olduğu için ve diğer kısıtlamalarda o doğrultuda yapıldığı için optimizasyon işlemi sonucunda buradaki değer sıfır olarak bulunmaktadır. Yani, video trafiğinin dağıtımında sunucu tarafına gelen bir trafik bulunmamaktadır.

$$\sum_{j \in \{O-X\}} e_{ij} f_{ij}^{iy} - \sum_{j \in \{O-X\}} e_{ij} f_{ji}^{iy} = r_y^i, \forall i \in X, \forall y \in Y \quad (4.14)$$

Üçüncü kısıt olan 4.15 ile sunucu ve istemci tarafındaki OF anahtarlama

Kümeler	Açıklamalar
O	Tüm OF anahtarlama cihazlarının kümesi $\forall o \in O$.
E	Ağ bağlantıları kümesi, $e_{ij} \in E$, $e_{ij} = 1$, i ve $j \in O$ arasında bir bağlantı olduğunu göstermektedir.
X	Sunucu tarafındaki OF anahtarlama cihazları kümesi, $\forall x \in X$.
Y	İstemci tarafındaki OF anahtarlama cihazları kümesi, $\forall y \in Y$.
M	$\forall y \in Y$ ile $\forall x \in X$ dizisi için kullanılan Map veri tipi.
C_y	$y \in Y$ 'ye bağlı olan istemci kümesi.
C	Tüm istemcilerin kümesi, $\forall c \in C$
K_x	Mevcut optimizasyon tekrarında xX tarafından gönderilecek olan video katman sayısından azaltılacak olan miktar, $\forall x \in X$.
Değişkenler	Açıklamalar
r_y^x	x 'den y 'ye uçtan uca kullanılabilecek olan maksimum hat kapasitesi, $\forall x \in X$ and $\forall y \in Y$
f_{ij}^{xy}	x 'den y 'ye giden trafiğin ij bağlantı hattından geçen kısmı, $\forall x \in X$, $\forall y \in Y$, $\forall i$ ve $j \in O$.
Γ_{ij}	OPM'de kullanılan ve bir hatta belirli bir miktar kapasitenin boş bırakılması için kullanılan sağlamlık değişkeni
Γ_{ij}^w	İPM'de kullanılan ve bir hatta belirli bir miktar kapasitenin boş bırakılması için kullanılan sağlamlık değişkeni
ψ_{cl}^x	x 'den c istemcisine l katmanının gönderilip gönderilmeyeceğinin belirlendiği ikili değişken, $\forall x \in X$, $\forall c \in C$ ve $l = 1 : L_c$.
t_{ij}^{cl}	c istemcisinin l katmanının ij bağlantı hattından geçip geçmeyeceğinin belirlendiği ikili değişken, $\forall c \in C$, $l \in L_c$, $\forall i$ ve $j \in O$.
N_c	c istemcisinin, en başından mevcut zaman dilimine kadar olan video kalite değişim miktarının normalize edilmiş değeri, $\forall c \in C$.
I_c	c istemcisinin, en başından mevcut zaman dilimine kadar olan video kalite değişim derinliğinin normalize edilmiş değeri, $\forall c \in C$. Kalite 4'den 2'ye düşmüşse veya 2'den 4'e çıkmışsa, bu değişim değeri N_c hesaplamasına 1 olarak alınırken I_c hesaplamasına 2 olarak alınır.
Sabitler	Açıklamalar
b_{cl}	c istemcisinin l katmanının iletilebilmesi için gerekli olan hat kapasitesi, $\forall l \in L_c$ ve $\forall c \in C$
b_{ij}	ij bağlantı hattının mevcut zaman dilimindeki kullanılabilir hat kapasitesi, $\forall i$ ve $j \in O$
P_c	c istemcisinin tampon bellek doluluğuna bakılarak belirlenen önceliklendirme sabiti, $\forall c \in C$
L_c	c istemcisinin izleyebileceği en yüksek video kalitesi katmanı, $\forall c \in C$
$\bar{\psi}_{cl}^x$	APM'den elde edilen ψ_{cl}^x değeri, $\forall x \in X$, $\forall c \in C$ ve $\forall l \in L_c$
$\hat{\psi}_{cl}^x$	APM'in ilk döngüsü sonucunda elde edilen ψ_{cl}^x değeri, $\forall x \in X$, $\forall c \in C$ ve $\forall l \in L_c$
$\bar{r}_y^x$	OPM'den elde edilen r_y^x değeri, $\forall x \in X$ ve $\forall y \in Y$

Tablo 4.1: Notasyonlar

cihazları haricindeki tüm anahtarlama cihazlarının, kendilerine ne kadar ağ trafiği gelmişse o kadar trafiği göndermek zorunda oldukları belirlenmektedir. $\sum_{j \in O} e_{ij} f_{ij}^{xy}$ ile i cihazına bağlantıları vasıtasıyla gelen toplam trafik, $\sum_{j \in O} e_{ij} f_{ji}^{xy}$ ile de i cihazından bağlantıları vasıtasıyla çıkan toplam trafik belirlenmektedir.

$$\sum_{j \in O} e_{ij} f_{ij}^{xy} - \sum_{j \in O} e_{ij} f_{ji}^{xy} = 0, \quad (4.15)$$

$$\forall i \in O - \{X \cup Y\}, \forall x \in X, \forall y \in Y$$

Dördüncü kısıt olan Formül 4.16 ile istemci tarafındaki OF anahtarlama cihazı i 'nin sunucu tarafındaki x OF anahtarlama cihazından gönderilen ağ trafiğinin tamamını bağlantıları vasıtasıyla almak zorunda olduğu belirtilmektedir. Burada ağ trafiğinden kasıt, hat kapasitesinin kullanımıdır. Yani, sunucu tarafındaki cihazdan istemci tarafındaki cihaza doğru oluşturulan trafikte ne kadar hat kapasitesi kullanılmışsa, istemci tarafına gelen toplam trafiğin o kadar olması gerekmektedir.

$$\sum_{j \in O} e_{ij} f_{ij}^{xi} - \sum_{j \in O} e_{ij} f_{ji}^{xi} = -r_i^x, \forall x \in X, \forall i \in Y \quad (4.16)$$

Beşinci kısıt olan Formül 4.17 ile tüm sunucu tarafı OF anahtarlama cihazı x 'lerden, tüm istemci tarafı OF anahtarlama cihazı y 'lere, ij hattı üzerinden gönderilen toplam trafiğin $(\sum_{x \in X} \sum_{y \in Y} f_{ij}^{xy})$ ij hattının mevcut kullanılabilir hat kapasitesini (b_{ij}) geçemeyeceği belirtilmektedir. Γ_{ij} değişkeni ile de ij hattı üzerinde belirli bir miktar hat kapasitesi boş bırakılmak istenmektedir. Bunun üç nedeni bulunmaktadır;

1. Ani gelişen yüksek miktardaki ağ trafiği artışlarında bir tampon vazifesi görmesi ve olası video oynatımındaki duraksamaları en aza indirmesi,
2. Kullanılabilir ağ kapasitesinin olası yanlış tahminlemelerine karşı önlem oluşturmaktadır. Bu yanlış tahminin sebebi, ONOS tarafından yapılan hat kullanım güncellemelerinin yeterince sık aralıklarla yapılmaması kaynaklı olabilmektedir.

3. Video katmanlarının ihtiyaç duyduğu hat kapasitesi üzerinden geçtiği hat kapasitesinden fazla olması durumunda APM ile İPM arasındaki döngü ile katman sayısı azaltılarak sorun çözülmektedir. Ancak Γ_{ij} ile hatta boş alan oluşturulduğu için bu sorun ile karşılaşılması en aza indirilmektedir. Bu nedenle APM ile İPM arasındaki döngü sayısını azaltmaktır. Optimizasyon işlemleri genellikle ilk döngüde sonuca ulaşmaktadır.

Altıncı kısıt olan formül 4.18 ile Γ_{ij} için alt sınır ve üst sınır belirlenmektedir. Burada video kalitesi ve duraksamalara karşı tedbirli olunması arasında bir dengeye varılması söz konusudur. Alt değer $lb = 0$ olarak belirlenirse tüm trafiği iletmek için tüm iletim hatlarında boş alan kaldığı sürece tüm hatlarda boşluk oluşturulacaktır. Ancak bu durum söz konusu değilse hatlarda herhangi bir boşluk oluşturulmayacaktır. Bu durumda hem video kalitesi mevcut imkanlar çerçevesinde en üst seviyede tutulmuş olacak hem de imkan ölçüsünde tedbirli davranılmış olacaktır. $lb > 0$ bir değer yapılırsa o ölçüde bir tedbir alınmış olunacak ancak yine o ölçüde video kalitesinden taviz verilmiş olunacaktır. ub 'in ise en fazla hat kapasitesinin yarısı kadar belirlenmesi mantıklı bir yaklaşım olacaktır. Bu değerden fazla belirlenirse de tüm hatlarda benzer bir durum olacağı için belirlenen değer sayısal olarak artsa da rezerv edilecek olan hat kapasitesine bir etkisi olmayacaktır.

Buraya kadar oluşturulan kısıtlarla uçtan uca kullanılabilen en yüksek hat kapasitesinin belirlenmesi için tüm kısıtlar belirlenmiş olmaktadır.

$$\sum_{x \in X} \sum_{y \in Y} f_{ij}^{xy} + \Gamma_{ij} \leq b_{ij}, \forall i, j \in O \quad (4.17)$$

$$lb_{ij} \leq \Gamma_{ij} \leq ub_{ij}, \forall i, j \in O \quad (4.18)$$

Hedef fonksiyonu Formül 4.19 ile verilmiştir. Buradaki amaç sunucu tarafındaki OF anahtarlama cihazlarından, kullanıcı tarafına gönderilecek olan trafik için hat kapasitesinin maksimum düzeyde tespit edilmesidir. Bunun için $-\xi_1 \sum_{x \in X} \sum_{y \in Y} r_y^x$ ifadesi kullanılmıştır. Matematiksek ifadenin başlangıcında negatif işaretin bulunmasının amacı hedef fonksiyonun *minimize*

et amacının tersi olan maksimizasyon işleminin yapılması içindir. ξ_i sabitleri ile hedef fonksiyon içerisindeki matematiksel ifadelerin ağırlığı belirlenmektedir. Buradaki asıl amacımız bu ifade ile belirtildiği için ξ_1 'e en yüksek sabit değeri atanmaktadır. $-\xi_2 \sum_{i,j \in O} e_{ij} \Gamma_{ij}$ ile iletim hatlarında rezerv olarak ayrılacak olan hat kapasitesinin imkanlar ölçüsünde maksimum seviyede olması istenmektedir. $\xi_3 \sum_{x \in X} \sum_{y \in Y} \sum_{i,j \in O} f_{ij}^{xy}$ ifadesi ile de iletim hatlarından gönderilecek olan verinin en az seviyede hatları meşgul etmesi amaçlanmaktadır. Yani, hedefe en kısa yoldan gidilmesi istenmektedir. Bu hedef fonksiyonu 4.13) - (4.18) kısıtlamaları altında ve $r_y^x, \Gamma_{ij}, f_{ij}^{xy} \geq 0$ değişkenleri kullanılarak modellenmiştir.

$$\begin{aligned} \text{minimize et } & -\xi_1 \sum_{x \in X} \sum_{y \in Y} r_y^x - \xi_2 \sum_{i,j \in O} e_{ij} \Gamma_{ij} + \xi_3 \sum_{x \in X} \sum_{y \in Y} \sum_{i,j \in O} f_{ij}^{xy} \\ \text{Kısıtlar : } & (4.13) - (4.18), \\ \text{Değişkenler : } & r_y^x, \Gamma_{ij}, f_{ij}^{xy} \geq 0 \end{aligned} \quad (4.19)$$

OPM'nin önemli faydalarından biri de istemcilerin ihtiyaç duyduğu video oynatıcısı tampon bellek artışının ne kadar yapılabileceğinin belirlenmesini sağlamasıdır. Video tampon belleklerinin artırılması gereken istemcilerin hangileri olduğu, o istemcilerin mevcut video tampon bellek durumlarına göre belirlenmektedir. Bu da CMCD benzeri bir haberleşme ile istemcilerden optimizasyon aracına iletilmektedir. Bir istemcinin tampon belleğini artırmak için, istemcinin mevcut zaman diliminde talep ettiği segment ve bu segmentten sonra gelen segmentin birlikte dağıtımının yapılması için yeterli hat kapasitesi olup olmadığına bakılarak karar verilmektedir. Yeterli hat kapasitesi varsa bu durum optimizasyon işleminde değerlendirilmektedir. Yeterli hat kapasitesi olup olmadığına da uçtan uca hat kapasitesinin maksimum ne olacağının tutulduğu $\bar{r}_y^x$ değerine bakılarak karar verilmektedir. Bu yaklaşım, bant genişliğinin yetersiz olduğu durumlarda da benzer şekilde geçerlidir. Eğer video tampon bellekleri minimum eşik değerinin üzerinde olan istemciler varsa, bu istemcilerin bir kısmının veya hepsinin talepleri mevcut optimizasyon döngüsünde dikkate alınmaz ve bir sonraki zaman dilimine ertelenirler. Bu

durum tampon belleklerinin izin verdiği sürece devam ettirilebilmektedir. Bu şekilde hata kapasite tahsisi tampon belleği yetersiz olan istemciler lehine kaydırılmış olmaktadır. Bu durum hem video kalitesinin artmasını hem de duraksamaların azaltılmasını sağlamaktadır. Burada kaç tane istemcinin bekletileceğine yine $\bar{r}_y^x$ değerine bakılarak karar verilmektedir. İstemci bilgisayarların ihtiyaç duyduğu kapasite karşılana kadar bekletilecek istemci sayısı arttırılabilir. Bu yöntemin daha yumuşak geçişler sağlayabilmesi için mevcut tampon bellek imkanı biraz daha uzun zamana yayılarak kullanılabilir. Bu durum ayrı bir tasarım konusu içermektedir. Çalışmamızda en fazla bekletilecek istemci sayısı, o istemcilerin minimum seviyenin üzerindeki tampon bellekteki segment sayısının $1/5$ 'i kadar belirlenmiştir.

Ana programlama modeli (APM): Ana Programlama Modelinin ilk kısıtı Formül 4.20 ile verilmiştir. Bu kısıtla her bir istemciye talep ettiği segmentin en az temel katmanının gönderileceği garanti altına alınmaktadır. Yeterli hat kapasitesi olmaması durumunda bu kısıt APM'in çözümsüzlükle sonuçlanmasına neden olabilecek bir kısıttır. Bu durum oluştuğunda PPM'de kullanılan modelleme ile hat kapasite aşımaları minimuma indirerek istemcilere sadece temel katmanı sunan bir çözüm geliştirilmiştir.

$$\sum_{x \in X} \psi_{cl}^x = 1, \forall c \in C, l = 1 \quad (4.20)$$

APM'nin ikinci kısıtı olan 4.21 ile bir video katmanının yalnızca bir sunucu tarafı anahtarlama cihazından gönderilebileceği belirtilmektedir. ψ_{cl}^x değişkeni 0 olduğunda c istemcisinin talep ettiği segmentin l katmanı sunucu tarafı anahtarlama cihazı x tarafından gönderilmemekte, 1 olduğu zaman ise gönderilmektedir. Bu kısıtla tüm x 'lerden gönderilecek değerlerin toplamı da en fazla 1 olabileceği için gönderim sadece bir x 'den yapılabilecektir.

$$\sum_{x \in X} \psi_{cl}^x \leq 1, \forall c \in C, l = 1 : L_c \quad (4.21)$$

APM'in üçüncü kısıtı olan Formül 4.22 ile bir alttaki video katmanı gönderilmeden onun üstündeki katmanın gönderilemeyeceğini belirtmektedir. Örneğin: 1. katman gönderilmeden 2. katman gönderilemez. $\sum_{x \in X} \psi_{c2}^x = 1$ ve

$\sum_{x \in X} \psi_{c1}^x = 0$ olamaz. Ya $\sum_{x \in X} \psi_{c2}^x = 1$ ve $\sum_{x \in X} \psi_{c1}^x = 1$ ya da $\sum_{x \in X} \psi_{c2}^x = 0$ ve $\sum_{x \in X} \psi_{c1}^x = 1$ olmak zorundadır.

$$\sum_{x \in X} (\psi_{c(l+1)}^x - \psi_{cl}^x) \leq 0, \forall c \in C, l = 1 : L_c - 1 \quad (4.22)$$

APM'nin dördüncü kısıtı olan Formül 4.23, tüm istemcilerin video kalite seviyesinin birbirine yakın olması için kullanılan bir kısıttır. Bu işlem için F değişkeni kullanılmaktadır. Bu değişken tüm istemciler için ortak bir değişkendir. F hedef fonksiyonda minimize edilmeye çalışılan bir değişkendir ve 0 ile 1 arasında bir değer almaktadır. F 0'a ne kadar yaklaşırsa istemciler arasındaki video kalitesi o kadar birbirine yakın olmaktadır. F değerinin 0'a yaklaşması için Formül 4.23'deki kesirli ifadenin payının L_c 'ye, yani alabileceği en büyük değere yaklaşması gerekmektedir. Bu durum tüm istemciler için geçerli olacağı için hepsi birbirine yakın video kalitesi elde edecektir. Burada görülmesi gereken bir başka husus da Formül 4.23'ün video kalitesinin yükseltilmesi için zorlayıcı bir unsur olmasıdır. Yani, sadece birbirine yakın video kalitesi elde edilmiyor aynı zamanda yüksek seviyede video kalitesi elde ediliyor.

$$F + \left(\frac{\sum_{x \in X} \sum_{l=1:L_c} \psi_{cl}^x}{L_c} \right) \geq 1, c \in C \quad (4.23)$$

APM'nin beşinci kısıtı Formül 4.24, video kalite değişim derinliğini minimum seviyeye çekemeye çalışmaktadır. Burada I_c normalize edilmiş video kalite değişim derinliği değişkenidir. $\bar{l}_c$, istemci c tarafından bir önceki zaman diliminde elde edilen video kalitesini göstermektedir. $\bar{\mu}_c$ ile c istemcisinin başlangıçtan mevcut zaman dilimine kadar olan toplam video kalitesi değişim derinliği tutulmaktadır ve $I_{max} = \max\{\bar{\mu}_c + L_c - 1 \mid \forall c \in C\}$ ifadesiyle belirlenmektedir. Başka bir ifadeyle başlangıçtan mevcut zaman dilimine kadar gerçekleşmiş olan video kalitesi değişim derinliği en yüksek olan istemcinin bu değeri ve mevcut zaman diliminde olası en büyük değişim derinliği I_{max} 'ı belirlemektedir. Formül 4.24'de I_c 'nin minimum seviyeye çekilmesi için mevcut zaman diliminde belirlenecek olan toplam ψ_{cl}^x değerinin $\bar{l}_c$ değerine yakın olması hatta aynı olması gerekmektedir. Böylelikle segmentler arasındaki değişim derinliği en aza indirilmeye çalışılmaktadır. Video kalitesinin artırılması ve

aynı seviyede kalması arasındaki seçim, hedef fonksiyonda ilgili değişkenlerin katsayı ağırlığı ile belirlenmektedir.

$$I_c - \frac{\bar{\mu}_c + |\sum_{x \in X} \sum_{l=1:L_c} \psi_{cl}^x - \bar{l}_c|}{I_{max}} \geq 0 \quad (4.24)$$

APM'nin altıncı kısıtı Formül (4.25) ile video kalite değişim sayısı en aza indirilmeye çalışılmaktadır. Burada N_c normalize edilmiş video kalite değişim sayısının belirlendiği değişkendir. $\bar{v}_c$ ile c istemcisinin başlangıçtan mevcut zaman dilimine kadar olan toplam video kalitesi değişim miktarı tutulmaktadır. v_c ikili değişkeni ile c istemcisinin mevcut zaman diliminde video kalite değişimi olup olmadığı belirlenmektedir ($v_c L_c \geq |\sum_{x \in X} \sum_{l=1:L_c} \psi_{cl}^x - \bar{l}_c|, \forall c \in C$). Burada, optimizasyon işlemi esnasında mutlak değer içerisi 0 olarak belirlenirse, yani mevcut video kalitesi bir önceki video kalitesiyle aynı kalırsa, v_c de 0 olarak belirlenmek zorundadır. Çünkü Formül 4.25'te v_c kesirli ifadenin payında yer almaktadır ve N_c hedef fonksiyonda en aza indirilmek istenmektedir. N_c 'nin en aza indirilebilmesi için v_c 'nin 0 yapılmasında bir engel olmadığı için v_c değişkeninin değeri 0 olarak belirlenir. v_c 'nin belirlendiği eşitsizlikte mutlak değer ifadesinin içerisi 0'dan farklı bir değer aldığı zaman v_c değişkeni 1 değerini almak zorundadır. Aksi halde eşitsizlik sağlanamaz. N_{max} sabiti, $N_{max} = \max\{\bar{v}_c, \forall c \in C\} + 1$ ile belirlenmektedir, yani en fazla video kalite değişim miktarı olan istemcinin değişim miktarına mevcut zamanda değişim olmuş kabul edilip 1 değeri eklenerek bulunmaktadır. Bunun nedeni her koşulda tüm istemciler için paydanın paydan büyük olmasını sağlamak içindir. Bir önceki kısıtta I_{max} 'ın bu kısıtta da N_{max} 'ın tüm istemciler için aynı belirlenmesinin sebebi, zorunlu bir değişim olacağı zaman tüm istemcilere eşit olarak yaklaşılması içindir. Formül 4.25'te v_c değişkenin 0 olarak belirlenmesi N_c 'yi en düşük değerine indirecektir. Dolayısıyla video kalite değişimi olmayacaktır. Tabii olarak bu durum yeterli kaynak olduğunda gerçekleşecektir. Yeterli kaynak olmaması durumunda video kalite değişimi kaçınılmaz olacaktır.

$$N_c - \frac{\bar{v}_c + v_c}{N_{max}} \geq 0, \forall c \in C \quad (4.25)$$

APM'nin yedinci kısıtı olan Formül 4.26, sunulacak olan toplam video katmanı sayısına bir üst sınır getirmektedir. Bunun nedeni ise APM ve İPM arasındaki ilk döngü sonrasında başarılı bir sonuca ulaşılamamasıdır. APM ve İPM arasındaki ilk döngüden sonra her bir sunucu tarafı OF anahtarlama cihazı ($\forall x \in X$) tarafından gönderilen toplam katman sayısı $K_x, \forall x \in X$ kadar azaltılır. K_x değeri, $x \in X$ için yapılan bir kombinatoriyal seçimle bulunur. Başlangıçta, X kümesinden bir x seçilir ve bu x 'in sunacağı toplam katman sayısından $k + n$ (K_x bileşenleri) kadar katman, APM ve İPM arasındaki yeni döngüde, APM'de Formül 4.26 kısıtı kullanılarak azaltır. (Başlangıçta $k=1$ ve $n=0$ olarak belirlenebilir. Hızlı çözüme ulaşma, video kalitesinin düşürülmesinden daha önemli ise k değeri daha yüksek seçilebilir.) APM yeniden çalıştırılır ve sonucunu ($\bar{\psi}_{cl}^x, \forall x \in X, \forall c \in C, l = 1 : L_c$) İPM'ye verilir. Şayet İPM'de çözüm bulunamazsa, kombinatoriyal seçim, APM'deki azaltma işlemi ve İPM'deki APM'den alınan veriyle bir sonuca ulaşıp ulaşılamadığı kontrolü, tüm kombinatoriyal seçimlerin tamamının üzerinden geçilene kadar tekrarlanır. Eğer hâlâ çözüm bulunamazsa, tekrar işleminde herhangi bir değişiklik yapılmaz ancak n artırılır (Örn.: $n = n + 1$). Bu döngü, bir çözüm bulunana kadar veya bulunan çözümün gerçekçi bir çözüm olmadığı belirlenene kadar (toplam katman sayısı istemci sayısından az ise bu çözüm gerçek bir çözüm olarak kabul edilmez.) devam eder. Ancak, bu algoritma ile çözüm genellikle ilk döngüde bulunmaktadır. Bunun nedenleri ise;

1. OPM ile uçtan uca kullanılabilir hat kapasitesinin bulunması ve bunun bir üst sınır olarak belirlenmesi,
2. Γ_{ij} ve Γ_{ij}^w değişkenleri kullanılarak dağıtım yollarında boş alanlar oluşturulmasıdır. Böylelikle tüm trafik öncelikli olarak en iyi yolun ve ardı sıra gelen iyi yolların tüm kapasitesini doldurmamakta ve imkan olduğu sürece trafiğin bir kısmını diğer hatlara aktarılmaktadır.

$$\sum_{c \in C} \sum_{l=1:L_c} \psi_{cl}^x \leq \sum_{c \in C} \sum_{l=1:L_c} \hat{\psi}_{cl}^x - K_x, \forall x \in X \quad (4.26)$$

APM'in sekizinci kısıtı Formül (4.27), $\bar{r}_y^x$ ile kullanılabilir hat kapasitesi

için bir üst limit sağlamaktadır. Bu kapasite sınırlandırması ψ_{cl}^x değişkenlerinin değerinin belirlenmesinde yardımcı olmaktadır. P_c katsayısı ile duraksamalar pro-aktif bir şekilde azaltılmaktadır. P_c katsayısının değeri, istemci c 'nin video tampon belleğinin önceliğine göre belirlenmektedir. Bu öncelik istemcinin tampon bellek seviyesine göre belirlenmektedir. Eğer istemcinin tampon bellek seviyesi belirli bir eşik değerin altındaysa, P_c kademeli olarak artırılmaktadır; eğer belirli bir seviyenin üzerindeyse, varsayılan değeri olan 1 korunmaktadır. Örneğin; tampon bellek seviyesi 0 olan istemcinin P_c değeri 1.1 olarak belirlenebilir. Böylelikle istemciye ihtiyaç duyduğu hat kapasitesinden daha yüksek bir hat kapasitesi ihtiyacı olduğu bilgisi optimizasyon aracına verilecektir. Bu şekilde o istemciye daha uygun yol ataması yapılacaktır. Bu işlem, tampon bellek seviyesi 0 olan istemcilerde veri indirim hızını arttırdığı için mevcut sorunlu durumdan daha çabuk çıkılmasına, tampon bellek seviyesi 1 olan istemciler içinse sorunlu duruma düşülme olasılığının azaltılmasına yardımcı olmaktadır. Tüm istemciler için benzer artışlar yapılması önceliklendirme işleminin etkisini azaltacağı için tampon belleği 1'den yüksek olan istemcilerde P_c değeri 1 olarak korunmuştur. b_{cl} değeri ise c istemcisi tarafından talep edilen segmente ait katmanların büyüklüğü arzu edilen iletim süresine (2 saniye) bölünerek bulunmaktadır. Ayrıca hat kapasitesinin daha etkin kullanılması için daha önce bahsedilen, ağdaki iletim protokollerinin getirdiği ek yük dosya boyutuna eklenmektedir.

$$\sum_{c \in Cy} \psi_{cl}^x P_c b_{cl} \leq \bar{r}_y^x, \forall x \in X, \forall y \in Y \quad (4.27)$$

APM'nin hedef fonksiyonu Formül (4.28), F değişkenini minimize ederek istemci video kalitelerini adil bir şekilde en iyi seviyeye çıkarmayı, I_c değişkenini minimize ederek c istemcisinin video kalite değişim derinliğini minimize etmeyi ve N_c değişkenini minimize ederek de c istemcisinin video kalite değişimi minimize etmeyi amaçlamaktadır. α katsayısı ile, F değişkenin önceliği belirlenmektedir. Bu değer yeterince yüksek belirlenmesi ile video kalite seviyelerinin I_c ve N_c değişkenlerinin etkisiyle düşük seviyede kalmasının önüne geçilmektedir. Testlerimiz esnasında bu değer 25-30 aralığında be-

lirlendiğinde en iyi sonuçların alındığı görülmüştür. I_c ve N_c değişkenlerinin önceliği ise β_{1c} ve β_{2c} katsayıları ile belirlenmektedir. Testlerimiz esnasında bu katsayılar sırasıyla 3 ve 7 olarak belirlenmiştir.

$$\begin{aligned} \mathbf{min.} \quad & \alpha F + \frac{\sum_{c \in C} (\beta_{1c} I_c + \beta_{2c} N_c)}{\text{len}(C)} \\ \text{Kısıtlar :} \quad & (4.20) - (4.27) \end{aligned} \quad (4.28)$$

$$\text{Değişkenler : } F, I_c, N_c \in [0, 1], \text{ and } \psi_{cl}^x, v_c \in \{0, 1\}$$

İşçi programlama modeli (İPM): İşçi Programlama Modelinin ilk kısıtı olan Formül 4.29 ile, istemci c tarafından talep edilen segmentin l katmanının, sunucu tarafı OF anahtarlama cihazı i 'den gönderilmesi durumunda belirtilen katmanın bu anahtarlama cihazının bağlantıları üzerinden diğer OF anahtarlama cihazlarına (diğer sunucu tarafı anahtarlama cihazları hariç) gönderilmesi sağlanmaktadır. t_{ij}^{cl} değişkeni ile aynı zamanda c istemcisinin l katmanının hangi yollardan geçerek hedefe ulaşacağı belirlenmektedir. Optimizasyon işlemi sonucunda t_{ij}^{cl} değeri 1 olarak belirlenmişse, c 'nin l katmanının i anahtarlama cihazından j anahtarlama cihazına gönderileceği anlamına gelmektedir. e_{ij} , sadece birbiriyle bağlantısı olan anahtarlama cihazlarını işleme almak için kullanılmıştır.

$$\begin{aligned} \sum_{j \in \{O-X\}} e_{ij} t_{ij}^{cl} - \sum_{j \in \{O-X\}} e_{ij} t_{ji}^{cl} &= \bar{\psi}_{cl}^i \\ \forall i \in X, \forall c \in C, l &= 1 : L_c \end{aligned} \quad (4.29)$$

İPM'in ikinci kısıtı Formül 4.30'da görülmektedir. Burada, eğer c istemcisinin l katmanı, sunucu tarafı ve istemci tarafı OF anahtarlama cihazları haricinde bir OF anahtarlama cihazına ulaşırsa ($\forall i \in O - \{X \cup Y\}$), bu katmanın o anahtarlama cihazının bağlı olduğu diğer OF anahtarlama cihazlarına gönderilmesini sağlamaktadır. Bu da t_{ij}^{cl} ve t_{ji}^{cl} değişkenlerinin toplamalarının birbiriyle eşit olmasıyla mümkündür.

$$\begin{aligned} \sum_{j \in O} e_{ij} t_{ij}^{cl} - \sum_{j \in O} e_{ij} t_{ji}^{cl} &= 0, \\ \forall i \in O - \{X \cup Y\}, \forall c \in C, l &= 1 : L_c \end{aligned} \quad (4.30)$$

İPM'nin üçüncü kısıtı Formül 4.31, c istemcisine gönderilen video katmanlarının c istemcisinin bağlı olduğu OF anahtarlama cihazına ulaşmasını sağlamaktadır. Bu kısıtla, anahtarlama cihazı $i \in Y$ 'ye gelen tüm bağlantıların toplamından gönderilen video katmanlarının $(\sum_{j \in O} e_{ij} t_{ji}^{cl})$, tüm sunucu tarafı OF anahtarlama cihazlarından gönderilen video katmanı sayısına $(\sum_{x \in X} \bar{\psi}_{cl}^x)$ eşit olması zorunluluğu getirilmektedir. $i \in Y$ 'den gönderilen bir video katmanı olmadığı için optimizasyon işlemi sonucunda $\sum_{j \in O} e_{ij} t_{ji}^{cl} = 0$ olacaktır.

$$\sum_{j \in O} e_{ij} t_{ji}^{cl} - \sum_{j \in O} e_{ij} t_{ji}^{cl} = - \sum_{x \in X} \bar{\psi}_{cl}^x \quad (4.31)$$

$$\forall i \in Y, \forall c \in C_i, l = 1 : L_c$$

İPM'nin dördüncü kısıtı (Formül 4.32) ile hat kapasitesi sınırı getirilmektedir. ij hattındaki trafiği mevcut bant genişliği b_{ij} ile sınırlamaktadır. Ayrıca akış yolu değişkeninin (t_{ij}^{cl}) belirlenmesine yardımcı olmaktadır. P_c katsayısı APM'de kullanıldığı şekliyle burada da aynı amaç için kullanılmaktadır. Burada kullanılan Γ_{ij}^w , daha önce OPM'de kullandığımız Γ_{ij} ile aynı amaç için kullanılmaktadır. Bu iki değişkenin katsayıları aynı olsa dahi optimizasyon sonucunda döndürülen değerleri farklı olabilir. Çünkü bu değişkenler imkan ölçüsünde arttırılmaktadır ve OPM ve İPM'in şartları birbirinden farklı olarak belirlenmiştir. Beşinci kısıt (Formül 4.33) ile de Γ_{ij}^w değişkeninin alt ve üst sınırı belirlenmektedir.

$$\sum_{c \in C} \sum_{l=1:L_c} P_c b_{cl} t_{ij}^{cl} + \Gamma_{ij}^w \leq b_{ij}, \forall i, j \in O \quad (4.32)$$

$$lb \leq \Gamma_{ij}^w \leq ub \quad (4.33)$$

İPM'nin amaç fonksiyonu (Formül 4.34), kaynak $x \in X$ 'ten hedef $y \in Y$ 'ye olan akış yolunu belirlemektedir. Bu amaçla gereksiz trafiği azaltarak en kısa akış yolunu belirlemek için $\sum_{c \in C} \sum_{l=1:L_c} \sum_{i \in O} \sum_{j \in O} P_c b_{cl} t_{ij}^{cl}$ ifadesini minimize etmektedir. Γ_{ij}^w değişkenini maksimize ederek de imkan ölçüsünde iletim hatlarında boş kapasite alanları oluşturmaktadır.

$$\begin{aligned} \text{minimize et } & \sum_{c \in C} \sum_{l=1:L_c} \sum_{i \in O} \sum_{j \in O} P_c b_{cl} t_{ij}^{cl} - \Gamma_{ij}^w \\ \text{Kısıtlar : } & (4.29) - (4.33) \end{aligned} \quad (4.34)$$

$$\text{Değişkenler : } \Gamma_{ij}^w \in [0, \infty], \text{ and } t_{ij}^{cl} \in 0, 1$$

Penaltı programlama modeli (PPM): PPM'de hat kapasitesi aşımaları en aza indirilmektedir. Kapasite aşımalarının en aza indirilmesinin önemini vurgulamak için hedef fonksiyon 4.35'te ζ katsayısı kullanılmaktadır.

$\zeta^{max} \cdot \left\{ \sum_{i \in O} \sum_{j \in O} \sum_{c \in C} e_{ij} b_{cl} t_{ij}^{cl} - b_{ij}, 0 \right\}$ kısmında her bir iletim hattındaki kapasite aşım miktarıyla 0 değeri karşılaştırılmakta ve aralarında hangisinin değeri büyükse o tercih edilmektedir. 0 seçilirse bir aşım söz konusu değildir ki 0 seçilmiştir. Diğer değer seçilirse zorunlu bir aşım söz konusudur. Çünkü bu hedef fonksiyonu en aza indirme işlemi olduğu için zorunlu olmadıkça 0'ı seçecektir. Kapasite aşımı zorunluysa her bir iletim hattında en az aşım yakalanmaya çalışılmaktadır. Ayrıca $\sum_{i \in O} \sum_{j \in O} \sum_{c \in C} e_{ij} b_{cl} t_{ij}^{cl}$ minimize edilerek ağ üzerinde gereksiz trafik oluşturulması önlenmektedir. Sadece ilk video katmanı gönderileceği için $l = 1$ olarak belirlenmiştir. Kısıtlar 4.36 - 4.38, İPM 4.29 - 4.31'le aynı amaçla kullanılmaktadır. Buradaki tek fark sadece bir video katmanı için işlem yapılmasıdır.

$$\text{minimize et } \sum_{i \in O} \sum_{j \in O} \sum_{c \in C} e_{ij} b_{cl} t_{ij}^{cl} + \zeta^{max} \cdot \left\{ \sum_{i \in O} \sum_{j \in O} \sum_{c \in C} e_{ij} b_{cl} t_{ij}^{cl} - b_{ij}, 0 \right\}, l = 1$$

$$\text{Kısıtlar : } (4.36) - (4.38)$$

$$\text{Değişken : } t_{ij}^{cl} \in \{0, 1\} \quad (4.35)$$

$$\sum_{j \in \{O-X\}} e_{ij} t_{ij}^{cl} - \sum_{j \in \{O-X\}} e_{ij} t_{ji}^{cl} = \bar{\psi}_{cl}^i, \forall i \in X, \forall c \in C, l = 1 \quad (4.36)$$

$$\sum_{j \in O} e_{ij} t_{ij}^{cl} - \sum_{j \in O} e_{ij} t_{ji}^{cl} = 0, \forall i \in O - \{X \cup Y\}, \forall c \in C, l = 1 \quad (4.37)$$

$$\sum_{j \in O} e_{ij} t_{ij}^{cl} - \sum_{j \in O} e_{ij} t_{ji}^{cl} = - \sum_{x \in X} \bar{\psi}_{cl}^x, \forall i \in Y, \forall c \in C_i, l = 1 \quad (4.38)$$

4.2.3 Optimizasyon Algoritması

Şu ana kadar ele aldığımız optimizasyon modellerinin birbiriyle uyumlu bir şekilde çalışması Algoritma 1 ile sağlanmaktadır. Algoritma 1, her bir istemci için mevcut şartlar altında, en iyi katman kalitesini elde etmek ve her bir katman için akış yolunu belirlemek amacıyla tasarlanmıştır. Ayrıca optimizasyon modelleri çözüldükten sonra, akış yolları atanmakta ve istemcilerin hangi sunucudan hangi kalitede video talep edeceği bilgisi istemcilere gönderilmektedir.

1. satırda, Sunucu anahtarlama cihazlarından ($x \in X$) istemci tarafı anahtarlama cihazlarına ($y \in Y$) uçtan uca maksimum hat kapasitesini belirlemek için, OPM (Formül 4.19) çalıştırılmaktadır ve sonucu ($\bar{r}_y^x, \forall x \in X, \forall y \in Y$), APM'ye iletilmektedir. Ayrıca her bir $y \in Y$ için kullanılabilir hat kapasitesine göre tüm $x \in X$, azalmayan bir sırada sıralanmakta ve bu sıralama, harita M 'de tutulmaktadır ($M : Y \rightarrow X^n, M(y) = (x_1, x_2, \dots, x_n), \forall x \in X$). Ana döngü Satır 2'te başlamakta ve 29. satırdaki istisnalar haricinde bir çözüm bulunana kadar devam etmektedir. 3. satırda, her bir $y \in Y$ için M haritası ziyaret edilmektedir. M haritası iki boyutlu bir dizi gibi de düşünülebilir. 4. satırda ise $M[y]$ 'de yer alan ilgili $x \in X$ dizini ziyaret edilmektedir. Yani, her bir istemci tarafındaki OF anahtarlama cihazına ($y \in Y$) hangi sunucu tarafı OF anahtarlama cihazlarının ($x \in X$) ne kadar gönderim yaptığını belirlemek için iki tane iç içe döngü oluşturulmuştur. Bununla her bir istemci tarafı anahtarlama cihazı için bütün sunucu tarafı anahtarlama cihazları gezilmektedir. 5. satırda, döngüde üzerinde işlem yapılan sunucu tarafı anahtarlama cihazı tarafından istemcilerin kullanımına ayrılan toplam hat kapasitesinin ne kadar olduğunu tutmak için `fixed_rate` değişkeni oluşturulmaktadır. Üzerinde işlem yapılan $M[y]$ 'nin üzerindeki son x üzerinde işlem yapılıyorsa döngüden çıkılmakta ve 22. satırdan devam edilmektedir (6. satır). Böylelikle ilgili $y \in Y$ 'ye en yüksek hat kapasitesiyle hizmet

Algorithm 1: Optimizasyon Algoritması

Data: $M, \bar{r}_y^x, \forall \psi_{cl}^x, \forall t_{ij}^{cl}, X, Y, O, C_y$
Result: $\forall \psi_{cl}^x, \forall t_{ij}^{cl}$ 'in optimize edilmiş hali

```

1 OPM'i çalıştır ve sonucuna göre ( $\bar{r}_y^x$ ) her bir  $M[y]$ 'yi azalmayan şekilde sırala
2 while sonuç bulunana kadar do
3   foreach  $M$ 'deki  $y$  do
4     foreach  $M[y]$ 'deki  $x$  do
5       double  $fixed\_rate \leftarrow 0.0$ 
6       if  $x, M[y]$ 'in son elemanıysa then döngüden çık (break)
7       if  $\bar{r}_y^x == 0$  then
8         foreach  $c \in C_y$  do
9           for  $l \leftarrow 1$  to  $L_c$  do  $\psi_{cl}^x \leftarrow (0,0)$ 
10        else
11          int  $cnt \leftarrow 0$ 
12          for  $l \leftarrow 1$  to  $L_c$  do
13            foreach  $c \in C_y$  do
14              if  $\psi_{cl}^x, 1$ 'e sabitlenmemişse then
15                 $fixed\_rate += P_c b_{cl}$ 
16                if  $fixed\_rate \leq \bar{r}_y^x$  then
17                  if ilk tekrarsa then  $\psi_{cl}^x \leftarrow (1,1)$ 
18                  else if ilk tekrar değilse and
19                     $cnt < \sum_{c \in C} \sum_{l \in L_c} \hat{\psi}_{cl}^x - K_x$  then
20                       $\psi_{cl}^x \leftarrow (1,1)$ 
21                       $cnt++$ 
22                  else  $\psi_{cl}^x \leftarrow (0,0)$ 
23           $x \leftarrow M[y]$ 'nin son elemanı
24          foreach  $c \in C_y$  do
25            for  $l \leftarrow 1$  to  $L_c$  do
26              if  $\psi_{cl}^x, 1$ 'e sabitlenmemişse then  $Ilo\_Expr += \psi_{cl}^x b_{cl}$ 
27           $APM.add(Ilo\_Expr \leq \bar{r}_y^x)$ 
28          if ilk döngü değilse then  $APM.add(\sum_{c \in C} \sum_{l \in L_c} \psi_{cl}^x \leq \sum_{c \in C} \sum_{l \in L_c} \hat{\psi}_{cl}^x - K_x)$ 
29          APM'yi çalıştır
30          if "APM'de çözüm yoksa" then break
31          Sonuçları ( $\bar{\psi}_{cl}^x$ ) İPM'ye ver ve İPM'i çalıştır
32 if "Çözüm varsa" then
33   OF cihazları ayarla, video kalite ve sunucu bilgisini gönder
34 else
35   PPM' çalıştır, OF cihazları ayarla, video kalite ve sunucu bilgisini gönder

```

eden sunucunun göndereceği video katmanlarıyla ilgili bir sabitleme yapılması önlenmiş olur ki QoE kriterlerine uygun olarak değerlendirilmeden taviz verilmemiş olunsun. Bütün $x \in X$ ve $y \in Y$ 'ler için geçerli olmak üzere, şayet bir $x \in X$ 'ten bir $y \in Y$ 'ye gönderim yapılacak hat kapasitesi yoksa ($\bar{r}_y^x == 0$) ilgili ψ_{cl}^x değişkenlerinin alt ve üst değeri 0 yapılarak o değişkenin alacağı değer sabitlenir ($\psi_{cl}^x \leftarrow (0, 0)$) (7-9. satırlar). Böylelikle optimizasyon yapılacak değişken sayısı azaltılmış olmaktadır. Şayet bir $x \in X$ 'ten bir $y \in Y$ 'ye gönderim yapılacak hat kapasitesi varsa (10. satır) 11. satırda **cnt** isimli bir değişken oluşturulmaktadır. **cnt** değişkeni, bir sunucu tarafından gönderilmek üzere ataması yapılan video katmanların ($\psi_{cl}^x \leftarrow (1, 1)$ işlemi) sayısını tutmak için bir sayaç olarak kullanılmaktadır. 12. satırda, ilk video katmanından başlayıp tüm katmanları dolaşacak olan bir dış döngü, 13. satırda ise işlem yapılan $y \in Y$ 'ye bağlı tüm istemcileri dolaşacak bir iç döngü başlatılmaktadır. Böylelikle hat kapasitesinin kullanımı için öncelikle tüm istemcilere ilk katmanlar gönderilmekte, şayet kullanılacak hat kapasitesi kalırsa diğer katmanlardan devam edilmektedir. 14. satırda, tekrarlı atamalar önlenmektedir. 15. satırda Hali hazırda ziyaret edilen istemciye gönderilecek olan video katmanının belirlenen sürede gönderilebilmesi için gerekli hat kapasitesi b_{cl} ve istemcinin bellek öncelik katsayısı P_c çarpımı, **fixed_rate** değişkenine eklenmektedir. 16. satırda bu değer, uçtan uca mevcut hat kapasitesi $\bar{r}_y^x$ ile karşılaştırılmaktadır ve hat kapasitesi aşılmamışsa 17. satırdan devam edilmektedir. Şayet döngünün ilk tekrarıysa ψ_{cl}^x değişkeninin alt değeri ve üst değeri 1 yapılarak ($\psi_{cl}^x \leftarrow (1, 1)$) optimizasyona gerek kalmadan değeri belirlenmektedir (17. satır). Şayet döngünün ilk tekrarı değilse ve **cnt** değeri ilk döngü ile gönderilen toplam video katmanı sayısının, K_x eksiğinden küçükse ($\hat{\psi}_{cl}^x - K_x$) (18. satır) ψ_{cl}^x değeri 1'e sabitlenir ($\psi_{cl}^x \leftarrow (1, 1)$) (19. satır) ve bu sabitlemelerin sayısını tutan **cnt** değeri bir arttırılır (**cnt**++) (20. satır). 18-20. satırlardaki işlemlerle bir önceki döngüde hat kapasitesinin yetersiz olmasından dolayı, toplam video katman sayısı azaltılmaktadır. Şayet 16. satırdaki kriter karşılanamamışsa, yani artık kullanılabilir hat kapasitesi kalmamışsa, 21. satırda ψ_{cl}^x değişkeninin değeri 0'a sabitlenmektedir ($\psi_{cl}^x \leftarrow$

$(0,0)$). $M[y]$ 'deki son OF anahtarlama cihazına ulaşıldığında, 4. satırda başlatılan döngüden 6. satırdaki *break* ile çıkılarak 22. satıra düşülmektedir. Daha önce sabitlenmemiş ψ_{cl}^x değişkenlerinin her biri bir *Ilo Expression*'a¹ eklenmekte (23-25. satırlar) ve bu *Ilo Expression* APM'ye eklenmektedir (Satır 26). Burada *Ilo Expression*'nın r_y^x değerini geçmemesi gerektiği de belirtilmektedir. Şayet döngünün ilk tekrarı değilse bu anahtarlama cihazından gönderilecek olan toplam video katmanı sayısının da azaltılması gerekmektedir. Bu işlem de 27. satırda gerçekleştirilmektedir. 28. satırda APM çalıştırılmakta ve bir çözüme ulaşamamışsa veya ulaşılan çözümdeki toplam video katmanı sayısı istemci sayısından az ise *while* döngüsünden çıkılarak (29. satır) 31. satırdan devam edilmektedir. Şayet 29. satır kriterleri dışında kalırsa 30. satırda APM çıktıları $(\bar{\psi}_{cl}^x, \forall x \in X, \forall c \in C, l = 1 : L_c)$ İPM'ye verilerek İPM çalıştırılmaktadır. Buradan *while* döngüsünün başına dönülür. Şayet İPM'de bir çözüm bulunamamışsa, süreç 2. satırdan 30. satıra kadar tekrarlanır. Şayet İPM'de bir çözüme ulaşılmışsa döngüden çıkılarak 31. satıra geçilir. Bir çözüm varsa (31. satır) OF anahtarlama cihazlarına video dağıtımı için gerekli olan akış atamaları yapılmakta (bu atamalar için t_{ij}^{cl} değişkenlerinden elde edilen sonuçlar kullanılmaktadır.) ve istemcilerin hangi kalitede video talep edecekleri ve bunu hangi sunucu/sunuculardan isteyecekleri bilgisi istemcilere gönderilmektedir (32. satır). Şayet bir çözüme ulaşamamışsa istemcilere en düşük video kalitesinde dahi olsa bir hizmet sunmak için PPM çalıştırılır ve belirlenen iletim yollarının ayarları OF anahtarlama cihazları üzerinde yapılır. Ardından istemcilere video kalite bilgisi ve hizmet alacakları sunucu bilgisi gönderilir.

¹*Ilo Expression*, IBM CPLEX'in bir parçası olarak doğrusal programlama modellerinde kullanılan matematiksel bir yapı olup, değişkenler ve katsayılar arasındaki doğrusal ilişkileri temsil eder. Bu ifadeler, amaç fonksiyonlarını ve kısıtları tanımlamak için kullanılır. Örneğin, bir toplam veya çarpım işlemine dayalı olarak belirli değişkenlerin doğrusal kombinasyonlarını oluşturmak için kullanılır. *Ilo Expression*, CPLEX'in modelleme API'sinin bir parçasıdır ve programatik olarak kısıtlar ve optimizasyon hedefleri oluşturmayı kolaylaştırır IBM (2024).

5 PERFORMANS DEĞERLENDİRMESİ

Değerlendirme ortamı, Ubuntu 20.04.6 LTS işletim sistemi, Mininet 2.3.0 simülatörü, IBM ILOG CPLEX Optimization Studio 22.1.1, ONOS 3.0.0.b, AMD Ryzen 5 5500U işlemci (Radeon grafik özellikli) ve 32 GiB sistem belleğinden oluşmaktadır. Ağ topolojisi Şekil 4.3'te gösterilmektedir. İki istemci alanına sahip konfigürasyonlarda OF anahtarlama cihazı 7 kullanılmakta ve istemciler bu kurulumda eşit olarak dağıtılmaktadır. Standart testlerde her bir OF anahtarlama cihazı arasında bağlantı için bant genişliği kapasitesi, 20 ve 30 istemcili konfigürasyonlarda 100 Mbps, 50 istemcili kurulumda ise 250 Mbps olarak belirlenmiştir. Ayrıca hat kapasiteleri Poisson dağılımına göre belirlenerek de testler yapılmıştır. Bu dağılıma göre belirlenen hat kapasiteleri Şekil 4.3'te ilgili hatların üzerinde görülmektedir. Bu hat kapasiteleri "20 ve 30 istemcili kapasite / 50 istemcili kapasite" formatında belirtilmiştir. İstemci ve sunucu ile OF anahtarlama cihazları arasındaki bağlantılar, çalışmanın ağ akışına odaklanmasını sağlamak amacıyla 1 Gbps olarak ayarlanmıştır. Testlerde tutarlılık sağlamak için sunucu tarafındaki ve istemci tarafındaki anahtarlama cihazlarının toplam hat kapasiteleri her iki topoloji için de aynı tutulmuştur. Standart topolojide toplam kapasite 300 Mbps'sa Poisson dağılımına göre belirlenen toplam kapasite de 300 Mbps olarak ayarlanmıştır.

Performans değerlendirmemizde bir sunucu tarafı anahtarlama cihazı, dört sunucu tarafı anahtarlama cihazı, bir istemci tarafı anahtarlama cihazı ve iki istemci tarafı anahtarlama cihazı kullanılarak bir çok farklı test yapılmıştır. Bu testler aşağıdaki şekilde sınıflandırılmıştır;

1. Bir sunucu tarafı OF anahtarlama cihazı (anahtarlama cihazı 1) ve bir istemci tarafı OF anahtarlama cihazı (anahtarlama cihazı 6),
2. Dört sunucu tarafı OF anahtarlama cihazı (anahtarlama cihazları 1, 2, 3 ve 4) ve bir istemci tarafı OF anahtarlama cihazı (anahtarlama cihazı 6),
3. Dört sunucu tarafı OF anahtarlama cihazı ((anahtarlama cihazları 1, 2, 3 ve 4) ve iki istemci tarafı OF anahtarlama cihazı (anahtarlama cihazları

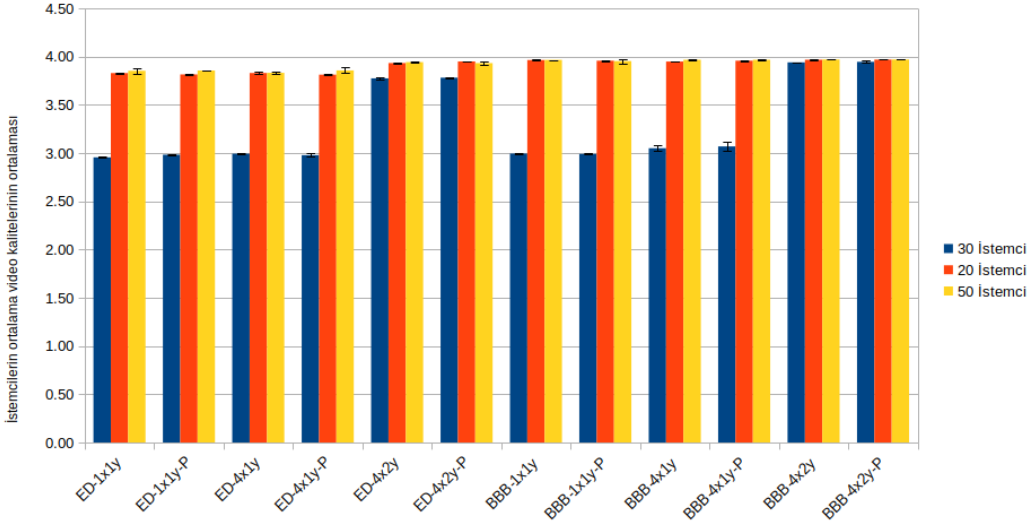
6 ve 7)

Her değerlendirme için BBB-I-1080p ve ED-I-1080p SVC video setleri kullanılmıştır itec.aau.at (2015a,b).

5.1 FROG'un Değerlendirilmesi

FROG kullanıldığında, her bir istemcinin elde ettiği ortalama video kalitelerinin ortalaması, ED ve BBB SVC video setlerinin kullanıldığı çeşitli senaryolar için Şekil 5.1'de gösterilmektedir. Bu bağlamda, **ED-1x1y**, ED seti kullanılarak bir sunucu tarafı OF anahtarlama cihazı (1x) ve bir istemci tarafı OF anahtarlama cihazı (1y) kullanılarak yapılan değerlendirmeyi, **BBB-4x2y** ise, BBB video seti kullanılarak dört sunucu tarafı anahtarlama cihazı (4x) ve iki istemci tarafı anahtarlama cihazı (2y) kullanılarak yapılan değerlendirmeyi temsil etmektedir. **P**, Poisson-dağılımlı bant genişliği kapasiteleriyle yapılan testleri belirtmektedir. Şekil 5.1'de görülen diğer testler ve takip eden değerlendirmeler için de aynı gösterim formatı kullanılmıştır.

Şekil 5.1'de görüldüğü gibi 20 istemci ve 50 istemci kullanılarak yapılan test sonuçları oldukça benzerdir. Çünkü her iki yapılandırmadaki istemciler için ayrılan ortalama hat kapasitesi aynıdır ($100 \times 3 = 300$ Mbps / 20 = 15 Mbps ve $250 \times 3 = 750$ Mbps / 50 = 15 Mbps). Ancak, 30 istemci ile yapılan testlerde, istemci başına düşen bant genişliği azaldığı için video kalitesinde düşüş gözlemlenmektedir. Bunun nedeni, hat kapasitesinin, 20 istemci yapılandırmasındaki gibi kalmasına rağmen fazladan 10 istemcinin ağı dahil edilmesidir. İstemci tarafı anahtarlama cihazlarının sayısının ikiye çıkarılması ve istemcilerin eşit şekilde dağıtılmasıyla daha iyi video kalitesi elde edildiği **BBB/ED-4x2y(P)** testlerinde görülmektedir. Bu normal bir durumdur, çünkü yapılan değişiklik istemci başına düşen hat kapasitesinin artmasını sağlamaktadır. ED video seti, BBB video setine göre daha fazla bant genişliğine ihtiyaç duyduğu için, test sonuçlarında, BBB video setiyle yapılan testlerde ED video setiyle yapılan testlerden daha iyi video kalitesi elde edilmektedir.



Şekil 5.1: FROG’da istemcilerin ortalama video kalitelerinin ortalaması

Şekil 5.2’de istemcilerin toplam duraksama sürelerinin ortalama değerleri görülmektedir. Bu değerler genellikle sıfırdır ve en yüksek değerin 0,2 saniyenin altında olduğu görülmektedir. Yok denilecek kadar az olan bu duraksama değerleri, uyarlamalı medya dağıtımı için kullanılan diğer istemcilerle yapılan testlerle karşılaştırıldığında buradaki sonuçların çok iyi düzeyde olduğu görülecektir. Bu karşılaştırma bu tez raporunun ilerleyen aşamalarında yapılacaktır. Yine de bu az miktardaki duraksamanın nedenlerini açıklamak gerektiğini değerlendiriyoruz. Bu duraksamalara neden olması muhtemel etkenler aşağıdaki şekildedir;

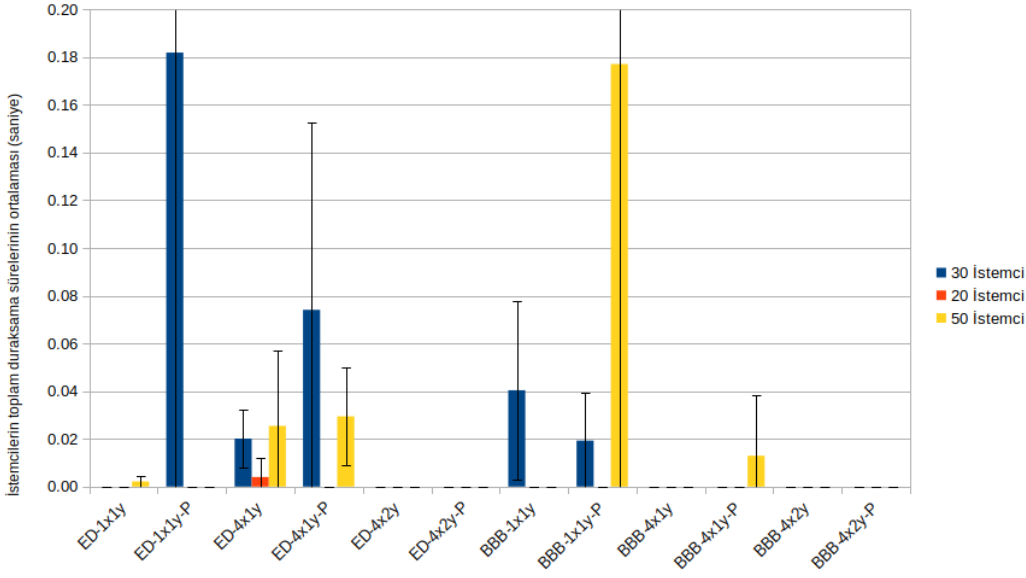
1. Birincil ve en büyük neden yetersiz hat kapasitesidir.
2. İkinci nedenin hat kapasitesi bilgisinin anlık olarak alınamaması olduğunu değerlendiriyoruz. Etkin bant genişliği (Bkz. Şekil 4.4) kullanım yöntemiyle, hat kapasitesinin eldeki veriler ışığında mümkün olan en üst düzeyde kullanılması ve ONOS denetleyicisi tarafından sağlanan kapasite kullanım bilgisinin nispeten uzun (beş saniyelik) güncelleme aralığı nedeniyle doğru sonuçtan az da olsa sapmalar olabilmektedir.
3. Optimizasyon aracının maksimum çalışma süresini aşması da bir etken olarak karşımıza çıkmaktadır (Bkz. Şekil 5.6). Optimizasyon aracının maksimum çalışma süresi bir video segment oynatma süresi kadar

belirlenmiştir. Bu süre aşıldığında bir optimizasyon cevabı bekleyen istemcinin tampon belleğindeki segmentler tükenebilmekte ve duraksama yaşanabilmektedir.

Çalışmamızda bu olumsuz etkiler şu şekilde hafifletilmektedir;

1. Zorunlu hat kapasitesi aşımını en aza indiren optimizasyon modeli olan PPM kullanılarak.
2. OPM ile uçtan uca kullanılabilir hat kapasitesi bilgisinin elde edilmesi ve bu bilgidен istemcilerin tampon bellek artırımında veya o tampon belleğin kullanımında faydalanılmasıyla.
3. Tampon bellek azlığı sıkıntısı yaşayan bir istemciye daha fazla bant genişliği tahsis edilmesiyle, ilgili video segmentinin dağıtımının daha hızlı gerçekleşmesine yardımcı olan P_c katsayısının kullanımıyla.
4. Birden fazla modelleme kullanarak, büyük bir problem olan uyarlamalı video dağıtım optimizasyon problemini küçük parçalara ayırarak, problem karmaşıklığının azaltılması ve böylece çalışma süresinin düşürülmesiyle. APM ve İPM bu büyük uyarlamalı video dağıtım optimizasyon probleminin ikiye bölünmüş ve bir döngü içerisinde çalışan halidir. İlerideki testlerde problem çözümünün ne kadar hızlandığı görülecektir.
5. OF anahtarlama cihazlarının yapılandırılması için ONOS'a REST API ile gönderilen komut sonrasında yapılandırılmanın etkin hale gelmesi biraz zaman almaktadır. Yapılandırma için geçen bu süreyi kısaltmak ve bu süreyi video dağıtımı için kullanmak amacıyla bir biri ardına gelen zaman dilimlerinde OF anahtarlama cihazlarının yapılandırılmasında akış öncelik değerinin (*flow priority*) arttırılmasıyla.

İstemcilerin toplam video kalite değişim derinliğinin ortalaması Şekil 5.3'te görülmektedir. Maksimum kalite değişimi ED SVC setinde meydana gelmiş olup, ED setinin BBB setine göre daha fazla bant genişliğine ihtiyaç duyduğunu göstermektedir ki gerçek durum da zaten o şekildedir (Bkz. Tablo 5.1). ED seti, BBB setinden %10 daha fazla bant genişliğine ihtiyaç



Şekil 5.2: FROG’da istemcilerin toplam duraksama sürelerinin ortalaması

duymaktadır ($6.08/5.51 = 1.10$). Bant genişliği gereksinimi azaldıkça, bunun doğal neticesi olarak kalite değişim ihtiyacının da azaldığı bu örnekte rahatlıkla görülebilmektedir. Bu durum video kalitesi ve duraksamalarda da benzer şekilde karşımıza çıkmaktadır. Bu eğilimin, özellikle bir istemci tarafı anahtarlama cihazı ve iki istemci tarafı anahtarlama cihazı yapılandırmaları karşılaştırıldığında belirginleştiği görülmektedir. İki istemci tarafı anahtarlama cihazı kullanılarak oluşturulan yapılandırmada hat kapasitesi önemli ölçüde yükseldiğinden dolayı bu iyileşme gerçekleşmektedir. Minimum kalite değişimi 3’tür, çünkü başlangıç katmanları temel katman olarak indirilmektedir ve bu durum temel katman ile en iyi katman arasında 3 fark oluşmasına neden olmaktadır. Yeterli bant genişliği olduğunda, optimizasyon ve beraberindeki tampon bellek yönetimimiz sayesinde başlangıç segmentleri nedeniyle oluşan kalite değişimi haricinde kalite değişim ihtiyacı duyulmamaktadır (BBB-4x2y ve BBB-4x2y-P örnekleri) veya minimum miktarda kalite değişimi gerçekleşmektedir. Ayrıca istemci başına düşen hat kapasitesi aynı olsa da toplam hat kapasitesinde artış olduğunda daha iyi video kalitesi yönetimi yapılabildiği 20 istemci ve 50 istemci karşılaştırılmasıyla görülebilmektedir.

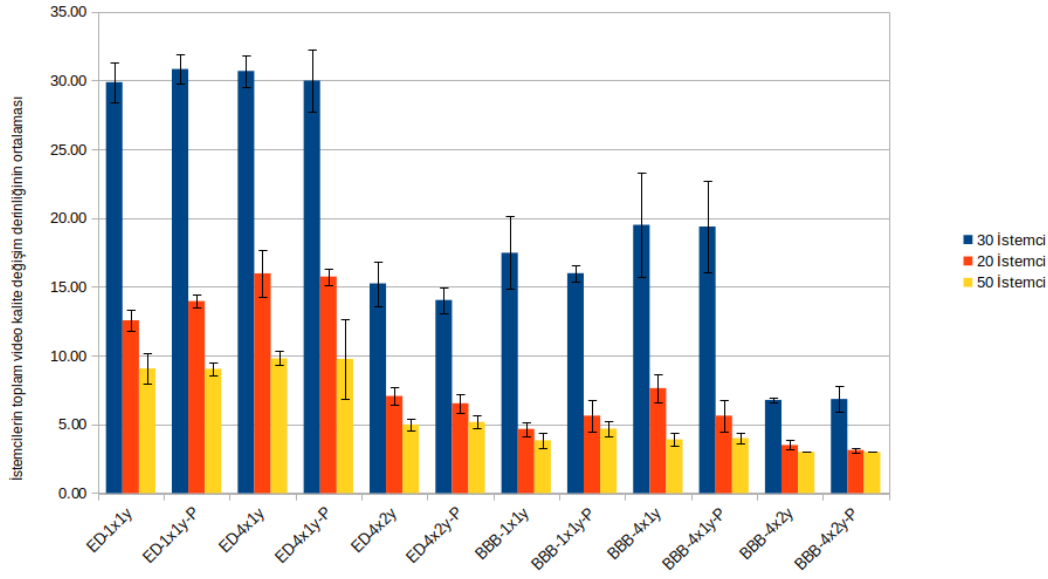
İstemcilerin ortalama kalite deneyim (QoE) değerleri, her bir test için

Tablo 5.1: Video setlerinin ortalama dosya büyüklükleri (Mb)

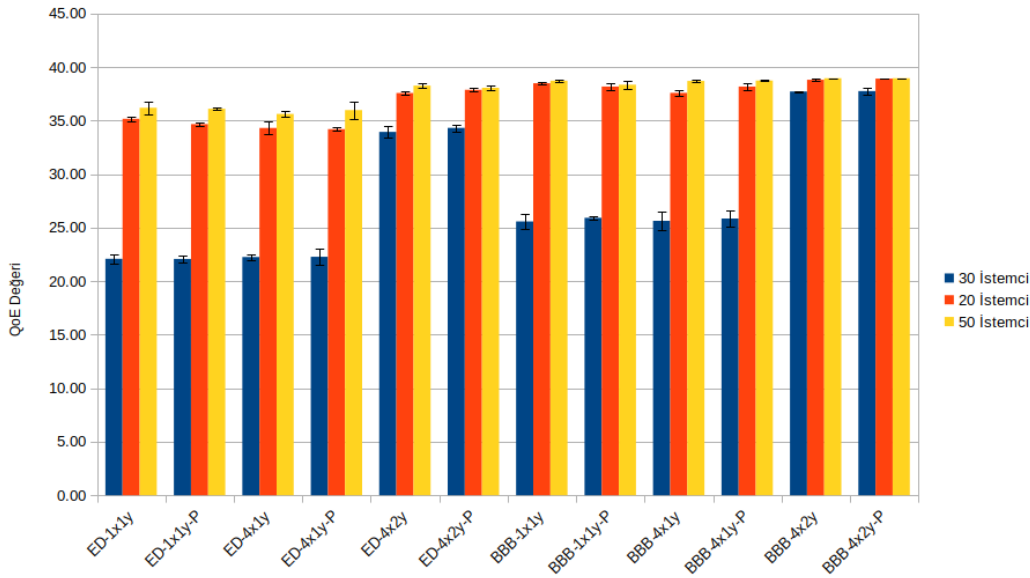
Video Seti	SVC Katmanları	Ortalama Dosya Büyüklüğü (Mb)
ED-I-1080p	0	9.06
	1	3.13
	2	3.20
	3	8.94
	Ortalama	6.08
BBB-I-1080p	0	8.04
	1	3.16
	2	3.10
	3	7.73
	Ortalama	5.51

Şekil 5.4'te gösterilmektedir. Her bir istemci için Formül 5.1 kullanılarak QoE değeri hesaplanmış ve tüm istemciler için bu değerlerin ortalaması alınarak Şekil 5.4'te verilmiştir. Burada Q_i , segment i 'nin video kalitesini, p_i segment i için duraklama süresini, s toplam segment sayısını ve α , β ve γ sırasıyla video kalitesi, video kalite değişim derinliği ve duraklama süreleri için ağırlık faktörlerini temsil etmektedir. Çalışmamızda $\alpha = 10$, $\beta = 0,25$ ve $\gamma = 0,25$ olarak alınmıştır. Duraksamaların negatif etkisinin sıfır veya sıfıra yakın olması (Bkz. Şekil 5.2), video kalite değişiminin sınırlı bir seviyede kalması ve benzer şartlarda benzer davranışlar göstermesi nedenleriyle, QoE değerleriyle video kalite değerleri (Bkz. Şekil 5.1) benzer bir desen sergilemektedir. Çalışmamızdaki QoE değerinin maksimum seviyesi 40'tır ve bu durum başlangıç segmentleri dahil tüm segmentlerin en üst düzeyde dağıtımıyla ve hiç duraksama olmaması durumunda mümkündür. Kaynaklar yeterli olduğu ölçüde bu seviyeye oldukça yakın sonuçlar elde edildiği Şekil 5.4'te görülmektedir.

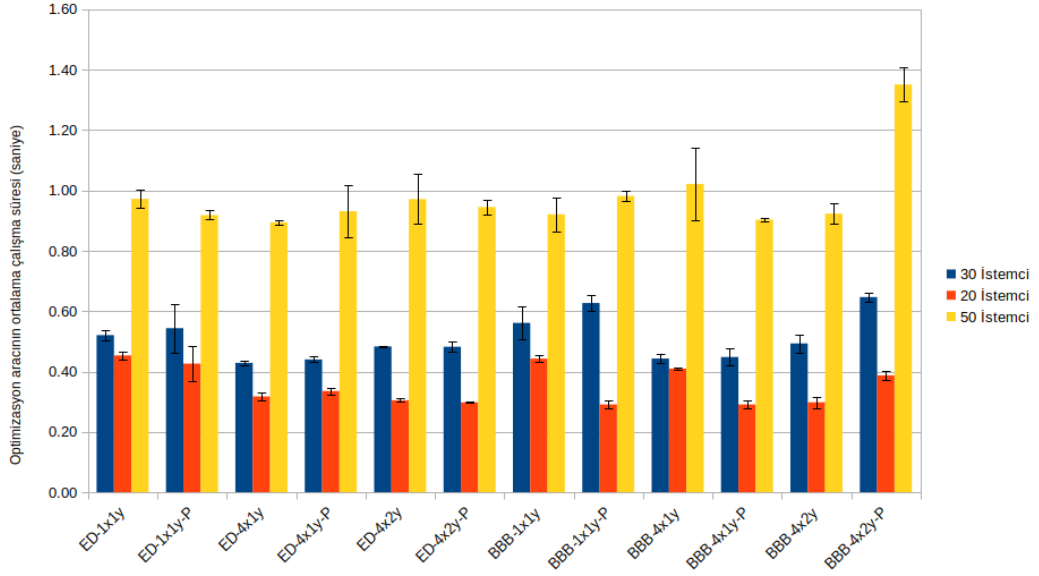
$$QoE = \alpha \frac{\sum_{i=1}^s Q_i}{s} - \beta \frac{\sum_{i=1}^s |Q_i - Q_{i-1}|}{s} - \gamma \frac{\sum_{i=1}^s p_i}{s} \quad (5.1)$$



řekil 5.3: FROG’da istemcilerin toplam video kalite deęiřim derinlięinin ortalaması



řekil 5.4: FROG’un ortalama kalite deneyimi (QoE) deęerleri

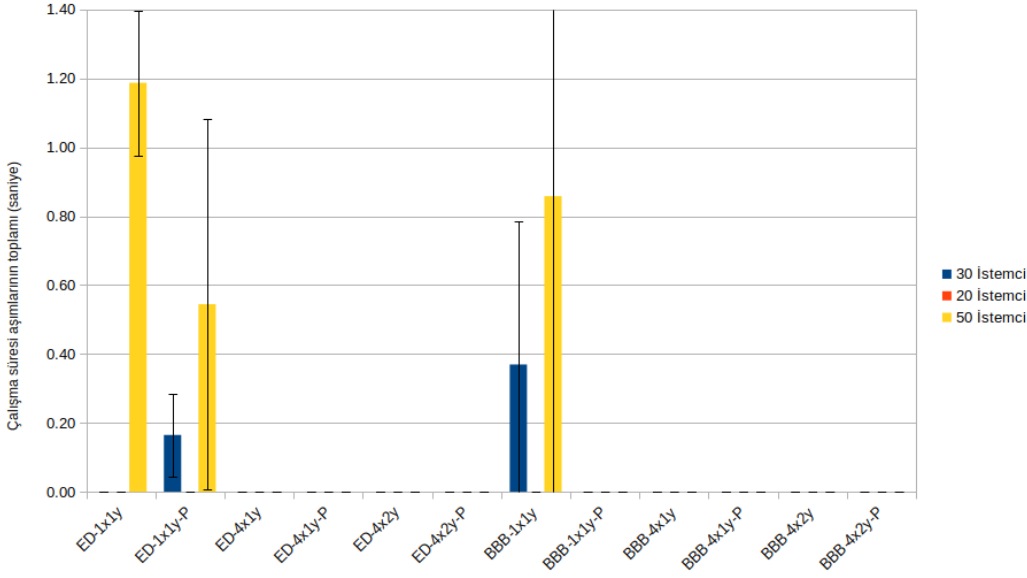


Şekil 5.5: FROG'un ortalama çalışma süreleri

FROG'un optimizasyon aracının ortalama çalışma süreleri Şekil 5.5'te gösterilmiştir. Beklendiği gibi, istemci sayısı arttıkça çalışma süreleri de artmaktadır ancak deneyimsel test için diz üstü bilgisayarda bile 50 istemciye kadar olan testlerde başarılı sonuçlar alınmıştır. Kullandığımız diz üstü bilgisayarda 50 istemci simülasyonunun üzerine çıkılamamıştır. Burada çalışma sürelerinin istemci sayısı arttıkça artması bize çalışma sürelerinin ağdaki hat kapasitesinin yeterli olup olmamasından bağımsız olarak değiştiğini göstermektedir. Başka bir deyişle, sadece ağdaki istemci başına düşen ortalama hat kapasitesinin azalmasının optimizasyon aracının çalışma süresine etkisi bulunmamaktadır veya sınırlı kalmaktadır.

Optimizasyon çalışma süresinin, duraksamalara neden olmaması için video segmentinin oynatma süresini aşmaması gerekmektedir. Segment süresi aşımalarının toplamı Şekil 5.6'da gösterilmiştir. Genel olarak, çalışma süresi aşımları meydana gelmemektedir; ancak bazı durumlarda küçük aşımalar yaşanmaktadır. Bu aşımaların duraksamalar üzerindeki etkisi minimal düzeydedir ve genellikle istemcilerin tampon bellekleri ile duraksamalar önlenmektedir.

Ortalama video başlatma süresini, kullanılabilir hat kapasitesi bilgisini ($b_{ij} \forall i, j \in O$) kullanan basit bir algoritma ile düşük seviyelerde tutabiliyoruz.

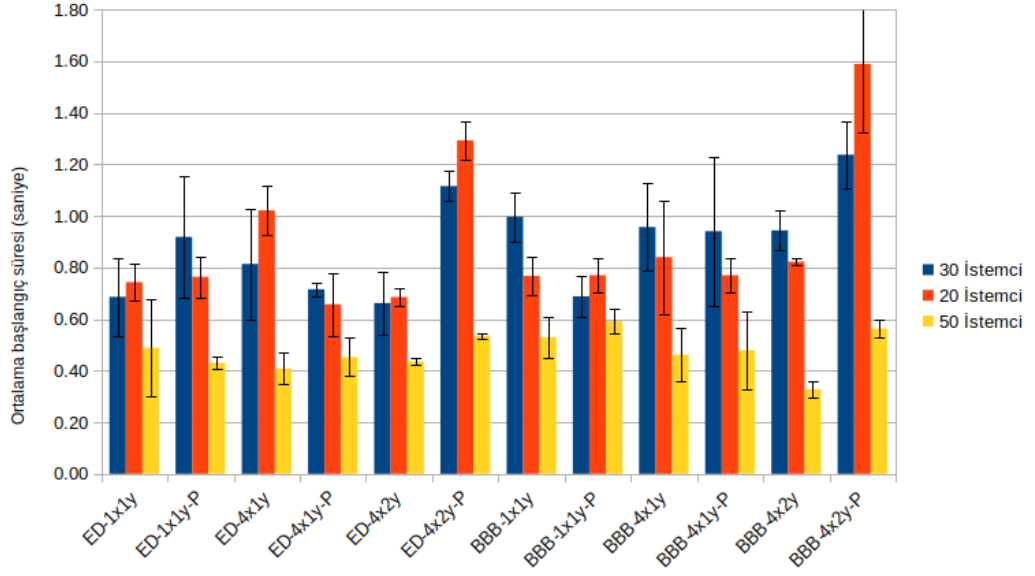


Şekil 5.6: FROG’da çalışma süresi aşımalarının toplamı

Algoritmamız her ne kadar basit bir algoritma olsa da tüm ağdaki kullanılabilir hat kapasitesi etkin bir şekilde tespit edildiği için başlangıç sürelerini genellikle 1 saniye civarında ve her zaman 2 saniyenin altında tutmayı başarmaktadır. Bu algoritma, yeni bir istemci talepte bulunmaya başladığında, ilk üç segmentin temel katmanlarının dağıtımı için en yüksek hat kapasitesinin olduğu yolu seçerek OF anahtarlama cihazları üzerinde gerekli akış atamalarını yapmaktadır. Ortalama başlangıç süreleri Şekil 5.7’de gösterilmiştir.

5.2 FROG ve Diğer İstemcilerin Karşılaştırılması

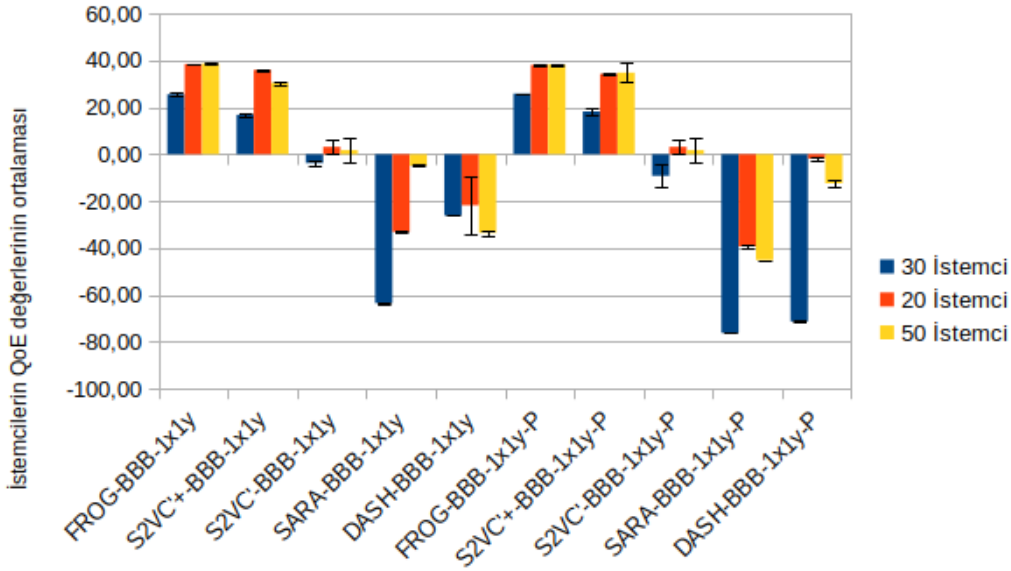
FROG ve diğer uyarlamalı video dağıtım istemcileri arasındaki karşılaştırmalar, statik topoloji (100 Mbps (20-30 istemci için) / 250 Mbps (50 istemci için)) ve Poisson-dağılımlı topoloji için, her birinde bir sunucu tarafı anahtarlama cihazı ve bir istemci tarafı anahtarlama cihazı bulunan testler yapılmıştır. Testler BBB-I-1080p ve ED-I-1080p SVC video setleri için ayrı ayrı yapılmıştır. Bu testlerden QoE karşılaştırmasının yapıldığı testlerin sonuçları Şekil 5.8 ve Şekil 5.9’da gösterilmektedir. Her iki değerlendirme de benzer desenler göstermekte olup, en iyi performansı *FROG* sergilerken, ikinci sırada *S²VC’* yer almaktadır. Bu ikisini karşılaştırdığımızda, istemci başına düşen



Şekil 5.7: Ortalama video başlatma süresi

ağ kaynaklarının en az olduğu, 30 istemciyle ED SVC video seti² kullanılarak yapılan testlerde *FROG*'un en yakın rakibi *S²VC'*+’dan %71.69 daha iyi performans gösterdiği, istemci başına düşen ağ kaynaklarının en fazla olduğu 20 istemciyle BBB SVC video seti için yapılan testlerde ise *FROG*'un en yakın rakibi *S²VC'*+’dan %6.85 daha iyi performans gösterdiği görülmektedir (Bkz. Tablo 5.2). Aradaki farkın bu kadar değişken olmasının temel nedeni ağ kaynakları azaldıkça ve/veya istemci sayısı arttıkça *S²VC'*+’ın optimizasyon aracının çalışma süresinin uzamasıdır. Bu da istemcilerin tampon belleğindeki video segmentlerinin tükenmesine dolayısıyla da duraksamalara neden olmaktadır. *S²VC'*’ın performansının *S²VC'*+’ya karşı çok kötü olmasının tek sebebi *S²VC'*’da hat kapasitesinin etkin olarak kullanılmamasıdır. *SARA* ve *DASH* istemcilerin performanslarının çok daha kötü çıkmasının sebebi ise bu istemcilerin ağ desteği olamamasıdır. Topolojimizde kaynaktan hedefe toplamda 300 / 750 Mbps hat kapasitesi bulunmaktadır. Ancak bu iki istemcinin ağ desteği olmaması nedeniyle ONOS tarafından sağlanan varsayılan akış

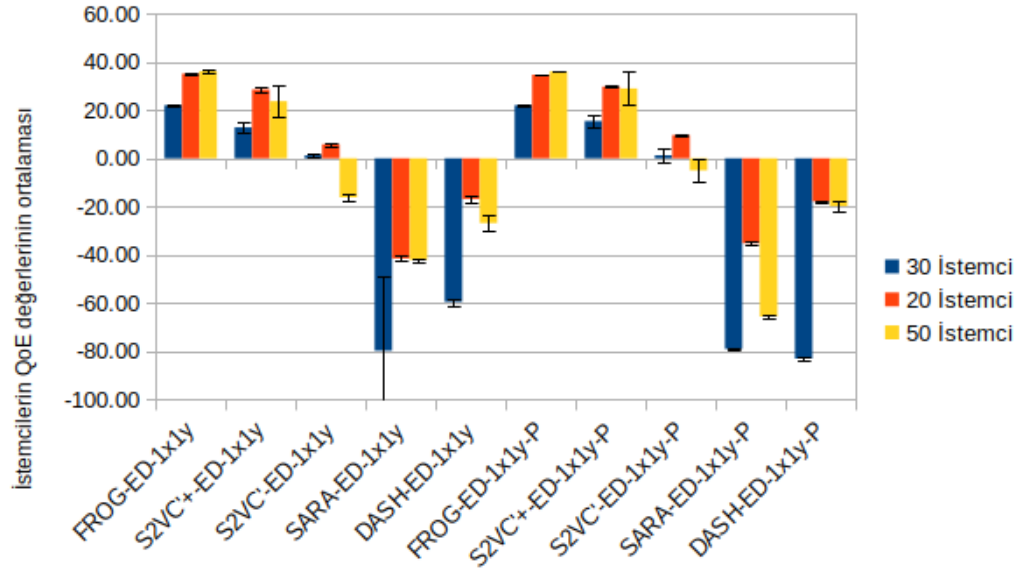
²ED setinin ortalama dosya büyüklüğü açısından BBB setinden daha büyük olduğu Tablo 5.1’de gösterilmiştir. Bu büyüklük %10 ($6.08/5.51 = 1.10$) kadardır. Dolayısıyla ED seti BBB setinden %10 daha fazla bant genişliğine ihtiyaç duymaktadır.



Şekil 5.8: 1x1y'nin BBB video seti için QoE karşılaştırması

yönlendirmesiyle ancak 100 / 250 Mbps hat kapasitesini kullanabilmektedirler.

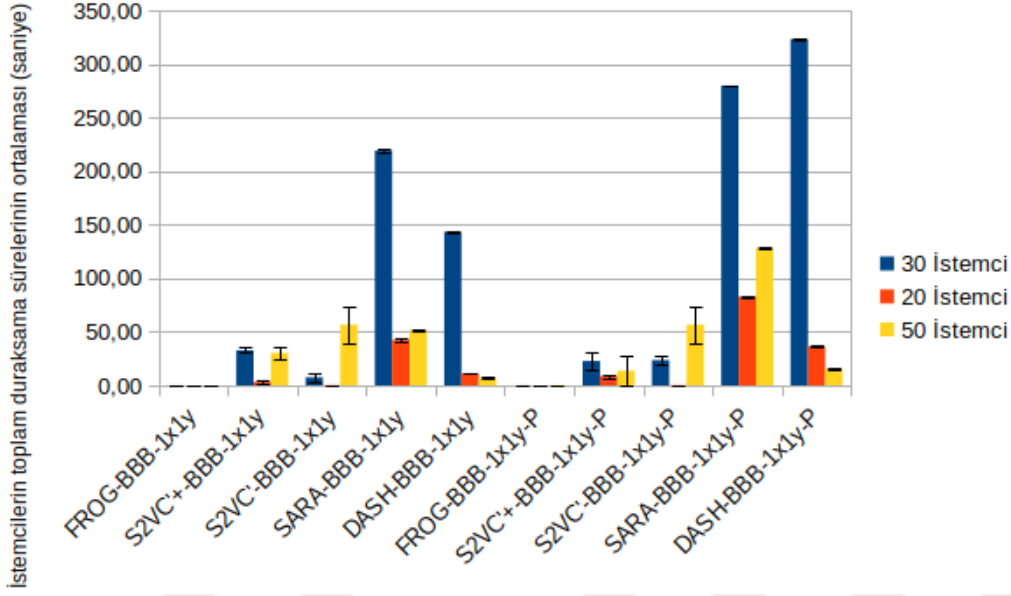
Yukarıdaki paragrafta bahsedilen duraksamalar Şekil 5.10 ve Şekil 5.11'te, bu duraksamaların sebebi olarak karşımıza çıkan optimizasyon araçlarının çalışma sürelerinin, segment oynatma süresini aşımalarının toplamaları ise Şekil 5.12'de görülmektedir. Daha önceden de bahsedildiği gibi *SARA* ve *DASH* istemcilerinin ağ desteği olmamasından dolayı temel video katmanını indirmede bile zorluklar yaşamaktadırlar. ONOS tarafından belirlenen ağ yolu (hop count) ile sadece belirli bir yol kullanıp geriye kalan yollardan habersiz olan bu istemcilerin kötü performans sergilemeleri de kaçınılmazdır. Mevcut tüm ağ kapasitesinden haberdar olunsa bile bu kapasitenin kullanımının doğru şekilde yapılması gerekmektedir. Bu durumun ne kadar önemli olduğu *S²VC'* ile *S²VC'+* karşılaştırıldığında açıkça görülmektedir. Her iki istemci arasındaki tek fark *S²VC'+*'da, *FROG*'da olduğu gibi ağ kaynaklarının etkin bir şekilde kullanılmasıdır. QoE değerlendirmesinde olduğu gibi duraksamalarda da *FROG* açık ara önde yer almaktadır. Bunun temel sebebi *FROG*'da çalışma süresinde yaşanan süre aşımalarının çok daha az olması ve daha önceden bahsedilen duraksamalara karşı alınan önlemlerdir (P_c , Γ vb.). *FROG*'un en yakın takipçisi olan *S²VC'+*'da istemci başına yaşanan ortalama duraksama



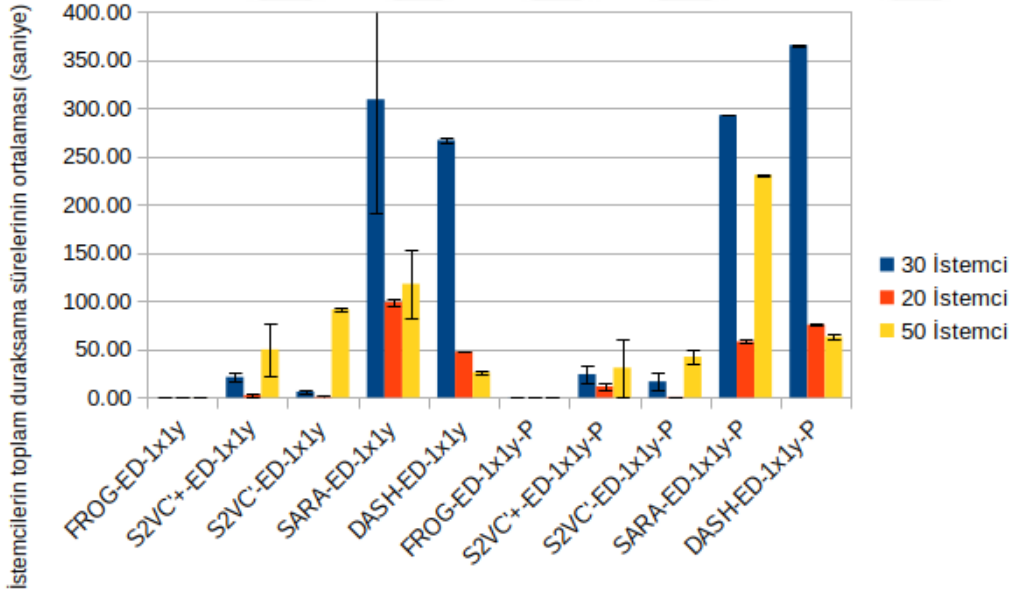
Şekil 5.9: 1x1y'nin ED video seti için QoE karşılaştırması

süresinin 50 saniyeye kadar çıktığı görülmektedir. Optimizasyon araçlarının süre aşımaları ve istemcilerin duraksamaları ile ilgili ilişkiler; 50 istemci ile yapılan testler için Şekil 5.13'te, 30 istemci ile yapılan testler için Şekil 5.14'te ve 20 istemci ile yapılan testler için Şekil 5.15'te görülmektedir. Şekil 5.13 ve Şekil 5.14'te bu ilişkinin doğrusala yakın olduğu, ancak Şekil 5.15'te $S^2VC'+$ -BBB-1x1y-P ve $S^2VC'+$ -ED-1x1y-P testlerinde bu doğrusallıktan sapmalar olduğu görülmektedir. Bu durum bize istemcilerde yaşanan duraksamaların sadece optimizasyon süresinin zaman aşımaları kaynaklı olmadığını, etkin bir şekilde hat kapasitesi kullanımının dez avantajlı olabileceğini ve *FROG*'da duraksamalara karşı geliştirdiğimiz tedbirlerin ne kadar etkin olduğunu göstermektedir.

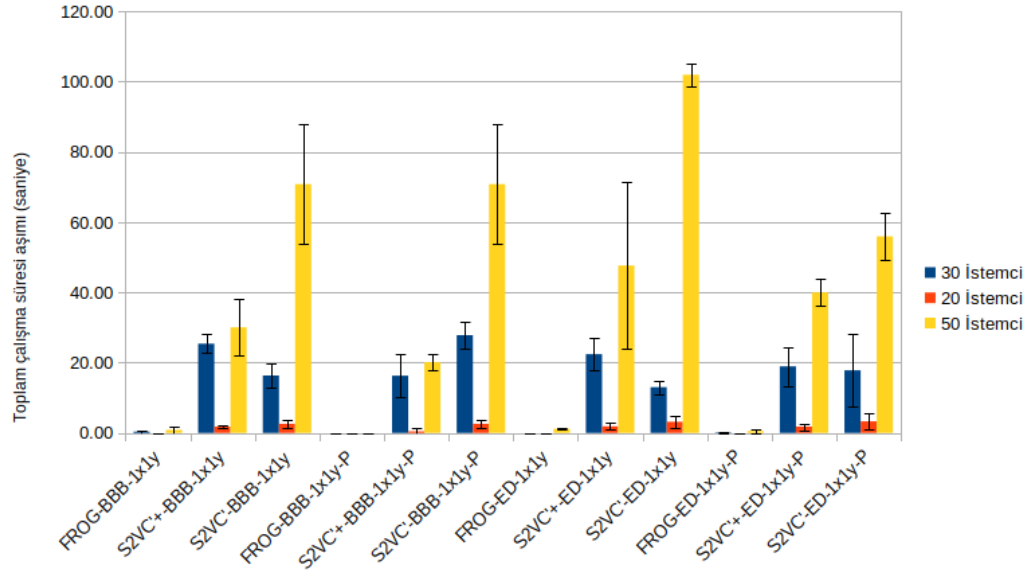
Testlerde elde edilen, istemcilerin ortalama video kalite değerlerinin ortalaması Şekil 5.16 ve Şekil 5.17'de gösterilmektedir. Bu test sonuçlarında *FROG* ve $S^2VC'+$ 'ın birbirine çok yakın değerler elde ettiği hatta 30 istemciyle yapılan testlerde $S^2VC'+$ 'ın *FROG*'dan biraz daha iyi performans gösterdiği görülmektedir. Her iki optimizasyon aracı da aynı şekilde hat kapasitesini etkin bir şekilde kullanırken ve her ikisi de benzer optimizasyon kriterleri kullanırken bu durum nasıl oluşabilir sorusu akla gelmektedir. Bunun nedeni yine duraksa-



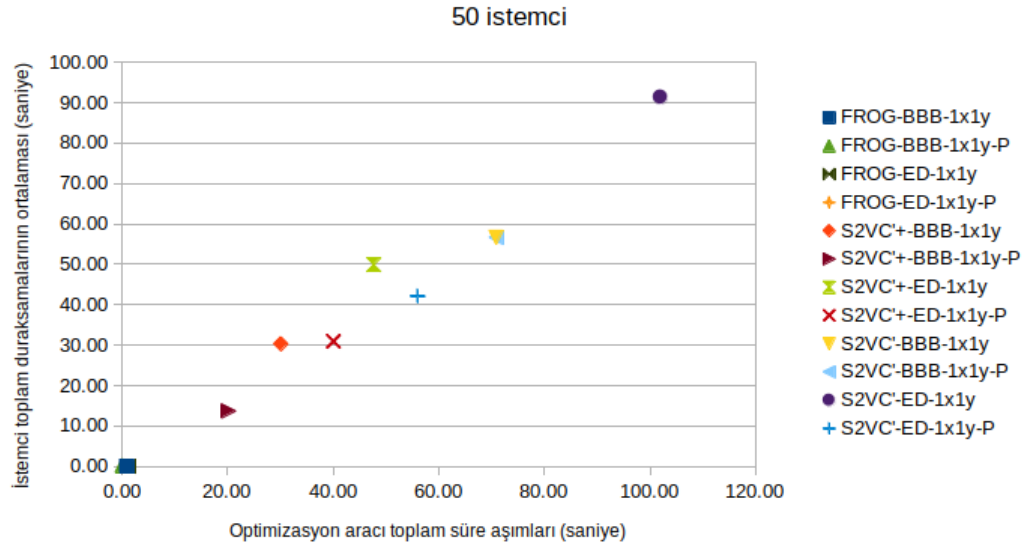
Şekil 5.10: 1x1y yapılandırmasında BBB video seti ile yapılan testte istemcilerin toplam duraksama sürelerinin ortalaması



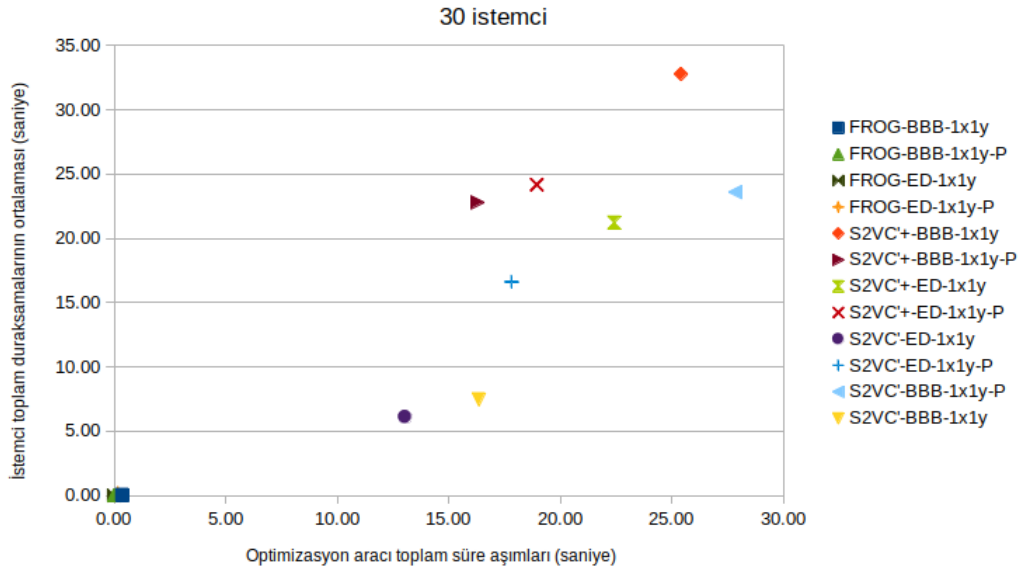
Şekil 5.11: 1x1y yapılandırmasında ED video seti ile yapılan testte istemcilerin toplam duraksama sürelerinin ortalaması



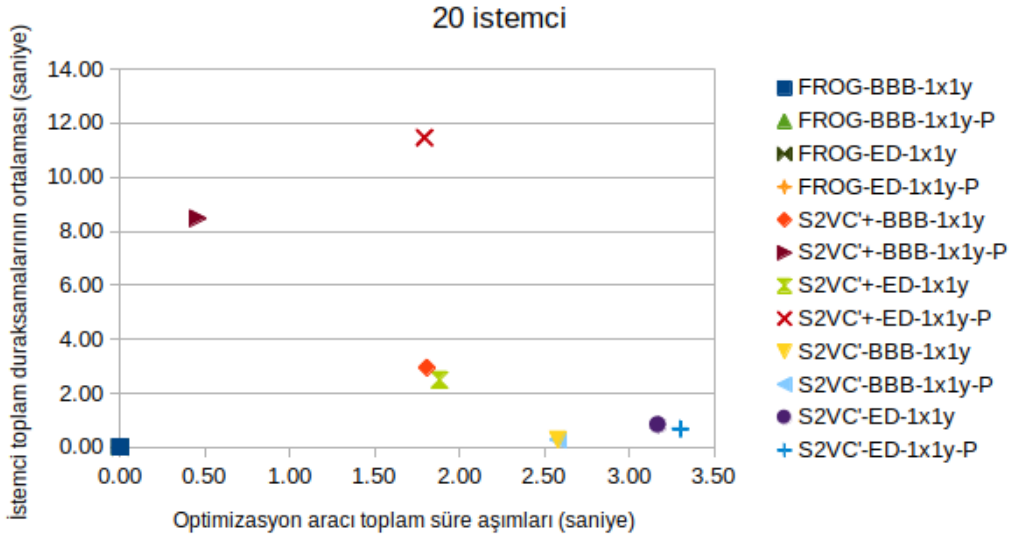
Şekil 5.12: Optimizasyon araçlarının çalışma süresi aşımalarının toplamı



Şekil 5.13: 50 istemci ile yapılan testlerde optimizasyon aracı süre aşımı ve istemci duraksama ilişkisi



Şekil 5.14: 30 istemci ile yapılan testlerde optimizasyon aracı süre aşımı ve istemci duraksama ilişkisi

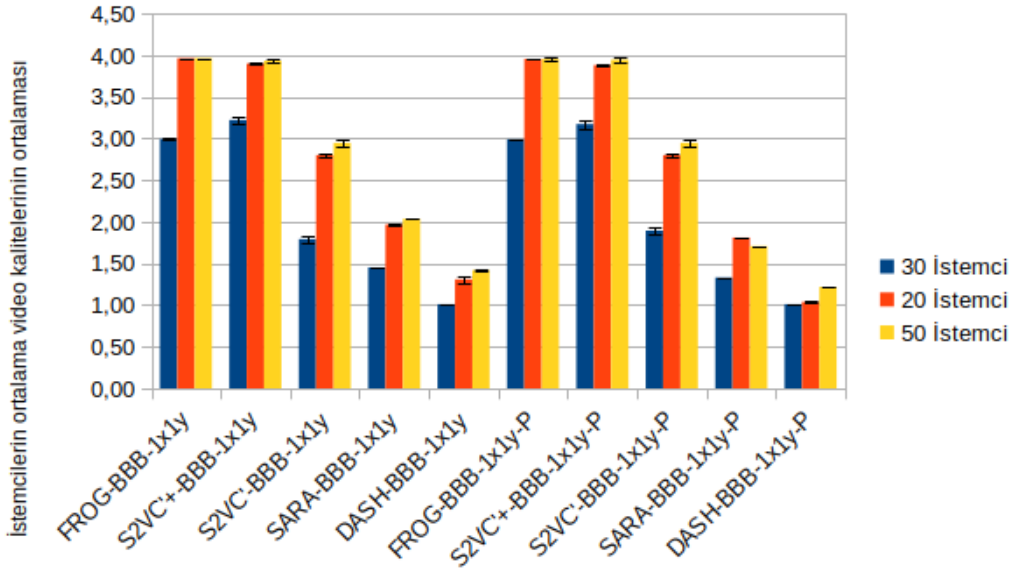


Şekil 5.15: 20 istemci ile yapılan testlerde optimizasyon aracı süre aşımı ve istemci duraksama ilişkisi

Tablo 5.2: $FROG$ ve $S^2VC'+$ 'nın QoE Performans Karşılaştırması

	$FROG$ QoE Değeri	$S^2VC'+$ QoE Değeri	Üstünlük (%)
BBB-1x1y			
30 İstemci	25.57	16.76	52.58
20 İstemci	38.48	36.01	6.85
50 İstemci	38.72	30.49	26.98
BBB-1x1y-P			
30 İstemci	25.91	18.24	42.05
20 İstemci	38.17	34.33	11.17
50 İstemci	38.32	34.82	10.04
ED-1x1y			
30 İstemci	22.09	12.87	71.69
20 İstemci	35.16	28.54	23.19
50 İstemci	36.19	23.73	52.47
ED-1x1y-P			
30 İstemci	22.08	15.48	42.64
20 İstemci	34.66	29.78	16.38
50 İstemci	36.07	28.98	24.46

malardır. QoE sonuçlarında duraksamaların $FROG$ lehine daha iyi bir sonuç oluşturduğundan bahsedilmişti ama video kalitesinde ise S^2VC' 'nin lehine daha iyi sonuçlar çıkmasına neden olabildiği görülmektedir. Bu bir çelişki gibi görülse de aslında olan duraksamalar nedeniyle istemcilerin duraksama sürecinde ağı kullanamaması nedeniyle ONOS'un ağ kapasitesini normalde olması gerekenden daha müsait görmesinden kaynaklanmaktadır. $S^2VC'+$ 'ın optimizasyon aracı tarafından bir sonraki zaman diliminde algılanan kullanılabilir bant genişliği normalden daha yüksek görünmektedir. Bu durum, özellikle bant genişliğinin daha kritik olduğu ağlarda daha fazla önem taşımaktadır. Buna karşılık, bu durum yeterli bant genişliğine sahip ağlarda meydana gelmemektedir. Başka bir deyişle aynı dosyanın indirilmesi $S^2VC'+$ optimizasyon

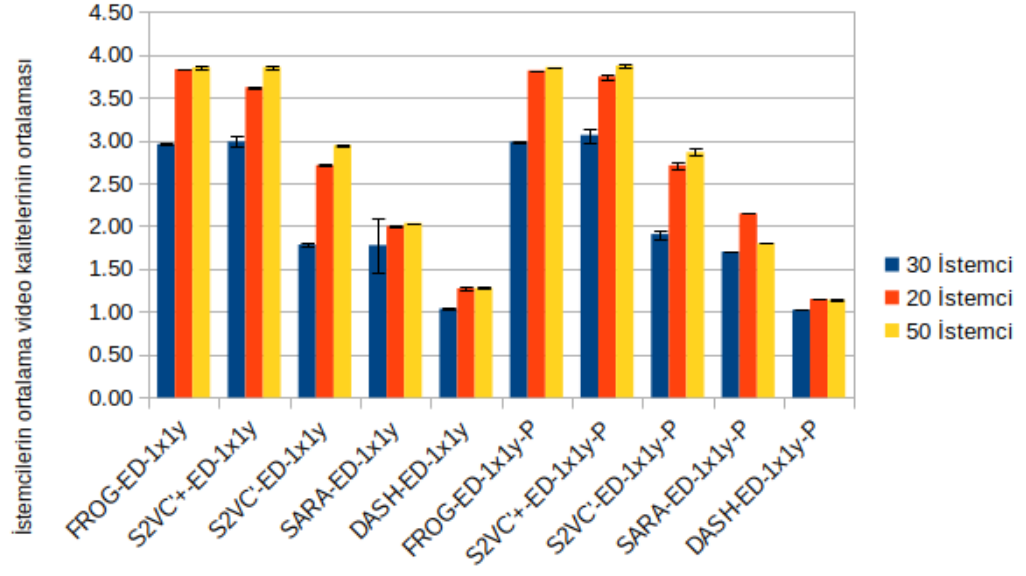


Şekil 5.16: 1x1y’de BBB SVC seti için istemcilerin ortalama video kalitelerinin ortalaması

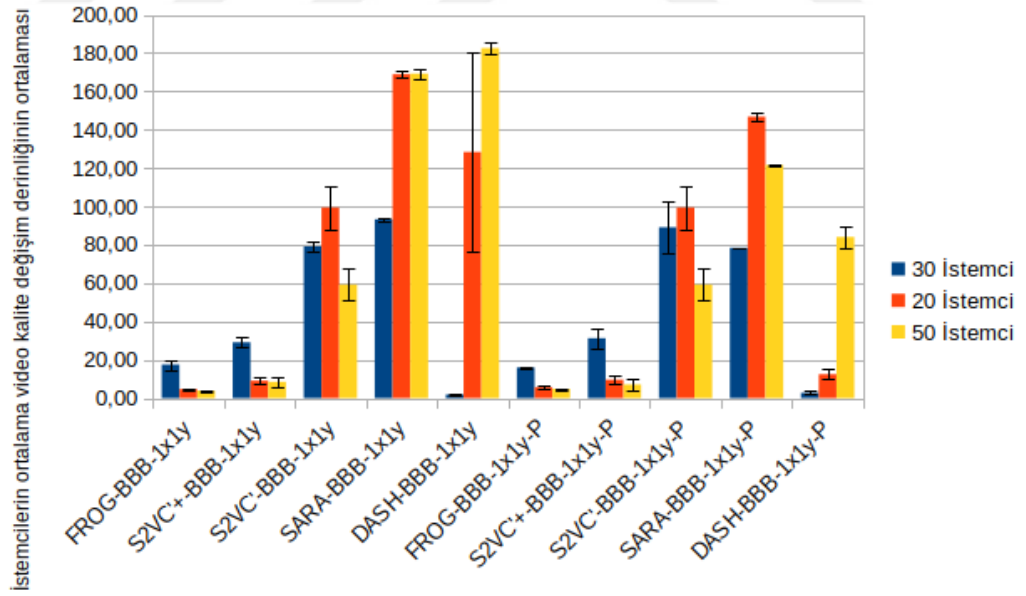
aracının kullanıldığı ortamda daha uzun dağıtım süresine sahip olunması gibi değerlendirilebilir. Örneğin: *FROG*’un kullanıldığı ağlarda istemcilerde duraksama olmadığı için 10 Mbps boyutundaki bir dosyanın istemcilere 2 saniyede dağıtılması gerekirken, *S²VC’+’*’da aynı dosyanın dağıtımı için sahip olunan süresinin 3 saniye olması gibi sahte bir durum ortaya çıkmaktadır. Bu sahte iyi durumun gerçekten iyi bir durum olmadığı da QoE sonuçlarında rahatlıkla görülebilmektedir.

Kullanıcı deneyim kalitesini etkileyen bir diğer kriter olan video segmentleri arasındaki kalite değişimleridir. Şekil 5.18 ve Şekil 5.19’da istemcilerin video kalite değişim derinliklerinin toplamalarının ortalamaları verilmektedir.

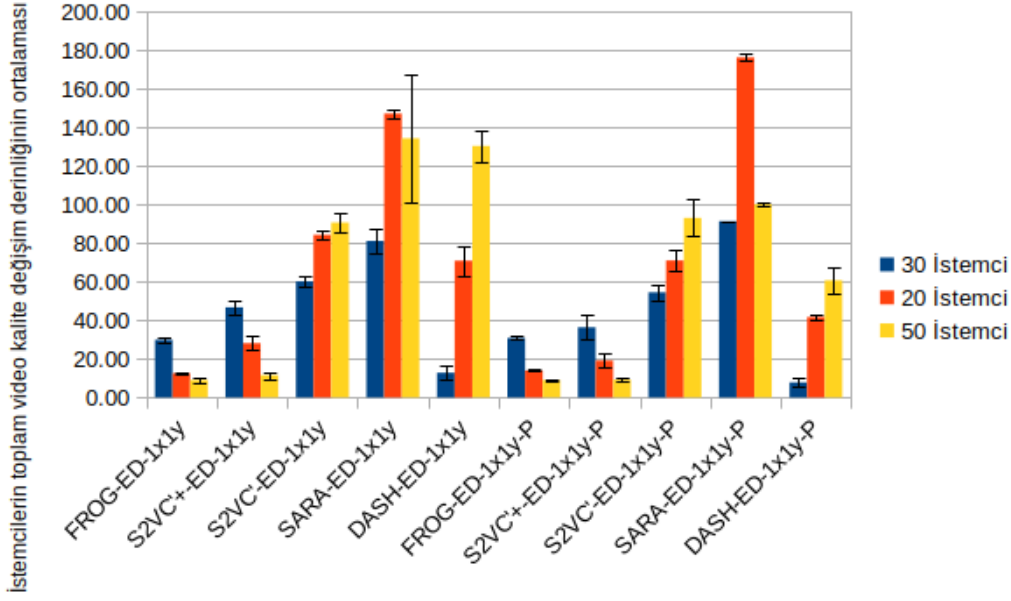
Kalite değişimi doğrudan optimizasyon yöntemiyle ilgili olsa da, etkin hat kapasitesi kullanımı ve tampon bellek yönetimi ile de yakından ilişkilidir. Kalite değişim değerlerimizin diğerlerine göre daha iyi olmasının başlıca sebepleri, etkin hat kapasitesi kullanımı ve tampon bellek yönetimidir. Bir istisna olarak, 30 istemcili testlerde *DASH* istemcisinin performansı dikkat çekmektedir; ancak burada genelde yalnızca temel katmanın indirilmiş olması, kalite değişiminin olmamasının tesadüfi bir durum olduğunu göstermektedir.



Şekil 5.17: 1x1y’de ED SVC seti için istemcilerin ortalama video kalitelerinin ortalaması

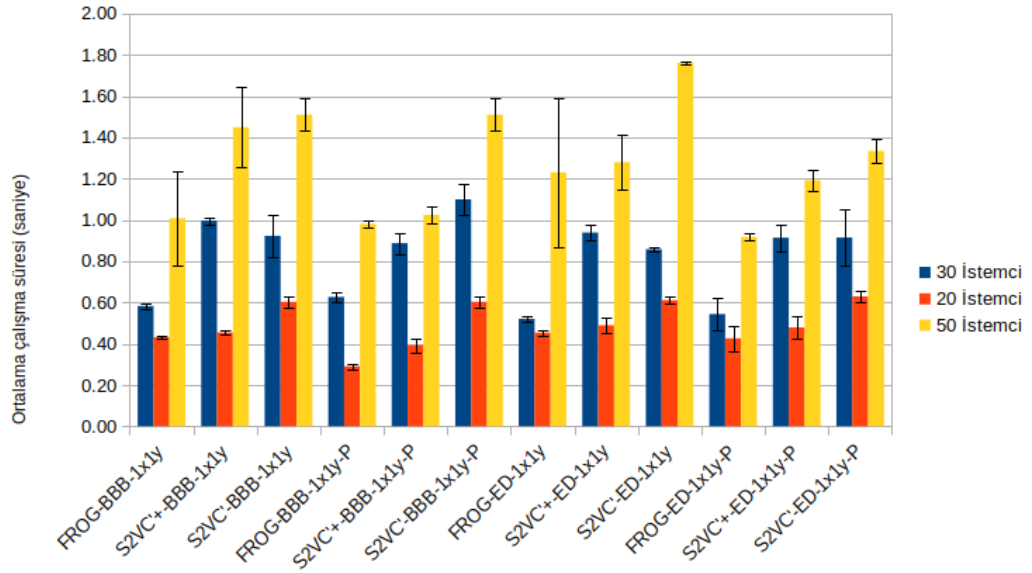


Şekil 5.18: 1x1y’de BBB SVC seti için istemcilerin toplam video kalite değişim derinliğinin ortalaması



Şekil 5.19: 1x1y'de ED SVC seti için istemcilerin toplam video kalite değişim derinliğinin ortalaması

Optimizasyon araçlarının ortalama çalışma sürelerinin grafiği 5.20'de, *FROG*'un en yakın takipçisi olan *S²VC'+* ile karşılaştırması ise Tablo 5.3'te verilmiştir. Ortalama çalışma sürelerinde de *FROG*'un diğerlerine göre daha iyi durumda olduğu görülmektedir. İstemci başına en az ağ kaynağının düştüğü 30 istemci ile yapılan ED-1x1y testinde *FROG*'un en yakın rakibi olan *S²VC'+*'dan %44.54 daha iyi performans sergilediği Tablo 5.3'te görülmektedir.

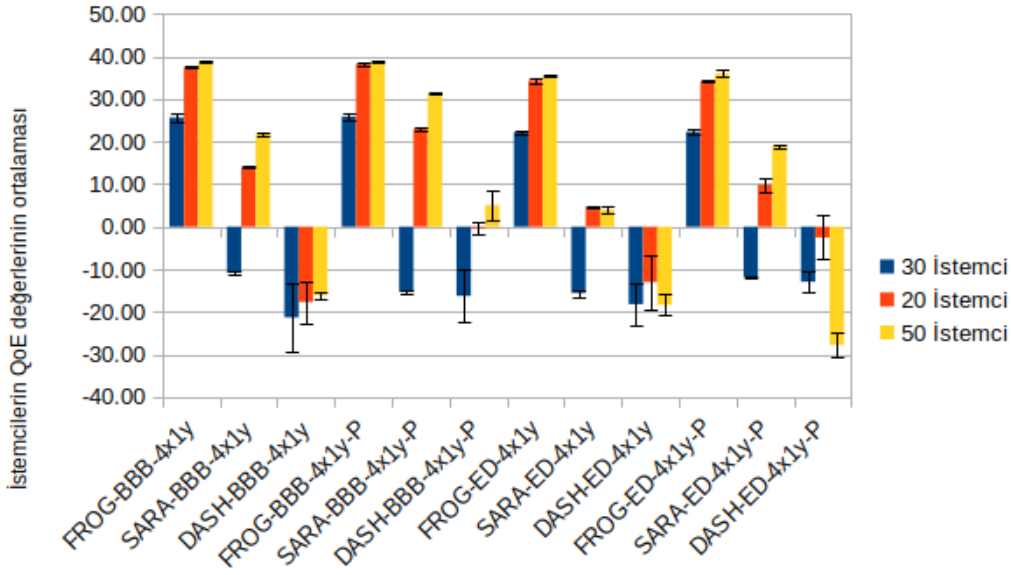


Şekil 5.20: Optimizasyon araçlarının ortalama çalışma süreleri

Tablo 5.3: FROG ve S2VC'+ Çalışma Sürelerinin Karşılaştırması

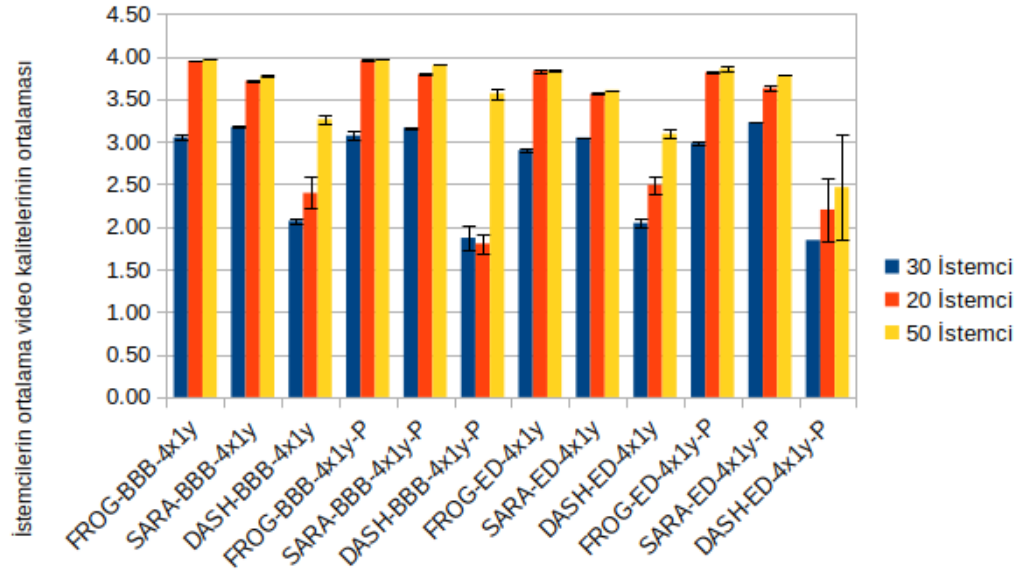
	<i>FROG</i> Çalışma Süresi (s)	<i>S²VC'</i> Çalışma Süresi (s)	Üstünlük (%)
BBB-1x1y			
30 İstemci	0.58	1.00	41.53
20 İstemci	0.43	0.45	5.11
50 İstemci	1.01	1.45	30.43
BBB-1x1y-P			
30 İstemci	0.63	0.89	29.38
20 İstemci	0.29	0.39	25.82
50 İstemci	0.98	1.02	4.16
ED-1x1y			
30 İstemci	0.52	0.94	44.54
20 İstemci	0.45	0.49	7.34
50 İstemci	1.23	1.28	3.88
ED-1x1y-P			
30 İstemci	0.54	0.91	40.49
20 İstemci	0.43	0.48	10.99
50 İstemci	0.92	1.19	22.95

Şu ana kadarki karşılaştırmalarda bir kullanıcı tarafı ve bir istemci

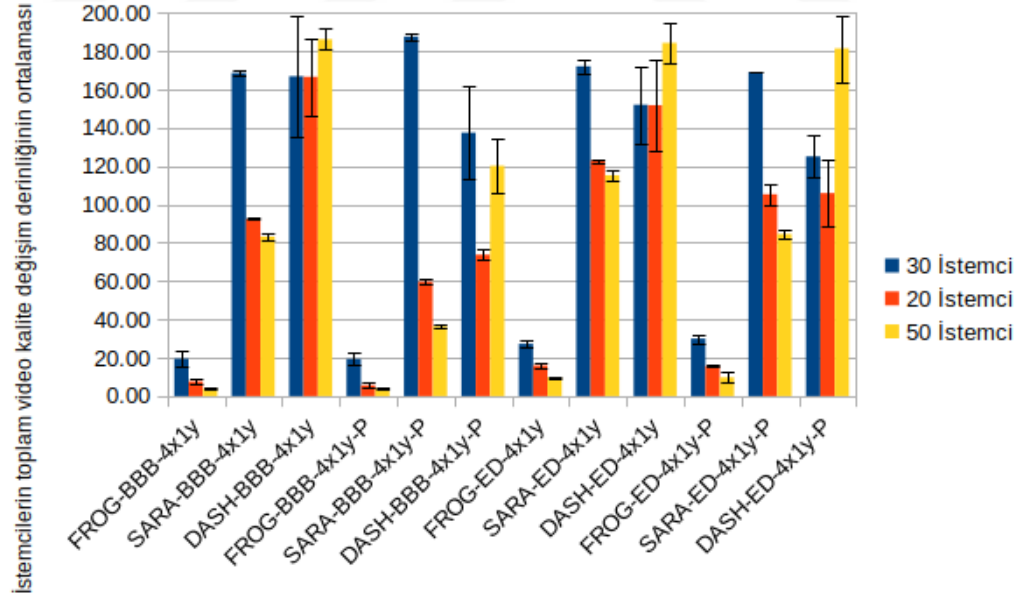


Şekil 5.21: 4x1y’de istemcilerin QoE değerinin ortalaması

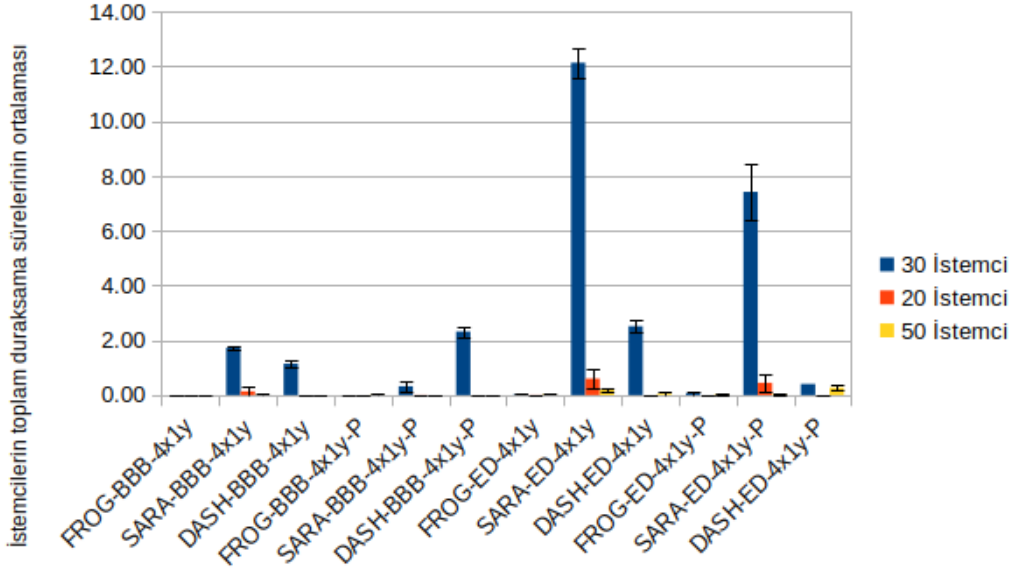
tarafı OF anahtarlama cihazı kullanılmıştır. Buradan itibaren dört kullanıcı tarafı ve bir istemci tarafı OF anahtarlama cihazıyla test ve karşılaştırma değerlendirmeleri yapılacaktır. S^2VC' ve $S^2VC'+$ ’ın birden fazla sunucu desteği olmadığı için bu testlerde *SARA*, *DASH* ve *FROG* kullanılmıştır. Dört sunucu tarafı anahtarlama cihazına birer sunucu bağlanmıştır. *SARA* ve *DASH* istemcileri, hizmet aldıkları dört farklı sunucu tarafı anahtarlama cihazları arasında eşit olarak dağıtılmıştır. Örneğin; 20 istemci ile yapılan testlerde, her birine $(x_1, x_2, x_3$ ve $x_4 \forall x_i \in X)$ 5’er istemci atanmıştır. Şekil 5.21’de görüldüğü üzere, *FROG* tüm testlerde QoE değeri açısından lider konumdadır. 30 istemci ile yapılan testlerde *SARA*, video kalitesi bakımından (Şekil 5.22) *FROG*’dan biraz daha iyi performans gösterse de kalite değişim değeri (Şekil 5.23) oldukça kötü ve duraksama değeri *FROG*’dan daha kötü durumdadır (5.24). *DASH* istemcisi ise QoE değeri ve video kalitesi açısından tüm testlerde en kötü durumdadır. Bu testler ile, istemciler eşit olarak paylaştırılsa bile ağ desteği olmadan başarı oranının sınırlı olarak artabileceği sonucu çıkarılmaktadır.



Şekil 5.22: 4x1y’de istemcilerin ortalama video kalitelerinin ortalaması



Şekil 5.23: 4x1y’de istemcilerin toplam video kalite değişim derinliğinin ortalaması



Şekil 5.24: 4x1y’de istemcilerin toplam duraksama sürelerinin ortalaması

6 SONUÇ

Bu çalışma ile uyarlamalı video dağıtım deneyim kalitesinin iyileştirilmesi adil bir şekilde sağlanmıştır. Sunduğumuz uyarlamalı medya dağıtım çerçevesi, istemci başına düşen ağ kaynaklarının en az olduğu durumda, en yakın rakibine karşı %71.69 daha iyi performans göstermiştir. Bununla birlikte neredeyse hiç duraksama yaşanmamıştır. Çalışmamız kapsamında temel olarak Yazılım Tanımlı Ağ teknolojisi ve matematiksel modelleme yöntemleri kullanılarak optimizasyon modelleri geliştirilmiştir. Bu modellerin ilki olan OPM ile uçtan uca kullanılabilir ağ kapasitesi tespit edilmiştir. Bu veri sayesinde Bender’in Ayrıştırma Yöntemi ile Γ Sağlam Optimizasyon Yöntemi’nden faydalanılarak ve geliştirdiğimiz algoritmayla yapılan gevşetmeyle, NP-Tam olarak tanımlanan optimizasyon probleminin optimum seviyeye yakın ve hızlı bir şekilde çözümü sağlanmıştır. Yine uçtan uca kullanılabilir ağ kapasitesi verisi ve istemcilerden CMCD’ye benzer şekilde alınan tampon bellek doluluk verisiyle tüm istemcilerin tampon bellek yönetimi organize bir şekilde yapılabilmektedir. Böylelikle, hem hangi istemcinin/istemcilerin tampon bellek arttırımı yapabilecekleri belirlenmekte hem de hat kapasitesinin yetersiz olduğu durumlarda hangi istemcinin/istemcilerin tampon belleğinin feda

edileceğinin (hangi istemcinin ilgili zaman diliminde bekletileceği) belirlenebilmektedir. Ağ kapasitesinin etkin kullanımı için kullandığımız yöntem ile hem video kalitesinin önemli ölçüde arttırılması hem de video kalite değişiminin azalması sağlanmıştır. Akış atamalarının etkin bir şekilde yapılması ile OF cihazlardaki akış ayarlarının beklenmesinde geçen sürenin video dağıtımında kullanılabilmesi sağlanmıştır. Γ Sağlam Optimizasyon Yöntemi sayesinde kaliteden ödün vermeden ve QoS uygulamadan, esnek bir şekilde ağdaki dalgalanmalardan etkilenmeyi azaltan bir yöntem geliştirilmiştir. Ağ kapasitesinin temel kalite seviyesini göndermeye dahi yeterli olmadığı durumlarda kaçınılmaz olan duraksamaları en aza indiren bir optimizasyon modeli geliştirilmiştir. Duraksama yaşayan veya duraksama yaşama ihtimali olan istemcilerin önceliklendirilmesi sağlanmıştır.

İlerleyen çalışmalarımızda, kullandığımız optimizasyon yöntemlerini daha da geliştirerek AVC için de deneyim kalitesi iyileştirme araştırmaları yapılması planlanmaktadır.

Kaynaklar

- MPEG Streaming, 2010, HTTP Streaming of MPEG Media, Retrieved from <https://multimediacommunication.blogspot.com/2010/05/http-streaming-of-mpeg-media.html>, Accessed: 2024-11-11.
- Bentaleb, A., Taani, B., Begen, A. C., Timmerer, C., & Zimmermann, R., 2018, A survey on bitrate adaptation schemes for streaming media over HTTP, *IEEE Communications Surveys & Tutorials*, 21(1):562-585 pp. doi: 10.1109/COMST.2018.2868249.
- Kreutz, D., Ramos, F. M. V., Verissimo, P. E., Rothenberg, C. E., Azodolmolky, S., & Uhlig, S., 2014, Software-defined networking: A comprehensive survey, *Proceedings of the IEEE*, 103(1):14–76. IEEE.
- Consumer Technology Association, 2020, CTA-5004: Final Standard, retrieved from <https://cdn.cta.tech/cta/media/media/resources/standards/pdfs/cta-5004-final.pdf>, [Online]. Available.
- Consumer Technology Association, 2022, CTA-5006 Common Media Server Data (CMSD), retrieved from <https://cdn.cta.tech/cta/media/media/resources/standards/pdfs/cta-5006-final.pdf>, [Online]. Available.
- ON.Lab, 2024, ONOS: Open Network Operating System, Retrieved from <https://opennetworking.org/onos/>. [Online]. Available.
- Mininet Team, 2024, Mininet: An Instant Virtual Network on your Laptop (or other PC), Retrieved from <http://mininet.org>. [Online]. Available.
- IBM, 2024, IBM ILOG CPLEX Optimization Studio, Retrieved from <https://www.ibm.com/products/ilog-cplex-optimization-studio>. Accessed: 2024-11-11.
- Schwarz, H., Marpe, D., & Wiegand, T., 2007, Overview of the scalable video coding extension of the H. 264/AVC standard, *IEEE Transactions on Circuits and Systems for Video Technology*, 17(9), 1103–1120. IEEE. doi: 10.1109/TCSVT.2007.904289

- Wiegand, T., Sullivan, G. J., Bjøntegaard, G., & Luthra, A., 2003, Overview of the H.264/AVC video coding standard, *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7), 560–576. IEEE. doi: 10.1109/TCSVT.2003.815165
- Juluri, P., Tamarapalli, V., & Medhi, D., 2015, SARA: Segment aware rate adaptation algorithm for dynamic adaptive streaming over HTTP, In *2015 IEEE International Conference on Communication Workshop (ICCW)* (pp. 1765–1770). IEEE.
- Wikipedia Contributors, 2024, Ethernet frame, *Wikipedia*, Retrieved November 12, 2024, from https://en.wikipedia.org/wiki/Ethernet_frame
- IETF, 1981, RFC 791: Internet Protocol, *Internet Engineering Task Force*, Available at: <https://tools.ietf.org/html/rfc791>.
- IETF, 1981, RFC 793: Transmission Control Protocol, *Internet Engineering Task Force*, Available at: <https://tools.ietf.org/html/rfc793>.
- Open Networking Foundation, 2011, Software-Defined Networking: The New Norm for Networks, White Paper, Available at: <https://opennetworking.org/wp-content/uploads/2011/09/wp-sdn-newnorm.pdf>.
- ONOS Project, n.d., ONOS Wiki, Retrieved from <https://wiki.onosproject.org/>.
- IBM, (n.d.), IBM ILOG CPLEX Optimization Studio, Available at: <https://www.ibm.com/products/ilog-cplex-optimization-studio/cplex-optimizer>, Accessed: 2024-11-12.
- IBM, (n.d.), IBM ILOG CPLEX User Manual, Available at: <https://www.ibm.com/docs/en/icos/20.1.0?topic=cplex-users-manual>, Accessed: 2024-11-16.
- itec.aau.at, 2015, BBB SVC DASH Dataset, Available at: <http://ftp.itec.aau.at/datasets/SVCDASHDataset2015/BBB/I/segs/1080p/>, Accessed: 2023-10-23.

- itec.aau.at, 2015, ED SVC DASH Dataset, Available at: <http://ftp.itec.aau.at/datasets/SVCDASHDataset2015/ED/I/segs/1080p/>, Accessed: 2023-10-23.
- Naoum-Sawaya, J., & Elhedhli, S., 2010, A nested Benders decomposition approach for telecommunication network planning, *Naval Research Logistics (NRL)*, 57(6), 519–539. Wiley Online Library.
- Bertsimas, D., & Sim, M., 2003, Robust discrete optimization and network flows, *Mathematical Programming*, 98(1), 49–71. Springer.
- Tashtarian, F., Erfanian, A., & Varasteh, A., 2018, S2VC: An SDN-based framework for maximizing QoE in SVC-based HTTP adaptive streaming, *Computer Networks*, 146, 33–46. Elsevier.
- Erfanian, A., Tashtarian, F., & Yaghmaee, M. H., 2018, On maximizing QoE in AVC-based HTTP adaptive streaming: An SDN approach, *2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS)*, 1–10. IEEE.
- Kalan, R., & Dulger, I., 2024, A survey on QoE management schemes for HTTP adaptive video streaming: Challenges, solutions, and opportunities, *IEEE Access*.
- Bentaleb, A., Begen, A. C., & Zimmermann, R., 2016, SDNDASH: Improving QoE of HTTP adaptive streaming using software defined networking, In *Proceedings of the 24th ACM International Conference on Multimedia*, 1296–1305.
- Bentaleb, A., Begen, A. C., Zimmermann, R., & Harous, S., 2017, SDNHAS: An SDN-enabled architecture to optimize QoE in HTTP adaptive streaming, *IEEE Transactions on Multimedia*, 19(10), 2136–2151.
- Begen, A. C., 2021, Manus manum lavat: Media clients and servers cooperating with common media client/server data, In *Proceedings of the Applied Networking Research Workshop*, 82–84.

- Sodagar, I., 2011, The MPEG-DASH standard for multimedia streaming over the internet, *IEEE Multimedia*, 18(4), 62–67. IEEE.
- Akhshabi, S., Anantakrishnan, L., Begen, A. C., & Dovrolis, C., 2012, What happens when HTTP adaptive streaming players compete for bandwidth, In *Proceedings of the 22nd ACM NOSSDAV Workshop*, 9–14.
- Huang, T.-Y., Handigol, N., Heller, B., McKeown, N., & Johari, R., 2012, Confused, timid, and unstable: Picking a video streaming rate is hard, In *Proceedings of the Internet Measurement Conference*, 225–238.
- Kalan, R. S., & Sayit, M., 2021, SDN assisted codec, path and quality selection for HTTP adaptive streaming, *IEEE Access*, 9, 129917–129932. IEEE. doi: 10.1109/ACCESS.2021.3112815.
- Kleinrouweler, J. W., Cabrero, S., & Cesar, P., 2016, Delivering stable high-quality video: An SDN architecture with DASH assisting network elements, In *Proceedings of the 7th International Conference on Multimedia Systems*, 1–10.
- Kalan, R. S., 2021, Improving quality of HTTP adaptive streaming with server and network-assisted DASH, In *Proceedings of the 2021 17th International Conference on Network and Service Management (CNSM)*, 244–248. IEEE.
- Thomas, E., van Deventer, M. O., Stockhammer, T., Begen, A. C., Champel, M-L., & Oyman, O., 2016, Applications and deployments of server and network assisted DASH (SAND), *IET*.
- Lu, Z., Ramakrishnan, S., & Zhu, X., 2017, Exploiting video quality information with lightweight network coordination for HTTP-based adaptive video streaming, *IEEE Transactions on Multimedia*, 20(7), 1848–1863.
- Bhat, D., Rizk, A., Zink, M., & Steinmetz, R., 2018, SABR: Network-assisted content distribution for QoE-driven ABR video streaming, *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 14(2s), 1–25.

- Gohar, A., Raza, A., & Lee, S., 2019, Dynamic adaptive streaming over HTTP using scalable video coding with multipath TCP in SDN, *2019 International Conference on Information Networking (ICOIN)*, 480–484.
- Gohar, A., & Lee, S., 2020, Multipath dynamic adaptive streaming over HTTP using scalable video coding in software defined networking, *Applied Sciences*, 10(21), 7691.
- Ozcelik, I. M., & Ersoy, C., 2019, Chunk Duration-Aware SDN-Assisted DASH, *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 15(3), 1–22.
- Farahani, R., Tashtarian, F., Amirpour, H., Timmerer, C., Ghanbari, M., & Hellwagner, H., 2021, ES-HAS: An Edge- and SDN-Assisted Framework for HTTP Adaptive Video Streaming, In *Proceedings of the 2021 IEEE 46th Conference on Local Computer Networks (LCN)*, 581–588. IEEE.
- Farahani, R., Tashtarian, F., Amirpour, H., Timmerer, C., Ghanbari, M., & Hellwagner, H., 2021, CSDN: CDN-aware QoE optimization in SDN-assisted HTTP adaptive video streaming, In *Proceedings of the 2021 IEEE 46th Conference on Local Computer Networks (LCN)*, 525–532. IEEE.
- Farahani, R., Tashtarian, F., Timmerer, C., Ghanbar, M., & Hellwagner, H., 2022, LEADER: A collaborative edge- and SDN-assisted framework for HTTP adaptive video streaming, In *Proceedings of the ICC 2022–IEEE International Conference on Communications*, 745–750. IEEE.
- Farahani, R., Shojafar, M., Timmerer, C., Tashtarian, F., Ghanbari, M., & Hellwagner, H., 2023, ARARAT: A Collaborative Edge-Assisted Framework for HTTP Adaptive Video Streaming, *IEEE Transactions on Network and Service Management*, 20(1), 625–643. <https://doi.org/10.1109/TNSM.2022.3210595>.
- Tashtarian, F., & Timmerer, C., 2024, REVISION: A roadmap on adaptive video streaming optimization, *arXiv preprint arXiv:2409.06051*.

- Farahani, R., Bentaleb, A., Timmerer, C., Shojafar, M., Prodan, R., & Hellwagner, H., 2023, SARENA: SFC-enabled architecture for adaptive video streaming applications, *ICC 2023 - IEEE International Conference on Communications*, 864–870.
- Yeznabad, Y. F., Helfert, M., & Muntean, G.-M., 2024, QoE-Driven Cross-Layer Bitrate Allocation Approach for MEC-Supported Adaptive Video Streaming, *IEEE Transactions on Network and Service Management*.
- Kırmızıoğlu, R. A., Tekalp, A. M., and Görkemli, B., 2025, Distributed virtual selective-forwarding units and SDN-assisted edge computing for optimization of multi-party WebRTC videoconferencing, *Signal Processing: Image Communication*, 130, 117173.
- Bentaleb, A., Lim, M., Akcay, M. N., Begen, A. C., & Zimmermann, R., 2021, Common media client data (cmcd) initial findings, *Proceedings of the 31st ACM Workshop on Network and Operating Systems Support for Digital Audio and Video*, 25–33.
- Nguyen, M., Taraghi, B., Bentaleb, A., Zimmermann, R., & Timmerer, C., 2022, CADLAD: Device-aware bitrate ladder construction for HTTP adaptive streaming, *2022 18th International Conference on Network and Service Management (CNSM)*, 198–204. IEEE.
- Pham, S., Law, W., Begen, A. C., Silhavy, D., Berthelot, B., Arbanowski, S., & Steglich, S., 2023, Common Media Server Data (CMSD)–Update on Implementations and Validation of Key Use Cases, *Proceedings of the 2nd Mile-High Video Conference*, 137–138.
- Bentaleb, A., Lim, M., Akcay, M. N., Begen, A. C., & Zimmermann, R., 2024, Bitrate Adaptation and Guidance With Meta Reinforcement Learning, *IEEE Transactions on Mobile Computing*, 23(11), 10378–10392. <https://doi.org/10.1109/TMC.2024.3376560>.

Tran, T., Gageot, D., Neumann, C., Bichot, G., Tlili, A., Boutiba, K., & Ksentini, A., 2024, On the benefits and caveats of exploiting Quality on Demand Network APIs for video streaming, *Proceedings of the 34th Edition of the Workshop on Network and Operating System Support for Digital Audio and Video*, 43–49. <https://doi.org/10.1145/3651863.3651882>.

Global Internet Performance Report, 2024, Internet Traffic Report GIPR 2024, Retrieved November 24, 2024, from <https://www.sandvine.com/phenomena>



TEŞEKKÜR

Bu doktora tezi süresince bilgi ve görüşleriyle bana yol gösteren, desteğini hiçbir zaman esirgemeyen danışman hocam sayın Doç. Dr. Müge SAYIT'e (Uluslararası Bilgisayar Enstitüsü, Ege Üniversitesi, Türkiye) en içten saygı ve teşekkürlerimi sunarım. Ayrıca tez izleme jüri olarak görev yapan, yüksek lisans eğitimimde ve doktora eğitimimin tez aşamasına kadar olan bölümde danışman hocam olarak bana yol gösteren sayın Prof. Dr. Mehmet Emin DALKILIÇ'a (Uluslararası Bilgisayar Enstitüsü, Ege Üniversitesi, Türkiye) ve tez izleme jüri olarak görev yapan sayın hocam Prof. Dr. Ali Cengiz BEĞEN'e (Bilgisayar Mühendisliği Fakültesi, Özyeğin Üniversitesi, Türkiye) değerli önerileri ve katkıları için en içten saygı ve teşekkürlerimi sunarım.

Sevgili eşim Zehra Hatice TÜRK'e, hep yanımda olduğu, desteğini hiçbir zaman esirgemediği ve bu zorlu süreçte bana gösterdiği sabır için sonsuz teşekkür ederim. Ayrıca, iki çocuğumuzun sevgili annesi olarak, onların mutluluğunu ve huzurunu her zaman ön planda tutarak bu süreçte beni desteklediği için minnettarım.

24/12/2024

İmzası

Mustafa TÜRK

ÖZGEÇMİŞ

Mustafa TÜRK, lisansını 2001 yılında Anadolu Üniversitesi'nde Maliye bölümünde tamamlamış, 2010 yılında Sakarya Üniversitesi'nden Bilgi Teknolojileri alanında Yüksek Lisans, 2016 yılında ise Ege Üniversitesi'nden yine Bilgi Teknolojileri alanında Yüksek Lisans derecesi almıştır. Ayrıca, Ege Üniversitesi'nde Bilgi Teknolojileri alanında doktora adaylığına sahiptir. Araştırma alanları; Bilgisayar Ağları, Optimizasyon, Yazılım Tanımlı Ağlar (SDN) ve Video dağıtımdır. Ek olarak, 20 yılı aşkın bir süre kamu ve özel sektörde ağ ve sistem uzmanı olarak çalışmıştır.

