CORA Cork Open Research Archive
Cartlann Taighde Oscailte Chorcaí

| | |
|---|---|
| Title | User experience and network resource optimization for live video streaming |
| Authors | Khalid, Ahmed |
| Publication date | 2019-10 |
| Original Citation | Khalid, A. 2019. User experience and network resource optimization for live video streaming. PhD Thesis, University College Cork. |
| Type of publication | Doctoral thesis |
| Rights | © 2019, Ahmed Khalid. - https://creativecommons.org/licenses/by-nc-nd/4.0/ |
| Download date | 2026-11-14 02:09:35 |
| Item downloaded from | https://hdl.handle.net/10468/9931 |

UCC
University College Cork, Ireland
Coláiste na hOllscoile Corcaigh

# User Experience and Network Resource Optimization for Live Video Streaming

Ahmed Khalid
MSc
115220403

**Thesis submitted for the degree of Doctor of Philosophy**

NATIONAL UNIVERSITY OF IRELAND, CORK

School of Computer Science And Information Technology

October 2019

Head of School: Prof Cormac J. Sreenan

Supervisors: Prof. Cormac J. Sreenan
Dr. Ahmed H. Zahran

Research supported by SFI

# Contents

List of Figures . . . iv
List of Tables . . . vi
List of Equations . . . vii
Acknowledgements . . . x
List of Abbreviations . . . xi
Abstract . . . xv

**1 Introduction** **1**
1.1 Motivation . . . 1
1.1.1 Core and wired access networks . . . 2
1.1.2 Cellular networks . . . 3
1.2 Thesis statement and contributions . . . 4
1.2.1 Thesis statement . . . 4
1.2.2 Key contributions . . . 5
1.3 Thesis organization . . . 7
1.4 List of publications . . . 7

**2 Background and literature review** **9**
2.1 Video streaming standards and protocols . . . 9
2.2 Video transmission modes . . . 10
2.2.1 IP unicast . . . 11
2.2.2 IP multicast . . . 12
2.2.3 Application-layer multicast . . . 13
2.3 Multicast in programmable networks . . . 14
2.3.1 SDN and its benefits . . . 14
2.3.2 IP multicast in SDN . . . 15
2.3.3 SDN-based network-layer multicast . . . 16
2.4 Multicast in cellular networks . . . 17
2.4.1 Background on eMBMS . . . 17
2.4.2 Related work on eMBMS . . . 18
2.4.3 eMBMS configurations . . . 19
2.5 Video performance metrics and indicators . . . 22
2.5.1 Quality of service . . . 22
2.5.2 Quality of experience . . . 24
2.5.3 Resource consumption . . . 27

**3 Tools and methodologies** **30**
3.1 Simulation vs emulation . . . 30
3.2 MiniNAM: A network animator for visualizing real-time packet flows in Mininet . . . 31
3.2.1 Introduction . . . 32
3.2.2 Design overview . . . 33
3.2.3 Applications of MiniNAM . . . 35
3.2.4 Demonstration overview . . . 35
3.3 eSMAL: Evaluating SDN-based live streaming . . . 37

3.3.1 Platform overview . . . 38
3.3.2 Demonstration overview . . . 41
3.4 eMBMS simulator and animator . . . 42
3.4.1 Network topology . . . 42
3.4.2 Physical layer parameters . . . 43
3.4.3 Animation GUI . . . 44

**4 mCast: Enabling inter-domain network-layer multicast for live streaming services 47**
4.1 Introduction . . . 47
4.2 Architecture and components . . . 49
4.2.1 Architecture overview . . . 49
4.2.2 Functional description . . . 53
4.2.3 Performance analysis . . . 55
4.3 Cost-based decision model . . . 55
4.4 Performance evaluation and comparison . . . 59
4.4.1 Dropped network packets and video frames . . . 60
4.4.2 Link utilization and bandwidth consumption . . . 61
4.4.3 Start-up delays . . . 62
4.4.4 Overhead cost of mCast . . . 63

**5 Danos: Device-aware network-assisted optimal streaming service 65**
5.1 Introduction . . . 65
5.2 Architecture and components . . . 67
5.2.1 Request classifier . . . 68
5.2.2 Danos CDN agent . . . 68
5.2.3 Danos ISP agent . . . 69
5.2.4 Danos database and bitrate selector . . . 70
5.2.5 Danos routing modules . . . 71
5.2.6 Danos optimization engine . . . 72
5.2.7 Danos flow manager . . . 72
5.2.8 Danos streaming server and client . . . 73
5.2.9 Danos workflow . . . 74
5.3 System design . . . 75
5.3.1 Smooth switching of client bitrates . . . 76
5.3.2 Synchronizing optimization and routing modules . . . 78
5.3.3 ISP transrating services . . . 80
5.4 Global and real time optimization . . . 81
5.4.1 System model . . . 81
5.4.2 Problem formulation . . . 82
5.4.3 Real-time guided optimization . . . 84
5.4.4 Performance analysis . . . 85
5.5 Performance evaluation and comparison . . . 88
5.5.1 Prototype implementation . . . 89
5.5.2 Flash crowd scenario . . . 91
5.5.3 Cross-traffic scenario . . . 92

**6 RTOP: Optimizing SFN clusters and user groups in eMBMS 95**
6.1 Introduction . . . 95
6.2 Joint optimization model . . . 97
6.2.1 System model . . . 97
6.2.2 Problem formulation . . . 99
6.2.3 Time scale of optimization . . . 100
6.3 Real-time heuristics-based algorithm . . . 101
6.3.1 Handling a single video . . . 101
6.3.2 Handling multiple videos . . . 104
6.4 Performance evaluation and comparison . . . 108
6.4.1 Simulation setup . . . 108
6.4.2 Generic scenario: Multiple videos . . . 110
6.4.3 Mega event scenarios . . . 112
6.4.4 Impact of available resources on various metrics . . . 114

**7 NIMBLE: Network optimization for eMBMS live video QoE 117**
7.1 Introduction . . . 117
7.2 Optimal user experience model . . . 119
7.2.1 System model . . . 119
7.2.2 Problem formulation . . . 120
7.2.3 Practical considerations . . . 123
7.3 Lightweight heuristics-based algorithm . . . 125
7.3.1 Candidate SFN layouts and their utility . . . 126
7.3.2 Optimal SFN layout for each video . . . 129
7.3.3 Optimal network configuration . . . 130
7.3.4 Controlling network reconfiguration . . . 131
7.3.5 Complete NIMBLE algorithm . . . 132
7.4 Performance evaluation and comparison . . . 133
7.4.1 Simulation setup . . . 134
7.4.2 Analysis of control parameters . . . 136
7.4.3 Comparison with state-of-the-art algorithms . . . 138

**8 Conclusion and future work 143**
8.1 Summary . . . 143
8.2 Key contributions and findings . . . 144
8.3 Future work . . . 147

# List of Figures


1.1 Twitch viewers statistics . . . 2
1.2 eMBMS architecture . . . 3

2.1 SDN architecture . . . 14
2.2 An example of eMBMS configuration . . . 17
2.3 Example of no eMBMS and standard eMBMS configuration . . . 20
2.3 Example of SFN clustering and user grouping configuration . . . 21
2.3 Example of joint eMBMS configuration . . . 22

3.1 System design of MiniNAM . . . 33
3.2 MiniNAM dialog boxes . . . 35
3.3 MiniNAM demonstration examples . . . 36
3.4 eSMAL setup and components . . . 39
3.5 eSMAL panoramic GUI . . . 41
3.6 eSMAL snapshot of MiniNAM GUI . . . 42
3.7 An example of eMBMS hexagonal grid layout . . . 43
3.8 A snapshot of eMBMS animator GUI for shopping mall . . . 45

4.1 Architecture and components of mCast . . . 49
4.2 Important message exchanges to establish and serve mCast. . . . 53
4.3 Experimental testbed for evaluation of mCast . . . 59
4.4 Link utilization and startup delay of mCast . . . 62
4.5 OpenFlow overhead in mCast . . . 63

5.1 Architecture and components of Danos . . . 68
5.2 Message exchanges in Danos . . . 69
5.3 An example workflow of Danos . . . 74
5.4 An example of smooth bitrate switches in Danos . . . 77
5.5 Flow diagram of Danos optimization framework . . . 79
5.6 Danos computation-time analysis . . . 87
5.7 Correctly handled users by Danos and mCast . . . 88
5.8 Prototype implementation of Danos . . . 89
5.9 Danos vs mCast: PMF of average user-bitrates . . . 91
5.10 Danos vs mCast: handling flash crowds . . . 92
5.11 Danos vs mCast: cross-traffic scenario . . . 93

6.1 RTOP: Example of eMBMS configurations . . . 96
6.2 RTOP Heuristic example . . . 102
6.3 Analyzing optimal number of user groups in eMBMS . . . 103
6.4 RTOP Regression Example . . . 105
6.5 RTOP comparison for generic scenario with multiple videos. . . . 111
6.6 RTOP comparison for mega-event scenario . . . 113
6.7 RTOP comparison for varying RBs available to eMBMS . . . 115

7.1 A flow chart of NIMBLE heuristics . . . 125
7.2 An example of SFN layout classification by NIMBLE . . . 127

7.3 Sample utility vs RB graph for eMBMS clusters . . . . . 129
7.4 Analysis of NIMBLE recalculation interval . . . . . 136
7.5 Analysis of NIMBLE increment threshold . . . . . 138
7.6 Analysis of throughput, switches and stalls in NIMBLE . . . . . 139
7.7 Analysis of QoE and network reconfiguration for NIMBLE . . . 141

# List of Tables

2.1 Qualitative analysis of video transmission modes . . . . . 11

3.1 MiniNAM preferences and filter options . . . . . 34
3.2 eSMAL GUI parameters to modify network and video settings . 38

4.1 Results of mCast experiments for STAR topology . . . . . 60
4.2 Results of mCast experiments for MESH topology . . . . . 61

5.1 Notation for the Danos optimization model . . . . . 82

6.1 Notations For RTOP optimization model . . . . . 98
6.2 RTOP simulation parameters . . . . . 109

7.1 Notations For NIMBLE optimization model . . . . . 119
7.2 NIMBLE simulation parameters . . . . . 134
7.3 Fairness analysis of NIMBLE . . . . . 142

# List of Equations


3.1 RSRP calculation with log-normal shadowing . . . . . 43
3.2 SINR calculation with log-normal shadowing . . . . . 44

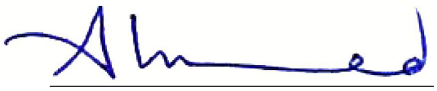
4.1 CDN power consumption to serve N clients . . . . . 56
4.2 Internet exchange point transit cost . . . . . 56
4.3 CDN bandwidth consumption to server N clients . . . . . 57
4.4 Cost of serving N clients with IP unicast . . . . . 57
4.5 Cost of serving N clients with mCast . . . . . 58

5.1 Danos bitrate assignment model . . . . . 83
5.2 Danos network load minimization . . . . . 84
5.2 Danos optimization by user classification . . . . . 85

6.1 RTOP optimization model . . . . . 100
6.2 Regression-based utility maximization in RTOP . . . . . 106

7.1 Utility achieved through bitrates of eMBMS users . . . . . 121
7.2 Total penalty for bitrate switches encountered by eMBMS users 121
7.3 Penalty for bitrate switches of an eMBMS user . . . . . 121
7.4 Throughput of a unicast user . . . . . 122
7.5 NIMBLE optimization model for unicast and eMBMS users . . . 123
7.6 Proportional resource distribution among videos for NIMBLE . 126
7.7 Utility function for an SFN layout . . . . . 129
7.8 Regression-based utility function for an SFN layout . . . . . 130
7.9 Utility maximization for a combination of SFN layouts . . . . . 130

I, Ahmed Khalid, certify that this thesis is my own work and has not been submitted for another degree at University College Cork or elsewhere.



*Ahmed Khalid*

To my family.

# Acknowledgements

First and foremost, I would like to express my sincere gratitude to my supervisors Prof. Cormac J. Sreenan and Dr. Ahmed H. Zahran for their continuous support during my PhD research, for their guidance, enthusiasm and immense knowledge. Their valuable feedback helped me throughout my research and writing of this thesis. I could not have imagined having better supervisors and mentors for my PhD.

Besides my supervisors, I would like to thank my fellow lab-mates Dr. Jason J. Quinlan, Dr. Yusuf Sani, Darijo Raca, Dr. Mustafa Al-Bado, Dr. David Stynes, Dr. Saim Ghafoor and Alexander Revyakin, for their insightful comments and encouragement, and also for the hard questions and the stimulating discussions which incented me to widen my research from various perspectives. I would also like to thank Mary Noonan for her wonderful administrative support and her kindness.

I also acknowledge Science Foundation of Ireland (SFI) for funding my PhD studies through iVID grant.

Last but not least, I would like to thank my family and friends, my parents, brothers and sister, for their motivation and prayers that assisted me throughout my PhD. My lovely wife, who not just encouraged me to work harder but also ensured, in every possible way, that I could spend as much time on my research as needed. Words cannot express how thankful I am for all of the sacrifices that she made on my behalf. My two sons whose presence and love provided stability to my otherwise hectic routine. I could not have achieved what I did, without my family's help and support.

# List of Abbreviations

**3GPP** Third Generation Partnership Project

**ALM** Application-Layer Multicast

**API** Application Programming Interface

**AVC** Advanced Video Coding

**AWGN** Additive White Gaussian Noise

**BLER** Block Error Rate

**CDF** Cumulative Distributive Function

**CDN** Content Delivery Networks

**CQI** Channel Quality Indicator

**Danos** Device-Aware Network-Assisted Optimal Streaming

**DASH** Dynamic Adaptive Streaming over HTTP

**DOCSIS** Data Over Cable Service Interface Specification

**DSL** Digital Subscriber Line

**DTTV** Digital Terrestrial Television

**eMBMS** Evolved Multimedia Broadcast Multicast Service

**eNB** Evolved-Node Base Stations

**EPC** Evolved Packet Core

**FEC** Forward Error Correction

**fps** frames per second

**Gbps** Gigabits per second

**GOP** Group of Picture

**GUI** Graphical User Interface

**HD** High Definition

**HTTP** Hyper-Text Transfer Protocol

**IGMP** Internet Group Management Protocol

**ILP** Integer Linear Program

**IP** Internet Protocol

**ISP** Internet Service Providers

**IXP** Internet Exchange Point

**JSVM** Joint Scalable Video Model

**Kb** Kilo-bits

**Kbps** Kilo-bits per second

**LTE** Long Term Evolution

**Mb** Megabits

**Mbps** Megabits per second

**MCCH** Multicast Control Channel

**MCE** Multicast Coordination Entity

**MCS** Modulation and Coding Scheme

**MDC** Multiple Description Coding

**MiniNAM** Mininet Network Animation Tool

**MME** Mobility Management Entity

**MOOD** MBMS Operation On-Demand

**MOS** Mean of Opinion Score

**ms** milliseconds

**MTCH** Multicast Transport Channel

**MTU** Maximum Transmission Unit

**NAT** Network Address Translation

**NFV** Network Function Virtualization

**NIMBLE** Network Optimization for eMBMS-Based Live video QoE

**NP** Non-deterministic Polynomial Time

**OFDMA** Orthogonal Frequency-Division Multiple Access

**OTT** Over The Top

**P2P** Point-to-Point

**PF** Proportional Fairness

**PIM** Protocol Independent Multicast

**PMF** Probability Mass Function

**QoE** Quality of Experience

**QoS** Quality of Service

**RB** Resource Blocks

**REST** Representational State Transfer

**RSRP** Reference Signals Received Power

**RTMP** Real Time Messaging Protocol

**RTOP** Real Time Optimal eNB and user Partitioning

**RTP** Real Time Transport Protocol

**RWP** Random Way Point

**SD** Standard Definition

**SDN** Software-Defined Networking

**SFN** Single Frequency Networks

**SINR** Signal to Interference Noise Ratio

**SVC** Scalable Video Coding

**SVEF** Scalable Video Evaluation Framework

**TB** Transport Block

**TCP** Transmission Control Protocol

**UDP** User Datagram Protocol

**UE** User Equipment

**vlog** Video Blogs

**VoD** Video-On-Demand

**WSGI** Web Server Gateway Interface

# Abstract

Recent social media trends have proliferated the demand for High Definition (HD) live streaming events over the Internet. Such events may include sports, TV channels or individuals broadcasting to an audience located across the globe. In live video streaming, multiple users subscribe to the same event at the same time, increasing the peak bandwidth requirements. Consequently, a dynamic and flexible provisioning of network and system resources becomes difficult. With user preferences shifting towards mobile devices, cellular network operators also face challenges when handling flash-crowds viewing live video.

The rising adaptation of Software-Defined Networking (SDN) by Internet Service Providers (ISP) and Content Delivery Networks (CDN) presents an opportunity to dynamically respond to short-lived broadcast events as well as HD mega events. SDN's centralized control is utilized to propose mCast, an architecture that enables inter-domain network-layer multicast and avoids redundant transmissions in core and wired access networks. To handle network congestion, Danos is proposed, that deploys an optimization model to maximize the user video quality while minimizing the resource consumption. The model considers device capabilities, network constraints and user's ISP or CDN subscription levels.

In the cellular domain, the standard Evolved Multimedia Broadcast Multicast Service (eMBMS) improves the utilization of scarce wireless resources. Two key decisions when configuring an eMBMS service area are: which Evolved-Node Base Stations (eNB) to synchronize and how to share resources among users with heterogeneous channel conditions. To optimize network configuration, first RTOP is proposed. RTOP formulates a joint optimization model and employs a real-time heuristics-based algorithm for a single network instance. Then NIMBLE is proposed to address design challenges that arise due to temporal aspects of network and user-state. NIMBLE reacts to variability and re-configures the network to increase user Quality of Experience (QoE).

Large-scale testbeds were developed to compare the proposed algorithms with state-of-the-art approaches and various network and video performance metrics were evaluated . Results showed a 70% increase in average user throughput and elimination of frame drops in core and access network. Similarly, the eMBMS algorithms increased average throughput by 150% and reduced the bitrate switches by 75%. Overall, 80% of the users had an improved QoE.

# Chapter 1

# Introduction

## 1.1 Motivation

In 2019, video accounts for 80% of all the traffic over the Internet [1]. Video streaming services can mainly be categorized as Video-On-Demand (VoD) and live streaming services. Live streaming is the fastest growing type of video traffic and will comprise of 20% of all video traffic by 2022 [1]. A key reason for this rapid increase is the latest social media trend that has seen an extreme rise in individuals sharing live videos with audiences located across the globe [2]. Services like Facebook Live[1] and Twitch[2] have changed the norms of live streaming where, Twitch alone saw a peak of around 4 million concurrent viewers in 2018 (Figure 1.1).

The proliferation of demand for High Definition (HD) video streaming [3] has resulted in frequent short-lived flash-crowd events over the Internet, such as sports, gaming events, Video Blogs (vlog), political events and crowd-source streaming. Consequently, a dynamic and flexible provisioning of network and system resources has become a challenge for Internet Service Providers (ISP). The heterogeneous device capabilities of UEs, ranging from smart-phones and tablets to ultra-HD 4K TVs, further elevate the challenge to dynamically deliver the video streams at multiple bitrates that match the specific needs and requirements of each UE [4].

[1] `https://live.fb.com/`
[2] `https://www.twitch.tv/`

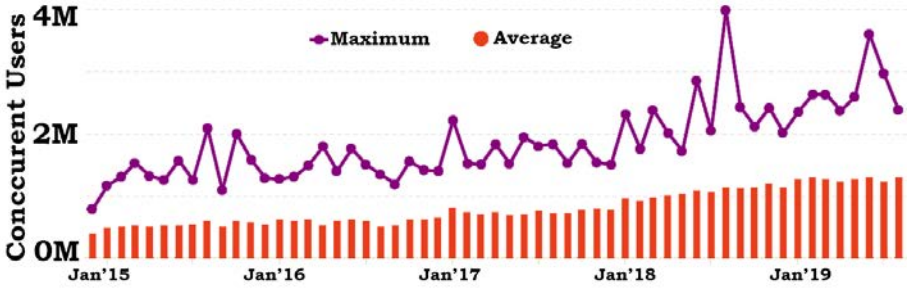



Figure 1.1: Twitch viewers statistics (Source: [2]). *In August 2018, a peak of 4 million concurrent users occurred.*

### 1.1.1 Core and wired access networks

Due to their limited resources, content providers generally rely on Content Delivery Networks (CDN) to distribute video streams globally. More than 60% of video traffic on the Internet passes through CDNs [1]. When delivering live video streams inside an ISP network, CDNs rely on either IP unicast e.g. in Dynamic Adaptive Streaming over HTTP (DASH)-based systems or overlay multicast e.g. in Point-to-Point (P2P)-based systems [5]. These approaches enable good control and management of end-devices by establishing end-to-end connections, but result in redundant transmissions in the network layer and waste of system resources for both ISPs and CDNs.

On the other hand, native IP multicast can help reduce resource consumption by eliminating packet duplication over network links and content servers, but its adoption is stymied by lack of desirable features such as management, authorization and accounting [6]. Today, IP multicast is limited to intra-domain pre-provisioned services such as ISP-oriented IPTV [7].

Software-Defined Networking (SDN) is an emerging approach for network programmability, with the capacity to initialize, control, change, and manage network behavior dynamically via open interfaces [8]. A logically centralized controller, with a global view of network, can monitor every traffic flow, make forwarding decisions and install efficient rules at run-time. The enhanced degree of control with SDN can enable dynamically manageable network-layer multicast over the Internet. An increasing number of ISPs and CDNs are incorporating SDN in their domains [9], which serves as a motivation to rethink the design of live video streaming services.

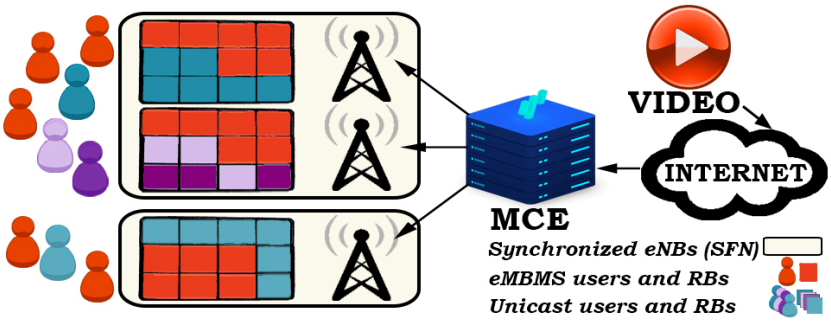



Figure 1.2: eMBMS architecture. *eNBs in a cluster transmit eMBMS content on shared resource blocks (RBs) and schedule their unicast users with the remaining RBs.*

### 1.1.2 Cellular networks

Over the past few years, user preferences have shifted towards streaming content over hand-held mobile devices [10], making it difficult for cellular operators to seamlessly deliver high quality videos to end-users over the scarce wireless spectrum. The problem of resource allocation becomes further challenging when users in densely populated and crowded areas subscribe to live video streams and request the same content at the same time, increasing the peak bandwidth requirements. Using unicast transmission mode in such an environment leads to wasteful resource utilization and poor user experience.

Evolved Multimedia Broadcast Multicast Service (eMBMS) is a Third Generation Partnership Project (3GPP) standard [11] that provides an alternative and more efficient method for delivering live content to a large number of cellular network users. To improve the resource utilization, eMBMS allows sharing resources among a group of users watching the same content and transmitting the content to a group just once. Furthermore, to improve the channel condition of UEs, eMBMS allows Evolved-Node Base Stations (eNB)s in a spatially local area to transmit a video synchronously at a common frequency and time (Figure 1.2), hence creating Single Frequency Networks (SFN) and improving Signal to Interference Noise Ratio (SINR) for UEs.

Cellular network operators continue to evolve their eMBMS deployment strategies. The ultimate goal is to maximize the Quality of Experience (QoE) for users while respecting the resource constraints. To best utilize the scarce wireless resources, operators have to make a few key decisions when configuring the physical network for eMBMS (Figure 1.2):

- As channel conditions of video UEs and resources available across eNBs can

vary, an operator must decide how many clusters to create for each video and which eNBs should be synchronized to form SFN clusters.

- Within an SFN cluster, users interested in the same content may have heterogeneous channel conditions. Placing such users in one multicast group is unfair to the users with good channel condition whereas creating too many groups fails to take advantage of multicast and reduces the cumulative throughput of the system [12]. Therefore, the number of groups, and hence the bitrates to serve for each video, must be chosen optimally.
- Finally, as eMBMS users have to share the Orthogonal Frequency-Division Multiple Access (OFDMA) resources with non-eMBMS unicast users, it is important to consider the unicast load on each eNB in the eMBMS service area and the impact of eMBMS decisions on unicast users.

The existing approaches [12, 13, 14] ignore the inter-dependence of these problems, hence yielding sub-optimal results. Also most of these models are either too complex to solve in real-time for a large number of users [15]; do not consider multiple videos served by eMBMS at the same time [12, 13]; aim to maximize the network throughput instead of the application-level video bitrates [13] or user-QoE [14]; do not consider the time-variance of the network state or mobile UEs and; ignore the impact of eMBMS resource allocation on unicast users [16]. To maximize the benefits and potential of eMBMS, operators need a solution that can run in real-time and jointly solve the user grouping, SFN clustering and resource allocation problems.

## 1.2 Thesis statement and contributions

### 1.2.1 Thesis statement

*Enabling network or physical layer multicast for live streaming services and optimizing or improving the resource utilization can reduce the operational cost for content and network providers and also improve the quality of experience for end-users. Dynamic, cross-layer and real-time solutions are needed for practical deployment and adaptability of multicast for live services over the Internet.*

### 1.2.2 Key contributions

To improve resource utilization and user experience for live streaming services, the following key contributions have been made:

**1. Enabling network-layer multicast for inter-domain live streaming**

A novel architecture, mCast, is proposed for live streaming that merges the flexibility and control of SDN with resource efficiency of multicast to reduce inter-domain and intra-domain traffic in an Internet backbone, core ISP network and wired access network. mCast reduces the implementation cost and complexity of network-layer multicast for ISPs and provides a dynamic and scalable mechanism for multicast tree construction in real-time. CDNs are provided with full control over user sessions and all the necessary information for management and billing of clients. A cost-based decision model is proposed to help CDNs identify, in real-time, when switching from unicast to multicast will be profitable. For evaluation, a large-scale emulated testbed is developed with CDN servers streaming real HD video content to clients located in an ISP network. Extensive experiments were conducted to show the feasibility, scalability and gains of mCast. mCast was compared with standard IP unicast and results showed that in similar network conditions mCast can, not only save significant network and system resources but also deliver a better quality of video to clients with lesser start-up delays and no dropped packets.

**2. Optimizing network resources to maximize user received bitrates and minimize resource utilization**

An Over The Top (OTT) live streaming service, Danos, is proposed that facilitates DASH-like bitrate adaptive delivery, even when using multicast at the network layer. Various design choices are analyzed that can be made by CDNs or ISPs to improve user experience. Where mCast enables inter-domain network-layer multicast, Danos introduces all the essential architectural components to handle network congestion and heterogeneous specifications of User Equipment (UE) devices. A novel multi-objective optimization problem is formulated to maximize the perceived video quality of users and minimize the traffic load on the ISP network. The optimization model accommodates operation constraints for both ISPs and CDNs and scales well with the number of users. The performance analysis of the model showed that it can be solved in the order of milliseconds for millions of users in the network. A prototype of Danos is built for demonstration and evaluation of multiple videos served at multiple bitrates. Real-world scenar-

ios were tested including cross-traffic and flash-crowd events. Results showed a significant increase in average throughput in comparison to mCast.

**3. Optimizing SFN-clustering and user-grouping problems in cellular networks**

A joint optimization problem is formulated for choosing the best SFN clustering configuration, user groups and bitrates to serve for multiple eMBMS video sessions. The solution of this problem determines the performance bound on a rate-based utility at a particular network instance and presents a practical mechanism to handle the impact of eMBMS decisions on unicast users. A scalable heuristics-based algorithm, RTOP, is proposed that finds optimal or near-optimal results in real-time, independent of the number of users in a typical eMBMS service area settings. A discrete event-based Long Term Evolution (LTE) physical layer simulator is designed to evaluate the performance of RTOP. RTOP was compared with state-of-the-art approaches [12, 14] in various network settings, user distributions, number of videos and bitrates to serve. Results indicated significant performance improvements and showed that RTOP always achieved a utility within a 1% gap from the globally optimal solution.

**4. Addressing the impact of temporal network or user state variability on the end-user QoE**

A novel resource management and allocation framework, NIMBLE is presented for cellular networks, that configures the physical eMBMS network with the objective to maximize the end-user experience. The proposed solution is extremely scalable and dynamic, making it feasible for deployment in real-world cellular networks. An optimization model is formulated that considers the three fundamental factors of QoE: the video bitrates received by users, video frames dropped or skipped by users and, the switches in video bitrates encountered by users over time. A heuristics-based algorithm is designed that is agnostic to user mobility pattern and introduces parameters to control the network-state. Frequent network reconfiguration can result in frequent bitrate switches for users and higher overhead cost of network management. Delaying reconfiguration can reduce the responsiveness to UE's channel condition and deteriorate user experience. NIMBLE considers these trade-offs when re-configuring the network. Comparison of NIMBLE with state-of-the-art approaches showed considerable improvement in user QoE and reduction in overhead costs.

## 1.3 Thesis organization

The remainder of the thesis is organized as follows:

**Chapter 2** describes the essential concepts, standards and approaches used for live video streaming services over the Internet and in cellular networks. Furthermore, state-of-the art solutions and metrics to quantify network and video performance are reviewed.

**Chapter 3** presents various simulators, emulators and animators that have been developed to evaluate the proposed algorithms and tools that have been incorporated in the testbeds.

**Chapter 4** introduces the architecture of mCast, proposed components at different SDN layers and, the cost-based decision model for switching between unicast and multicast. mCast is compared against standard unicast in different network topologies and the results are analyzed.

**Chapter 5** shares the additional components, design choices and, the global optimization model implemented by Danos in SDN architecture and end-nodes, to enable bitrate adaptive streaming. Danos is compared against mCast in real-world scenarios and various network and user metrics are studied.

**Chapter 6** formulates the joint optimization problem for RTOP and explains the heuristic approaches applied for real-time computation. Results of comparing RTOP with state-of-the-art eMBMS research are shared.

**Chapter 7** presents NIMBLE and various modeling techniques, stability parameters and design choices made by NIMBLE to maximize user QoE over time. NIMBLE is compared with RTOP and other similar solutions from literature and the results are analyzed.

Finally, **Chapter 8** summarizes the work and key findings and explores the future research directions.

## 1.4 List of publications

Peer-reviewed conference and journal publications from the work described in this thesis are listed below:

1. **Ahmed Khalid**, Jason J. Quinlan, and Cormac J. Sreenan, MiniNAM:

A network Animator for Visualizing Real-time Packet Flows in Mininet", Proceedings of the IEEE 20th Conference on Innovations in Clouds, Internet and Networks (ICIN), March 2017.

2. **Ahmed Khalid**, Ahmed H. Zahran, and Cormac J. Sreenan, "mCast: An SDN-Based Resource-Efficient Live Video Streaming Architecture with ISP-CDN Collaboration", Proceedings of the IEEE 42nd Conference on Local Computer Networks (LCN), October 2017.

3. **Ahmed Khalid**, Ahmed H. Zahran, and Cormac J. Sreenan, "Prototyping and Evaluating SDN-based Multicast Architectures for Live Video Streaming", Proceedings of the IEEE 42nd Conference on Local Computer Networks (LCN), October 2017.

4. **Ahmed Khalid**, Ahmed H. Zahran, and Cormac J. Sreenan, "RTOP: Optimal User Grouping and SFN Clustering for Multiple eMBMS Video Sessions", Proceedings of the IEEE Conference on Computer Communications (INFOCOM), April 2019.

5. **Ahmed Khalid**, Ahmed H. Zahran, and Cormac J. Sreenan, "An SDN-Based Device-Aware Live Video Service For Inter-Domain Adaptive Bitrate Streaming", ACM Multimedia Systems Conference (MMSys), June 2019.

As complementary contribution (and not part of this thesis), the following papers were published to help the multimedia community by presenting platforms for evaluating DASH-based videos:

6. Jason J. Quinlan, Darijo Raca, Ahmed H. Zahran, **Ahmed Khalid**, K.K. Ramakrishnan, and Cormac J. Sreenan, "D-LiTE: A platform for evaluating DASH performance over a simulated LTE network", Proceedings of the 22nd IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN), June 2016.

7. Jason J. Quinlan, Alexander Reviakin, **Ahmed Khalid**, K.K. Ramakrishnan, and Cormac J. Sreenan, "D-LiTE-ful: An evaluation platform for DASH QoE for SDN-enabled ISP offloading in LTE", Proceedings of the 10th ACM International Workshop on Wireless Network Testbeds, Experimental evaluation & Characterization (WINTECH) at the ACM MobiCom Conference, June 2016.

# Chapter 2

# Background and literature review

## 2.1 Video streaming standards and protocols

The Internet is a network of interconnected network domains. Similar to most online services today, there are three important domains for Over The Top (OTT) video delivery services: Content providers generate the video content, however it is usually not profitable for them to maintain their own infrastructure for global delivery; Content Delivery Networks (CDN), such as Amazon or Akamai [17], distribute the content from content providers across the globe and; regional Internet Service Providers (ISP) deliver it to end-users over a wired medium, e.g. in Digital Subscriber Line (DSL), or a wireless medium, e.g. in cellular networks. The video streaming services provided over such an Internet setup can be classified into two main categories:

- Video-On-Demand (VoD): VoD services, such as Hulu, YouTube and Netflix [18] allow users to download and stream pre-recorded content at any time. Different users may request different videos at the same time or the same content but at different times.
- Live Streaming: Services, such as Periscope, Facebook Live and Twitch [19], serve users with live content, possibly encoded at multiple bitrates to deal with heterogeneous user network and device specification. Different users are delivered the same content at the same time.

At the application layer most of these services implement either: Real Time Messaging Protocol (RTMP) [20] that offers low-latency and hence is a preferable choice for live video streaming or; Dynamic Adaptive Streaming over HTTP

(DASH) [21] that partitions a video file into segments and delivers to a client using standard HTTP, increasing the reach of the streaming service. At the transport layer, Transmission Control Protocol (TCP) can be used for reliable delivery. However, due to the high-speed and low-latency provided by User Datagram Protocol (UDP), and relatively better reliability provided by its variants, e.g. QUIC [22], UDP has become the most widely used protocol for Over The Top (OTT) streaming [23].

Regardless of the higher-layer protocols used, popular events with large number of interested users give rise to events known as flash-crowds [19]. Flash-crowd events can result in either the content servers being unable to handle the volume of requests for content or the underlying network capacity falling short of the bandwidth requirements. The strict temporal locality in live streaming services, gives rise to frequent flash-crowd events. This serves as a motivation for both ISPs and CDNs to devise dynamic and efficient solutions that can maintain reliable service during flash-crowd events.

Note that such events can also occur in VoD services, especially for newly uploaded videos on popular channels [24]. VoD users requesting the same content within a certain time-frame posses similar characteristics as live streaming users, in terms of temporal aspects and as such, solutions devised for handling live flash-crowds can also be adapted for VoD.

## 2.2 Video transmission modes

Traditional TV channels are delivered to users using broadcast transmission over satellite, cable [25] or via Digital Terrestrial Television (DTTV). Broadcasting is a one-to-all transmission approach where, subject to authorization, all the users in the network are delivered the content throughout the streaming duration, possibly resulting in unnecessary transmissions and lack of user-specific session details. For better control and utilization of available resources, the following transmission modes are used when delivering TV channels or any other form of live content to users over the Internet. A qualitative analysis of these modes is summarized in Table 2.1

Table 2.1: Qualitative analysis of video transmission modes

| | **IP Unicast** | **Broadcast** | **IP Multicast (IPM)** | **Overlay Multicast** |
|---|---|---|---|---|
| Bandwidth Efficiency | Low | High | Very High | Very Low |
| Server Load | High | Low | Low | Low |
| Inter-domain Operability | High | Very low | Very low | High |
| CDN Controlability | Very High | Very Low | Low | High |
| Reliability | TCP/UDP | - | UDP Only | TCP/UDP |
| Client-Side Requirements | None | System Dependent | Must support IPM protocols | Runs overlay application |

### 2.2.1 IP unicast

Unicast is a one-to-one form of transmission where each destination node is identified over the Internet by a unique IP address. This provides a high inter-domain operability and hence unicast is the most common OTT delivery mechanism. For video streaming, unicast involves establishing an end-to-end connection between a content server and a video client. This provides the content providers or CDNs with full control over the sessions of their end-users and a granular detail of the content received by each video client. Such information is necessary for accounting purposes and is a key element of the business models of content providers and CDNs. The end-to-end connection also facilitates reliable transport by using TCP or a reliable form of UDP, e.g. QUIC [22] or RUDP [26].

Because users of live video streaming are concurrent, a drawback of having a separate connection for each client is a large number of connections from a content server to end-users, where each connection carries the same content. For spatially localized users, this problem extends to ISP's network as well. Several efforts have been made, e.g. NOVA [27] and SABR [28], to try and optimize the network for DASH-based video delivery over unicast. While such solutions can improve user experience in un-congested network scenarios, for mega-events and flash-crowd scenarios, using unicast results in inefficient bandwidth utilization and high server loads, negatively affecting user's received video quality [29].

### 2.2.2 IP multicast

From a functional point of view, IP multicast would be a desirable choice to avoid sending the same video stream to potentially millions of users in parallel. Multicast is a one-to-many or many-to-many transmission approach where a group of users is identified by a unique IP address and served with a single transmission i.e. a content is transmitted only once over a particular link. Unlike broadcast transmission, copies of video packets are only created and sent over links that are on routes to users in the multicast group. A multicast tree, with routes to all the group users, is constructed at the start of the stream using a multicast routing protocol, such as Protocol Independent Multicast (PIM) [30]. As more users join or leave the multicast group, a protocol, e.g. Internet Group Management Protocol (IGMP) [31], is used for user management that communicates with the routing protocol to update the multicast tree accordingly.

By avoiding redundant transmissions, IP multicast achieves high bandwidth efficiency and can significantly reduce the load on content servers. However, due to its various well-known limitations, the protocol is not used for inter-domain i.e. OTT streaming services. From a CDN's point of view, IP multicast does not provide features, such as user management, authorization, billing policies, data privacy and security [6]. As mentioned in Section 2.2.1, these features are of crucial importance to the business and financial model of CDNs and hence for live streaming, CDNs do not adopt IP multicast.

From an ISP's point of view, if the source of a multicast tree is located outside of their domain (which is the case for OTT services), the management of the tree becomes difficult and raises security concerns [32]. ISPs maintain strictly-configured network topologies and carefully planned routing policies [33]. As multicast group management and routing updates are usually shared via broadcast, an ISP will have to broadcast packets and configure its forwarding nodes based on messages generated outside of its domain which is undesirable. Furthermore, network management in IP multicast is challenging [34] because the focus is on managing multicast groups and users are not handled or identified uniquely by the management protocols.

Several efforts have been made to resolve the deployment issues of IP multicast. Lee et al. [35] propose changes to IGMP which can speed-up the group management operations. ESM [36] proposes an efficient multicast routing protocol for data centers by implementing a source-to-receiver expansion approach, rather

than the standard receiver-driven approach used by standard multicast routing protocols [30]. Hwang et al. [37] introduce Quality of Service (QoS) guarantees for IP multicast delivery of intra-domain traffic. Srinivasan et al. [38] use a logical-key tree structure and Chinese remainder theorem [39] to improve the security of multicast networks. While these approaches resolve some of the IP multicast issues, they do not address the factors that hinder inter-domain operability of IP multicast. As such, in today's Internet, the use of IP multicast is limited to services within a certain network domain. For example, ISP-oriented IPTV services [7] or cable networks (DOCSIS [25]) can pre-provision users and statically configure their infrastructures to efficiently deliver live content using multicast. However, OTT live streaming services still suffer from bandwidth inefficiency and high loads on content servers.

### 2.2.3 Application-layer multicast

In an attempt to reduce the load on content servers and enable inter-domain operability, Application-Layer Multicast (ALM), also known as Overlay Multicast, has been proposed [40]. Simulcast [41] is an early example of such an approach and recently Point-to-Point (P2P)-based systems [42] are a common example that use ALM for content delivery. To enable inter-domain operability and high control granularity, ALM streaming services use IP unicast at the network layer and instead implement multicast at the application layer by forming an overlay network of interconnected nodes. A hierarchical overlay multicast tree is constructed and users act as active transmitters by sharing parts of the content with users designated as their children. Prior research reports that P2P live streaming suffers from unstable video quality and playback lags due to peer churn and limited uplink bandwidth of end-users [42].

Due to end-users forming part of the multicast forwarding tree, ALM services have higher vulnerability to security attacks by malicious nodes [43]. The security concerns along with the need for complex client-side algorithms, makes it difficult to attract users watching live streams on low-end devices. Recent efforts, such as AngelCast [5], augment overlay multicast with CDN clouds to reduce the latency and improve the streaming quality, but ALM is oblivious to the underlying network state and the inability to handle congestion in the infrastructure repels users with dynamic network conditions. Even though ALM may reduce the load on content servers and improve the streaming situation for CDNs, it makes resource management more challenging for ISPs, as even more unicast flows are

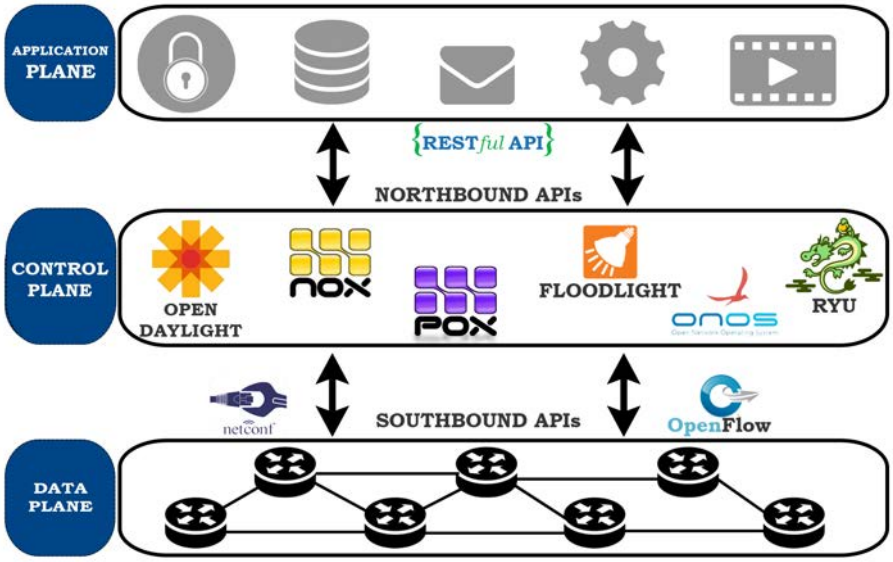



Figure 2.1: SDN architecture. *The separation of data and control plane provides SDN with global view and centralized programmable control.*

generated, in comparison to IP unicast, between clients inside and outside the ISP network.

## 2.3 Multicast in programmable networks

A programmable network is a network in which the behavior of network devices and flow control is handled by software that operates independently from network hardware. Various limitations of the traditional Internet architecture, outlined in Section 2.2, can be overcome by using a programmable underlying network and offering services on top to manage and share live video streams. Software-Defined Networking (SDN) is the current standard technology for network programmability.

### 2.3.1 SDN and its benefits

SDN [8] is a programmable network architecture that decouples the control plane from the forwarding plane and brings it to a logically or physically centralized location. Figure 2.1 shows the architecture of an SDN domain. The data plane consists of forwarding nodes that have the capabilities to process and forward incoming data traffic and are connected to the control plane using Southbound APIs, such as OpenFlow [44], NETCONF [45] or OVSDB [46]. The control plane

consists of a controller software, e.g. OpenDaylight [47], ONOS [48] or Ryu [49], with multiple control functions and features. The controller is responsible for configuring the data plane i.e. the forwarding nodes based on the instructions that it receives from the application plane. The controller also extracts information about the network from the hardware devices and communicates back to the SDN applications with an abstract view of the network using Northbound APIs. The application plane hosts programs and services that the network provider wants to provide to its users. These applications can include networking management, analytics, security, multimedia or business-related services.

SDN has several advantages over traditional networking. It is dynamic, manageable, cost-effective, and adaptable, making it ideal for the high-bandwidth, dynamic nature of live video streaming. The logically centralized controller, with a global view of network, can monitor every traffic flow, make forwarding decisions and install dynamic rules at run-time. The knowledge and awareness of network nodes and clients by the controller in SDN can be used to construct resource-efficient multicast trees in an ISP network. The flexibility provided by SDN at network layer and the ability for inter-controller communication can be utilized to develop a multicast service for inter-domain video traffic coming from a CDN to an ISP network. The abstract view provided to the services at the application plane can help build an inter-domain multicast service while retaining data privacy and providing security.

### 2.3.2 IP multicast in SDN

Pertaining to the increasing adaption of SDN by network providers and data centers [9], research has been conducted to improve the mechanism of IP multicast using the SDN architecture. In [50], an SDN-based system is proposed that allows fast failure recovery for IP multicast trees. CastFlow [51] presents an approach where a centralized SDN controller manages IP multicast; all IGMP messages are sent to the controller that calculates multicast groups and configures network in advance. Craig et al. [52] measure real-time link cost modification to apply load balancing on multicast traffic and handle or avoid network congestion.

Other works propose some elaborate multicast routing algorithms [53, 54] that can be used in an SDN. The SDN controller can receive network statistics from all the network nodes and switches and can use this global information to construct efficient paths. Reza et al. [55] model multicast routing as a delay constraint least

cost (DCLC) problem, solves an approximation and dynamically configures the network using SDN to guarantee QoS.

While such solutions improve the management of IP multicast, other issues such as handling inter-domain traffic remain unaddressed. In Lcast [56], a network-layer inter-domain multicast framework is proposed that creates a router overlay to connect multicast hosts in different domains. Solutions like Lcast enable inter-domain multicast but fail to present CDNs with enough control over clients to provide a viable live streaming service.

### 2.3.3 SDN-based network-layer multicast

To fully exploit the benefits of SDN, a few researchers have proposed to redefine the way network layer multicast is handled and introduce sufficient control to content or network providers to realize a video streaming service. Yang et al. [57] exploit Scalable Video Coding (SVC) to divide a video stream in separate flows and creating a multicast group for each flow. A flow represents a layer of SVC video and users located on congested paths are added to lesser number of flows i.e. layers. Instead of using IGMP, the centralized control of SDN is used to dynamically reconfigure the network and user groups. This work does not define a communication framework between ISPs and CDNs and does not take into account data privacy concerns of a CDN. Similar work for Multiple Description Coding (MDC) is proposed in [58] where different descriptions of the video are configured as different multicast flows and policies are implemented to ensure minimum quality to users by delivering at least one description.

The advent of Network Function Virtualization (NFV) along with SDN has further revitalized the demand of multicast at the network layer. LiveJack [59] presents a network service that allows CDN servers to leverage ISP cloud resources and extend multicast towards the edge. In [60], SDM is proposed to enable ISPs to support resource efficient P2P streaming; a virtual peer is created inside an ISP network allowing an external streaming source to gain a presence. The ISP then distributes traffic to its clients using Network Address Translation (NAT)-like forwarding rules. Although a P2P service, this work gives a good idea of what SDN can do to create dynamic networks, save resources in core or access networks, and improve video quality for end-users.

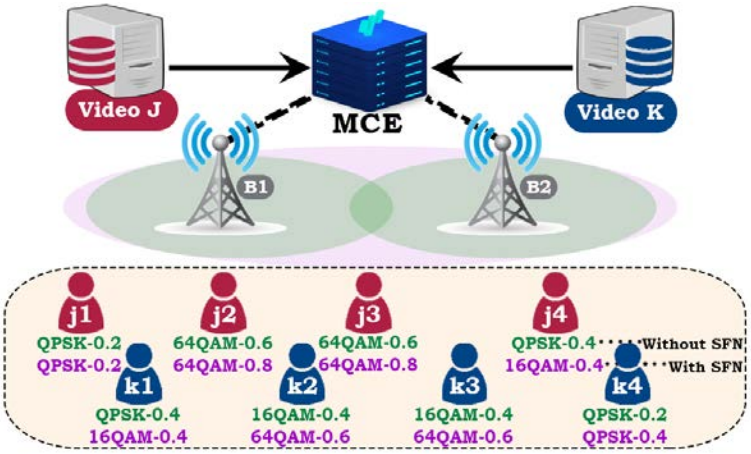



Figure 2.2: An example of eMBMS configuration. *Synchronizing eNBs can improve the achievable MCS and spectral efficiency of users.*

## 2.4 Multicast in cellular networks

In comparison to core or wired access networks, the resources available in cellular networks are substantially lower. Therefore, when delivering live video streams using IP unicast, cellular networks suffer even more with inefficient resource management especially during flash-crowd events or high cross-traffic. To improve the resource utilization, 3GPP has standardized Evolved Multimedia Broadcast Multicast Service (eMBMS). eMBMS provides an alternative and more efficient method for delivering live content to a large number of cellular User Equipment (UE) devices.

In LTE networks, Resource Blocks (RB) are the unit of resource allocation (Figure 1.2) and they represent the time and frequency at which a signal is transmitted. Evolved-Node Base Stations (eNB) schedule their non-eMBMS unicast UEs by allocating them RBs, often through some variant of a proportionally fair scheduler [61]. While doing so, the eNB chooses a Modulation and Coding Scheme (MCS) for each UE based on their reported channel condition. A higher MCS yields higher spectral efficiency however, if a UE is assigned an MCS higher than what their channel condition can support they are unable to decode the signal properly, and experience packet losses.

### 2.4.1 Background on eMBMS

For eMBMS [11], users can be combined into groups based on their video of interest and channel conditions, with each group receiving the video at a different

bitrate. The content is transmitted just once to a group and the UEs are scheduled to receive the transmission over the same set of RBs. To ensure that all group-users can properly decode the signal, the MCS of a group is restricted to the UE with the worst channel condition [12]. Therefore, to increase the spectral efficiency, it is essential to create groups of users commensurate with their channel conditions. To improve the achievable MCS for UEs, eMBMS proposes creating Single Frequency Networks (SFN)s in densely populated areas when multiple neighboring eNBs have to serve the same video (Figure 2.2). eNBs that are part of an SFN, transmit a video in a synchronous manner over the same set of RBs. Interested UEs combine the received signals, boosting their Signal to Interference Noise Ratio (SINR). As each eNB may have a different load of non-eMBMS UEs, placing an eNB with a higher load in an SFN may limit the amount of RBs available to an eMBMS session. Therefore, to increase the achievable throughput, it is important to consider unicast load at each eNB as well as eMBMS user distribution.

As an eMBMS service area may include multiple eNBs, the scheduling, user grouping and SFN clustering for eMBMS is handled by a centralized entity called Multicast Coordination Entity (MCE). Rather than handling themselves, eNBs rely on an MCE to manage the eMBMS users and resource allocation. MCE is a logical entity and, physically it may be integrated into another network element. eMBMS adds MCE to the Evolved Packet Core (EPC) of LTE and defines an interface (M2) for communication between eNBs and MCE. Additionally, an interface (M3) is defined for communication between MCE and Mobility Management Entity (MME) and helps MCE in acquiring eMBMS session details. Note that there is no communication between end-users and MCE, because eMBMS is a multicast-based service where users do not provide detailed feedback to the serving entity.

### 2.4.2 Related work on eMBMS

eMBMS enables multicast at the physical and link-layer of cellular networks by configuring UEs to receive video content over shared wireless resources. Some research, e.g. [62] has been conducted to incorporate forward-error correction in eMBMS at the application layer to improve reliability in the absence of detailed user feedback. However, such works do not address issues and challenges at lower layers e.g. efficient physical-layer resource scheduling and allocation.

Several researchers have proposed multicast resource allocation techniques for eMBMS users. Hou et al. [63] propose using UEs with good channel conditions as relays for UEs with bad channel conditions. Such methods are not realistic due to the greedy nature of users and low-latency requirements of live streaming. Muvi [13] uses scalable video coding in an attempt to maximize the utility for multicast users but does not consider the impact on unicast users. Won et al. [64] solve the user grouping problem for a single-cell network and assigns an MCS value to each group with the goal to maximize Proportional Fairness (PF) which is defined as the sum-log of bitrates assigned to users. This work sometimes places users in groups where the MCS value is higher than what they can decode, resulting in packet losses for such users.

Chen et al. [12] propose an optimization model that considers grouping users based on their channel conditions, while considering the impact on unicast users. The objective of the optimization is to maximize throughput based on PF-utility. Although this approach ensures that all users are assigned some resources, it does not guarantee that those resources are enough to achieve at least the minimum bitrate of the transmitted video. Also, the model does not consider the presence of multiple videos or the possibility of creating multiple SFN clusters.

In [65], Monserrat et al. evaluate how the number of eNBs in an SFN cluster affect the eMBMS service, but they do not propose any solution for determining the best eNB configuration. BoLTE [14] addresses the SFN clustering problem for multiple broadcast sessions by placing eNBs with less SFN gains in separate clusters. However, BoLTE does not explore the possibility of grouping UEs based on their channel conditions and always assumes all the UEs within an SFN cluster to be in the same group. This approach is unfair to UEs with good conditions and limits the achievable utility of the system.

### 2.4.3 eMBMS configurations

Based on the resource allocation schemes proposed in the literature, various key configurations for eMBMS can be identified. These are illustrated in this section, by considering the example shown in Figure 2.2. Two videos (J and K) are served using eMBMS at two different bitrates (200kbps and 400kbps) by two eNBs (B1 and B2). A set $M$ contains the eMBMS users interested in each video and consists of eight users ($j1$, $j2$, $j3$, $j4$, $k1$, $k2$, $k3$ and $k4$). In Figure 2.2, the MCS values on top are what users can achieve from eNB configuration {{B1}, {B2}}, i.e. eNBs

(a) No eMBMS. *All users treated as unicast.*

(b) Standard eMBMS. *1 User Group 1 SFN Cluster*

Figure 2.3: Example of no eMBMS and standard eMBMS configuration

split into two clusters and the bottom MCS values are the achievable MCS from cluster {{B1, B2}}, i.e. both eNBs are synchronized to form one SFN cluster. Both eNBs have 10 RBs for eMBMS and users of Video J and K can only be served with these RBs. The bitrate assigned to a user $m$ is denoted by $r_m$. Five possible network configurations are explored and two performance metrics are measured:

- The average user bitrate, calculated as $\frac{\sum_{m \in M} r_m}{|M|}$.
- *Sum-log* utility of user bitrates, calculated as $\sum_{m \in M} log(r_m)$.

Case a (**No eMBMS**): All users are scheduled separately as unicast users and the eNBs are placed in separate clusters i.e. not synchronized (Figure 2.3a). Due to the limited resources available (10 RBs) at each eNB, only three users ($j2$, $j3$ and $j4$) could be served with the higher bitrate (400kbps). The sum-log utility is 44.46 and the average user bitrate is 275kbps.

Case b (**Standard *eMBMS***): All eNBs are in one cluster and all users of a session in one group (Figure 2.3b). As RBs are shared, each user gets more RBs than in unicast. However, the user with the lowest MCS value ($j1$ for Video J) restrains the rate for all users. Therefore even though $j2$ and $j3$ have high spectral efficiency, they receive the lower bitrate (200kbps), limiting the total users served with 400kbps to four. The system utility is 45.12 and the average bitrate increases by 10% in comparison to unicast.

Case c (***eMBMS* with SFN clustering** [14]): For Video J, splitting eNBs into two clusters (Figure 2.3c) places $j4$ in a separate cluster than the user with the worst channel condition i.e. $j1$. This allows $j4$ to achieve a higher MCS and hence receive the higher bitrate. However users $j2$ and $j3$ are still restricted to the lower bitrate because of user $j1$, who is unable to receive the higher bitrate

(c) eMBMS with SFN clustering but no user grouping

(d) eMBMS with user grouping but no SFN clustering

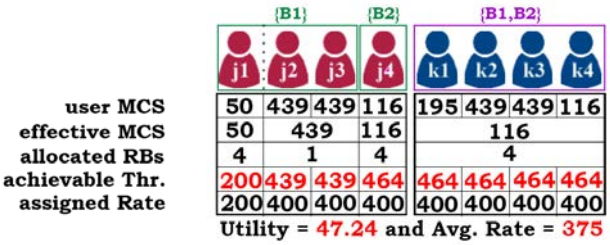
Figure 2.3: Example of SFN clustering and user grouping configuration

due to its poor channel condition. No change is made to Video K as all the users are already receiving the higher bitrate and assigning further resources will not increase the utility for these users. With five users receiving 400kbps, the system utility increases to 45.84 and the average bitrate by 8% in comparison to standard eMBMS.

Case d (***eMBMS* with user grouping** [12]): Instead of SFN clustering, user grouping is applied to standard eMBMS case (Figure 2.3d). With the eNBs synchronized, users $j2$ and $j3$ have distinctly higher MCS in comparison to $j1$ and $j4$. Hence they form a separate group and receive the higher bitrate, while leaving enough resources for the second group ($j1$ and $j4$) to receive the lower bitrate. Now six out of eight users are served with the higher bitrate and the average bitrate increases by 17% in comparison to standard eMBMS. The sum-log utility of the system increases to 46.54.

Case e (***eMBMS* with user grouping and SFN clustering**): Unlike Video K, the user distribution of Video J is such that the users achieve little ($j2$, $j3$ and $j4$) or no ($j1$) benefit by synchronized eNBs. Splitting the eNBs in two clusters for Video J places $j1$, $j2$ and $j3$ in B1 and $j4$ in B2 which now has less users and hence more RBs available for $j4$. This enables $j4$ to receive the higher bitrate as well (Figure 2.3e). On the other hand, users in the cluster B1 can be split into two groups, similar to Case d. For video K, the same solution is maintained i.e. all eNBs synchronized and all users in one group. This configuration gives us the optimal solution with seven out of eight users receiving the higher bitrate. The system utility increases to 47.24 and the average bitrate by 25% in comparison to standard eMBMS.

This example shows that the total utility of the system depends on the eNB configuration chosen for each video, the number of user groups created in each




(e) eMBMS with both SFN clustering and user grouping

Figure 2.3: Example of joint user grouping and SFN clustering configuration

SFN cluster, the number of users placed in a group and the number of RBs and bitrate assigned to each user group. None of the existing approaches in the literature propose jointly optimizing user grouping and SFN clustering with a different configuration possible for each video.

## 2.5 Video performance metrics and indicators

There are three key areas to consider when measuring the performance of a video streaming service: the Quality of Service (QoS) provided by the network, the Quality of Experience (QoE) for the video received by users and the network or system resources consumed due to the service. For each of these categories, some of the following metrics are used in the subsequent chapters for performance evaluation.

### 2.5.1 Quality of service

In OTT streaming, QoS is affected by various factors, such as reliability, scalability, effectiveness, maintainability and congestion in the network. As the video stream travels through the Internet from a content server to an end-user, it may encounter degradation.

#### 2.5.1.1 Errors and losses

Sometimes packets are corrupted due to bit errors caused by noise or interference, especially in wireless communications. The unit of physical layer transmission in cellular networks is a Transport Block (TB) and transmission errors can result

in lost TBs at the receiver [11]. Because there is no data re-transmission in multicast-based services, e.g. eMBMS, minimizing lost TBs is of crucial importance. At the network layer, even when there are no transmission errors, congestion in the network queues due to insufficient capacity can result in lost packets. A reliable network and service should either avoid network congestion or detect it when it happens and handle the congestion to reduce or, when possible, eliminate packet losses.

#### 2.5.1.2 Throughput and goodput

Throughput is the size of successfully delivered packets over a time period. It is generally measured in Kilo-bits per second (Kbps) or Megabits per second (Mbps). As a video frame is composed of one or more network packets, failure to deliver some such packets may prevent a video client from properly decoding the video frame, even if some of the packets belonging to that frame are successfully delivered. Goodput is defined as the rate of useful packets delivered to a UE and it ignores the received packets that get discarded. The common unit to measure Goodput is also Kbps or Mbps. Note that measuring goodput involves the knowledge of a user's protocol stack and hence, is a metric that is measured at the user-end rather than in-network.

#### 2.5.1.3 Link utilization

Link utilization is the average traffic over a particular link expressed as a percentage of the total link capacity. An ISP may implement service differentiation in the network and assign a slice of network to the streaming service. In such cases, even a 100% link utilization will not affect cross-traffic users, i.e. users that are not part of the streaming service. However, without service differentiation, where network queues are shared, a high link utilization by live streaming could result in network congestion or reduction in throughput for the cross-traffic users. Therefore, an effective service should try to minimize link utilization in such scenarios, while maintaining a good or high quality of video delivery.

#### 2.5.1.4 Failed client connections

If the content servers are overloaded, i.e. their processing capabilities is insufficient to handle the volume of received requests in a certain time frame, then the

connection requests sent by video clients of UEs will start getting dropped. This event is defined as a failed client connection [66]. A scalable and well-managed streaming service should be able to minimize or eliminate failed client connections.

#### 2.5.1.5 System utility

To measure the overall performance of the network, providers usually define a system-wide utility function. The utility can be an aggregated function of one or more aforementioned metrics. Generally, system throughput is used to calculate the utility, for example when scheduling resources for UEs, 4G Long Term Evolution (LTE) networks calculate Proportional Fairness (PF) utility, defined as sum-log of throughput achieved by UEs [61].

### 2.5.2 Quality of experience

From a user's perspective, the QoS and associated metrics hold little importance and what actually matters is the video-watching experience. This is captured by Quality of Experience (QoE) metrics. QoE is a measure of the overall delight or annoyance of a customer's experiences with a service. This is generally a subjective metric and measured on a scale such as Mean of Opinion Score (MOS) [67]. The main factor that defines user experience is the average bitrate or throughput of the user over the streaming duration. However, some other key QoE factors have been identified by research studies [68].

#### 2.5.2.1 Initial startup delay

Slow startup and video loading time increases user abandonment. A study conducted on 23 million video views [69] revealed that for more than 2 seconds loading time, users started to leave. At 10 second delay, more than 50% users abandoned the stream and the majority of them never came back. For some live events, e.g. sports, these delays can be far more crucial due to the time-sensitivity of the content.

#### 2.5.2.2 Dropped frames and stalls

Another key factor that affects user experience is the amount of dropped frames. A frame may be dropped due to network losses or missing dependencies. Based on the number of frames per second (fps) in the encoded video, a loss of a few frames may not be noticeable by a human eye. But too many dropped frames, especially consecutive frames, can result in video stalls or freezes. To improve a user's experience, it is critical to minimize the frequency and duration of stalls in a video stream. When a stall occurs, Video-On-Demand (VoD)-based clients start re-buffering packets and resume when enough packets have been received to successfully decode and play frames. Due to the strict time deadlines in live clients, recovering from a stall may involve frame skipping i.e. resuming at the current time rather than where the stall occurred, or fast-forwarding by displaying selective frames and catching up with the newest transmitted frames. Regardless of the technique used for recovery, stalls carry arguably the highest negative weight on user's live streaming experience [69].

#### 2.5.2.3 Switches in video bitrates

The adaptive bitrate streaming, such as DASH, vastly improves the overall user streaming experience by reacting to the underlying network or user state and dynamically increasing or reducing the transmitted bitrate. However, this results in bitrates switching between high and low resolutions. Frequent switches can have a negative impact on user QoE. Based on a study report [70], on average less than one switch per minute was acceptable by users, but more switches than that annoyed users and negatively impacted their streaming experience. In addition to the number of switches, a switching magnitude i.e. difference in the switching bitrate, may also be of relevance [71]. For example, switching down from a bitrate of 8Mbps to 4Mbps may not be as noticeable as switching down to 1Mbps. In general, while switching carries lesser weight on QoE in comparison to stalls, in a highly throughput-fluctuating environment, such as cellular networks, an algorithm that ignores the impact of bitrate switches on users may perform poorly in terms of the overall QoE.

#### 2.5.2.4 QoE metrics

To measure QoE as a metric, subjective and objective approaches are used to identify degradation that may arise at any stage in an end-to-end streaming service, e.g. due to congestion in network or decoding at video client. Although QoE is a subjective metric that can only truly be measured through user feedback, e.g. by using MOS, the objective functions can be helpful and convenient when designing algorithms and measuring or comparing performance of various algorithms [72].

These functions and metrics can be classified as direct and indirect [73]. Direct metrics include factors that directly affect the user perception of the received content, e.g. frame losses or blurs, whereas indirect metrics consider the factors that are not directly related to the content but can have undesirable side effects on user experience, such as start-up delays or asynchronous delivery to multiple live video users.

QoE inspection solutions can be assessed with full, reduced or no reference to the original video source [74]:

- Full Reference involves comparing the user's received video to the original uncompressed video source. Metrics such as PSNR [75], SSIM [76] and VQM [77] use this approach when evaluating QoE. Due to the need for presence of the original video frames, such metrics cannot be used for real-time computation and evaluation of streaming services.
- Reduced Reference, e.g. [78, 79], involves comparing the user's received video to the video transmitted by the server, often through an alternative channel between users and servers to transmit the parameters of the original content in a reduced way.
- No Reference involves evaluation of the video received at the user-end without using any reference video, allowing for fully flexible real-time implementation. Such metrics rely on a combination of the aforementioned QoE parameters, such as delays, drops and switches. In [80], Liu et al. conduct subjective tests to derive impairment functions for different QoE parameters and formulate an overall QoE model. A_PSQA [81] presents a hybrid subjective and objective approach to evaluate end-user QoE. Duanmu et al. [68] apply regression-based techniques on subjective scores reported by users and derive QoE equations with high correlation.

#### 2.5.2.5 Fairness

Aside from QoE, another important factor that may affect users is the fairness. Fairness measures are used in networks to determine whether users or applications are receiving a fair share of system resources. In wired systems, where users have homogeneous underlying network capacities and condition, Jain's index [82] is commonly used. Jain's index uses a variation coefficient to measure fairness and hence works well when ratio scales are applied, such as on user throughput. In contrast, QoEs are usually measured over interval scales e.g. a 5-point Mean of Opinion Score (MOS) with 1 indicating lowest quality and 5 indicating highest quality. Because coefficient of variance is meaningless for such scales, F-index [83] proposes standard deviation in QoE values of users along with lower and upper QoE bounds to measure fairness among user QoEs. F-index can take any QoE metric [68, 80] as an input and returns a 0 to 1 fairness ratio with 1 being perfectly fair.

While higher-layer fairness metrics work well for users with similar network state, UEs in wireless environment may have disparate channel conditions and the spectral efficiency or maximum achievable throughput of users may be different. Expecting such an environment to allocate similar bitrates to users and have low standard-deviation can result in inefficient resource utilization. For such networks, the fairly shared spectrum efficiency (FSSE) is a common approach to jointly measure fairness and system spectrum efficiency. A widely used metric in wireless networks is Proportional Fairness [61] which aims to maximize the sum-log of user throughput, often with variations to avoid scheduling starvation (i.e. some users allocated no spectrum). PF can be extended to maximize user bitrates or QoE, where such information can be measured or estimated by network entities.

### 2.5.3 Resource consumption

If a streaming service is too complex, it may consume too many network or system resources, resulting in high overhead costs and rendering the real-world deployment infeasible. A few common overhead costs incurred on network by streaming services are described below.

#### 2.5.3.1 Signalling messages

When reconfiguring Software-Defined Networking (SDN) nodes, messages are shared between the SDN controller and network forwarding nodes, and forwarding entries are modified or added to the forwarding tables of the network nodes. The signalling messages and forwarding entries are measured for the proposed schemes. Similarly in cellular networks, reconfiguring eMBMS implies sharing messages between the Multicast Coordination Entity (MCE) and eNBs and, actions taken in the form of advertising the reconfiguration updates. Due to the highly dynamic and mobile nature of cellular networks, the cost of reconfiguration can get high. For the eMBMS-related proposed solutions, various types of reconfiguration are measured and analyzed.

#### 2.5.3.2 Computation time

For implementation of an algorithm or service, in a real-world dynamic environment, it must be able to compute the solution in real-time. Large-scale live streaming services are usually very dynamic with users frequently leaving or joining video streams or user-state frequently changing, in the case of mobile users. When solving network reconfiguration problems for SDN or eMBMS, depending on how long the algorithm takes for computation, an optimally computed solution may have become sub-optimal. Additional time may also be needed to reconfigure the network based on the computed solution. Therefore, it is important to devise an algorithm that converges quickly, preferably in orders of milliseconds [84].

#### 2.5.3.3 CPU utilization

As the volume of user requests and data traffic increases in mega-events, the load on content servers as well as network nodes starts increasing. Insufficient processing capacities can result in failed client connections (Section 2.5.1.4). A simple, but expensive, solution is to continue adding more server or network nodes to cope with the increasing user demand, however, an efficient service must try to minimize system resources where possible. The proposed multicast-based solution for SDN (Chapter 4), evaluates the percentage load on the processing units when conducting large-scale evaluation and compares it with standard unicast-based solution.

## Summary

This chapter provided an insight into various concepts, standards and approaches used by live video streaming services over the Internet and in cellular networks. A review was presented for transmission modes, algorithms and models proposed for live video streaming in literature. A detailed description of various metrics to quantify network and video performance was provided. These metrics include Quality of Service (QoS)-based parameters, Quality of Experience (QoE) indicators and overhead costs for different network and system elements.

Various limitations of the state-of-the-art solutions and the challenges and issues that arise due to these limitations were identified. For IP multicast, the lack of features, such as user management, data control and privacy, inter-domain operability and transparency were pointed out. Lack of these features makes multicast incapable of streaming live content over the Internet. For cellular networks, the existing eMBMS resource management solutions were discussed. These solutions ignore the inter-dependence of user-grouping and SFN-clustering problems, yielding sub-optimal results. Also these models are either too complex to solve in real-time for a large number of users; do not consider multiple videos served by eMBMS at the same time; aim to maximize the network throughput instead of the application-level video bitrates; ignore the impact of eMBMS resource allocation on unicast users or; ignore the dynamic nature of cellular networks and mobile users.

# Chapter 3

# Tools and methodologies

## 3.1 Simulation vs emulation

Ideally, when bench-marking, evaluating or comparing the performance of new designs and algorithms, they would be tested on the actual hardware or the network for which they are developed. For example, an Over The Top (OTT) live streaming service may be tested by streaming content to actual users, connected to the Internet via an ISP. Testing on physical devices is more accurate, concise and user-specific. Such an approach can be adapted by large-scale services, e.g Netflix [18], by rolling out beta versions of the new features, updates or algorithms. However, for new innovative services or academic purposes, it might not always be feasible or cost-effective to deploy a real-world testbed for evaluation. For testing in a closed and controlled environment, two approaches are used:

- Simulation:A simulation is an abstract imitation of a system and is the most economical way of testing new algorithms. Simulation involves replicating the general behaviour of a system using trace-based or mathematical models of traffic, network behavior, channels and protocols. Most network simulators use discrete event-based simulation and may ignore the detailed internal functions of the original system.
- Emulation: An emulator works by duplicating almost every aspect of the original system. The emulation is efficiently a complete imitation of the system that operates in a virtual environment instead of the real world. Hence, by setting the virtual environment similar or close to live network, emulation can provide more accurate and concise results in comparison to

simulation.

Some systems or devices may be infeasible or too expensive to emulate such as eNBs in an LTE, in which case simulation may be the only viable testing option. The common tools for simulating LTE networks are OPNET [85] and NS2 [86]/NS3 [87]. NS2 and NS3 also provide network animators, (NAM [88] and *NetAnim* [89] respectively), that allow users to quickly gather large amounts of traffic details and visually identify patterns in communication. Such animation tools are useful for better understanding and demonstration of network protocols and algorithms. OPNET and NS3 provide the core components of an eMBMS architecure but do not include the physical-layer implementation of some features such as Single Frequency Networks (SFN). Therefore, to evaluate the performance of the proposed work in Chapter 6 and 7, a custom discrete event-based LTE physical layer simulator was developed (Section 3.4).

For developing,validating and testing SDN applications, Mininet [90] is a well-known and widely used emulator that emulates network topologies. Mininet creates a realistic virtual network, running real kernel, switch and application code, on a single machine. While Mininet provides native support for OpenFlow [44] protocols, its use is not limited to OpenFlow and can also run legacy network nodes or connect to physical devices through well-defined interfaces. APIs are built for communication between network nodes and SDN controller. Mininet does not have a network animator available, like *NetAnim* for NS3. A real-time network animator for Mininet is proposed in Section 3.2 and a testbed for evaluation of live video streaming is described in Section 3.3 using Mininet for creating network topologies.

## 3.2 MiniNAM: A network animator for visualizing real-time packet flows in Mininet

Mininet is one of the most well-known network emulator in research and academia. Although Mininet is capable of emulating both traditional as well as SDN systems, it does not provide a tool to visually observe and monitor the packets flowing over the created network topology. To address this the Mininet Network Animation Tool (MiniNAM) is designed and implemented. MiniNAM includes all the components required to initiate, visualize and modify Mininet network flows in real-time. MiniNAM provides a graphical user interface that allows dynamic

modification of preferences and packet filters: a user can view selective flows with options to color code packets based on packet type and/or source node. This establishes MiniNAM as a very powerful tool for debugging network protocols or teaching, learning and understanding network concepts. A number of sample use cases and examples are provided, on how to use MiniNAM to create networks and view the generated network flows with customized preferences. The tool has been released as an open-source software[1] and is being used at several universities and research groups.

### 3.2.1 Introduction

Network simulation tools such as NS3 [87] and emulation tools such as Mininet are widely used for the validation of new network protocols in research and learning or understanding network concepts in academia. One of the main challenges with simulators and emulators is to monitor the state of the network across what could be a large number of network entities and to analyze the complex and frequent message exchanges between these entities. Packet traces generated by simulators generally contain static outputs and lengthy log files, limiting the user's ability to comprehend the data. Similarly in emulated networks, statistics are generally gathered in raw form or at the end of the emulation, hiding the dynamics of the protocol behaviour. Network visualization and animation tools address such problems. A network animator, like NAM [88] for NS2 [86], allows users to quickly gather large amounts of traffic details, to visually identify patterns in communication, and to better understand causality and interaction.

Mininet creates a virtual network with fully operational nodes and allows interaction among these nodes, giving a more realistic and complete interpretation of a network in comparison to simulators like NS3. One of the key features of Mininet is its support for OpenFlow [44] and SDN systems [8]. Creating SDNs with Mininet can be as easy as running a single command. Mininet provides some visual tools like MiniEdit and Virtual Network Description (VND) [91] that further simplify the generation of network topology however, unlike NS3, Mininet does not provide a tool for visualization of traffic and packets flowing over the created network topology.

[1] `https://github.com/uccmisl/MiniNAM`

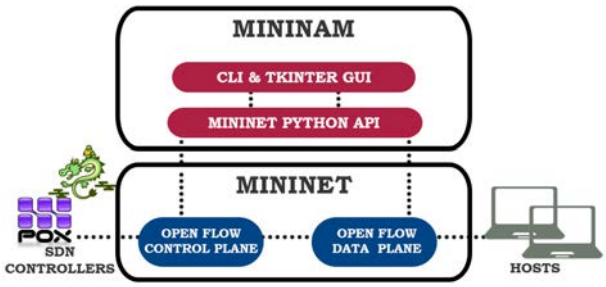



Figure 3.1: System design of MiniNAM

### 3.2.2 Design overview

The proposed animator, MiniNAM, is a Graphical User Interface (GUI) tool written with Tkinter and Mininet's Python API [92] (Figure 3.1). MiniNAM provides visualization of the main topology view, from which the user can derive a number of specialized views based on packet types and source or destination nodes. The GUI provides a means of viewing traffic flows, monitoring traffic patterns as well as the packet-level details necessary to design new protocols. MiniNAM has no additional dependencies and can run on any system that has Mininet installed on it. MiniNAM contains all the components needed to create a network, to capture traffic flows, customize visualization parameters and finally display the run-time progression of packets from source nodes to destination nodes over appropriate links. While the scalability of MiniNAM is dependent on the hardware on which it is being run, it can conveniently support networks of any size that can be created by Mininet on a certain machine. In our calculations, visualisation of the underlying Mininet topologies in MiniNAM generates little demand in overall CPU usage, but dependent on preferences and filters chosen, as defined in later sections, overall CPU loading does increase but not significantly.

Like Mininet, to create a network, users can execute the MiniNAM utility from the command line by passing various arguments. To simplify the use of MiniNAM, arguments are kept in the same format and structure as the Mininet *'mn'* utility. To provide support for custom defined topologies, MiniNAM provides a *'–custom'* argument which can be used to load a user-defined python script. MiniNAM also supports networks with multiple controllers to enable visualization of more complex networks. To create such a network, a custom python script can be written and switches can be attached to one or more controllers. On execution of a command, a GUI will load and present the created topology with options to drag nodes around in the created window. Positions (x,y) can also be defined for the nodes in the Python script.

Table 3.1: Preferences and filter options to customize flow display

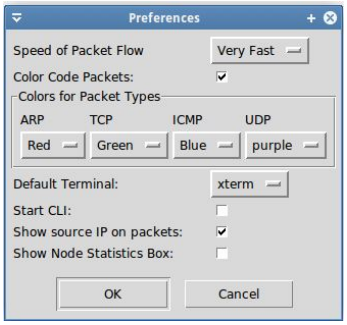
| **Preferences** | |
|---|---|
| Speed of packet flows | Flows can be viewed in real time or adjusted to a slower speed |
| Color code packets | If selected, flows from each source node will have a different color |
| Colors for packet types | Option to choose a different color for different packet types |
| Show source IP on packets | If selected, first and last octet of IP address of source node will be displayed on every packet |
| Show node statistics | If selected, node statistics will be shown instantly on mouse hover over any node |
| **Filters** | |
| Show Packets Types | Specified packet/frame types will be displayed. |
| Hide Packet Types | Specified packet/frame types not displayed. |
| Hide Packets from IP/MAC | Specified source IP/MAC not shown |
| Hide Packets to IP/MAC | Specified destination IP/MAC not shown |

A number of menus have been added to ease customization of the displayed flows:

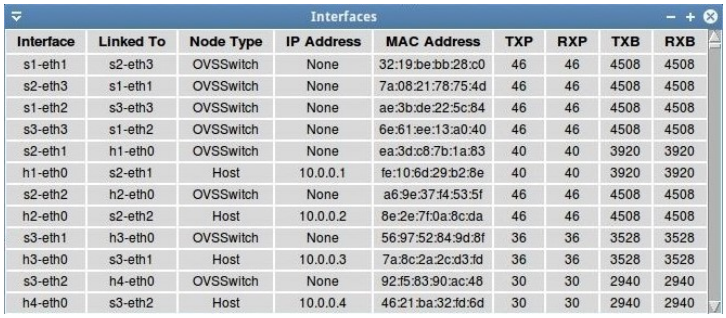
- *Preferences:* As illustrated in Figure 3.2a, this menu, selected from the *Edit* menu of MiniNAM, permits a means of customizing the view of flows that will be generated. Options in the *File* menu can be used to save these preferences and load them later for other emulations.
- *Filters:* As shown in Table 3.1, this menu offers a means of limiting the types of packet being visualised, based on packet types, and respective IP and MAC addresses.

Each of the respective parameters in the menus above are further detailed in Table 3.1. Further customization can be done by modifying the underlying source code. MiniNAM also provides other features through the *Run* menu that include: pausing the display of flows at a certain time, clearing the existing flows and viewing OpenFlow switch configurations. Once the traffic starts flowing, MiniNAM provides real-time information of the number of packets transmitted and received by every network node. Once this option has been turned on from the preferences menu, statistics will be shown when the cursor hovers over a node. In addition, statistical information of every interface in the network can be seen (Figure 3.2b) in the *Run* menu.

To provide interaction between elements in the network, each simulated node contains an option to open a terminal window by which additional network traf-




(a) Preferences




| Interface | Linked To | Node Type | IP Address | MAC Address | TXP | RXP | TXB | RXB |
|---|---|---|---|---|---|---|---|---|
| s1-eth1 | s2-eth3 | OVSSwitch | None | 32:19:be:bb:28:c0 | 46 | 46 | 4508 | 4508 |
| s2-eth3 | s1-eth1 | OVSSwitch | None | 7a:08:21:78:75:4d | 46 | 46 | 4508 | 4508 |
| s1-eth2 | s3-eth3 | OVSSwitch | None | ae:3b:de:22:5c:84 | 46 | 46 | 4508 | 4508 |
| s3-eth3 | s1-eth2 | OVSSwitch | None | 6e:61:ee:13:a0:40 | 46 | 46 | 4508 | 4508 |
| s2-eth1 | h1-eth0 | OVSSwitch | None | ea:3d:c8:7b:1a:83 | 40 | 40 | 3920 | 3920 |
| h1-eth0 | s2-eth1 | Host | 10.0.0.1 | fe:10:6d:29:b2:8e | 40 | 40 | 3920 | 3920 |
| s2-eth2 | h2-eth0 | OVSSwitch | None | a6:9e:37:f4:53:5f | 46 | 46 | 4508 | 4508 |
| h2-eth0 | s2-eth2 | Host | 10.0.0.2 | 8e:2e:7f:0a:8c:da | 46 | 46 | 4508 | 4508 |
| s3-eth1 | h3-eth0 | OVSSwitch | None | 56:97:52:84:9d:8f | 36 | 36 | 3528 | 3528 |
| h3-eth0 | s3-eth1 | Host | 10.0.0.3 | 7a:8c:2a:2c:d3:fd | 36 | 36 | 3528 | 3528 |
| s3-eth2 | h4-eth0 | OVSSwitch | None | 92:f5:83:90:ac:48 | 30 | 30 | 2940 | 2940 |
| h4-eth0 | s3-eth2 | Host | 10.0.0.4 | 46:21:ba:32:fd:6d | 30 | 30 | 2940 | 2940 |

(b) Statistics of all network interfaces

Figure 3.2: Example of some of the dialog boxes available in MiniNAM.

fic or application level programs can be executed. All generated traffic will be displayed according to user preferences. The user can also observe the behaviour of network failures on the emulated network, by using the link down option to simulate a broken link.

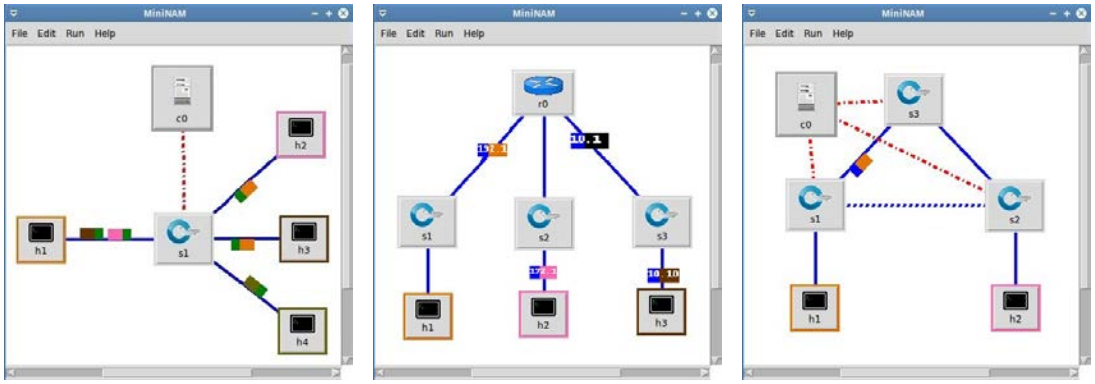
### 3.2.3 Applications of MiniNAM

MiniNAM is a useful tool in both education and research areas.

- *Education:* Teachers can use MiniNAM to animate networking principles in class or within a lab scenario. Students can use MiniNAM to compare and understand network protocols. By slowing down the speed of network flows, and viewing how packets traverse the network, students can understand the relationship between the control and data aspects of the network.
- *Research:* Researchers can use MiniNAM to investigate new networking concepts, debug real-life network applications, as illustrated in [93] where MiniNAM was used to show real-time video streaming, and do more advanced comparisons by using the network interface summary to check the number of packets and/or bytes crossing each network node and interface.

### 3.2.4 Demonstration overview

To demonstrate the working of MiniNAM, three pre-defined distinct network protocol examples are provided. It is illustrated how MiniNAM can be utilized to visualize a fully functioning network and associated packet flows. Also, and more importantly, the effectiveness of MiniNAM is shown, when the control of the network inefficiently manages the network, through the incorrect implementation




(a) Load balancing. *The traffic from three servers can be viewed as different colors travelling towards the client.*

(b) ICMP Packet visualization in NAT. *The change in IP address of packets can be seen once a packet crosses router 'r0'*

(c) Re-routing packets. *A broken data link (shown in dotted) results in packets taking a different route (s1-s2-s3).*

Figure 3.3: Three network protocol examples illustrating the generated topology and associated packet flows in MiniNAM.

of either protocols, nodes or flows. The three examples present a broad range of commonly used network protocols.

**Load balancing:** This example uses a Ryu [49] SDN controller to implement Server Load Balancing using an OpenFlow switch. A client requests traffic that can be served from three servers. The switch divides the traffic load among servers based on the logic implemented by the controller. This example[2] creates a topology with one switch and four hosts, as seen in Figure 3.3a. An echo server can be started on three of these hosts and if pings are sent from client, the packets can be seen distributed among the servers. This example illustrates an excellent use case where MiniNAM can make debugging easier when designing a new network protocol or algorithm.

**Network Address Translation (NAT)**: This example uses the simple linux-router example provided in Mininet to create a network with one NAT-enabled router connected to three switches. Three hosts, each on a different network communicate through the NAT-enabled Linux Router and if NAT is implemented properly, packets are routed correctly and this can be seen in MiniNAM. MiniNAM can display the flowing packets as well as the change in IP address of packets when they pass through the router. Figure 3.3b outlines how the real-time visualization of packets can make it easier for students to grasp network concepts.

[2] `https://github.com/OpenState-SDN/ryu/wiki/Server-Load-Balancing`

**Re-routing:** Taken from the Spanning Tree example in the Ryu controller, this example creates a network with multiple paths between hosts. If a path is broken and an alternate path is available, the controller tries to update the path. In this example, hosts are connected to each other through multiple paths. By simulating a broken link, the behavior of the routing protocol can be monitored and the response can be viewed in real-time, as shown in Figure 3.3c. This example demonstrates how MiniNAM can simplify the instantaneous analysis and debugging of a realistic network failure.

A detailed tutorial on how to build and run MiniNAM to replicate and repeat these examples is available [3]. Additional scripts are also provided, with inadequate code, which illustrate how MiniNAM can be used to determine when the code is logically incorrect or insufficient to create the required network functionality. The source code of MiniNAM is available on Github[4].

## 3.3 eSMAL: Evaluating SDN-based live streaming architectures

The rise of SDN presents an opportunity to overcome the limitations of rigid and static traditional Internet architecture and provide services like inter-domain network layer multicast for live video streaming. Over the past few years some SDN-based multicast frameworks [58, 60] have been proposed to offer Content Delivery Networks (CDN) with sufficient control and Internet Service Providers (ISP) with enough flexibility to realize a live Over The Top (OTT) streaming service. In this section, eSMAL is proposed, a modular platform to evaluate and compare SDN-based multicast architectures and algorithms for live video streaming and benchmark their performance against standard IP unicast.

eSMAL streams actual video content in High Definition (HD) quality over an emulated network and measures various network and application level metrics. For demonstration, the platform consists of two Graphical User Interface (GUI)s. A Panoramic GUI allows modification of various network and video parameters (Table 3.2), to generate variable evaluation scenarios and monitor the effect on output in form of graphs and live statistics. An Animator GUI displays the chosen network topology and shows traffic flows from video servers to the clients

[3] `http://www.cs.ucc.ie/misl/research/current/ivid_demo/mininam/`
[4] `https://github.com/uccmisl/MiniNAM`

Table 3.2: eSMAL GUI parameters to modify network and video settings

| | |
|---|---|
| **Network** | |
| Clients | Total number of clients in the ISP network that will request a video stream |
| Network Topology | Options include: Star, Mesh and Tree topology |
| Link capacity | The maximum link bandwidth for internal links of the ISP network |
| **Video** | |
| Video Servers | Number of CDN servers with video content to stream |
| Duration | Total duration (in minutes) for which the video will be streamed. |
| Quality | To choose High Definition (HD) or Standard (SD) video quality |
| Decoder | Options include: RAW - Do not display video at clients and H.264 - Displays video at the clients as they receive it |

in real-time. This GUI helps in visually identifying patterns in communication and better understand the working of the underlying protocols and algorithms.

For evaluation and comparison, eSMAL can be initiated without the GUIs, which enables light-weight mode and allows for large-scale experimentation. Due to its modular nature, eSMAL allows users of the platform to replace any or all modules with their own, to evaluate and compare the performance of their proposed design. To provide a fully functional testbed, eSMAL is pre-packaged with one to two implementations of all the components necessary to build an SDN-based live streaming service. A prototype of mCast (Chapter 4) and Danos (Chapter 5) was implemented on the eSMAL testbed for demonstration and evaluation of the proposed solutions.

### 3.3.1 Platform overview

The eSMAL platform for performance evaluation is based on an emulated testbed. Figure 3.4 illustrates the testbed setup.

#### 3.3.1.1 Network topology

Mininet is chosen for emulation and provides a means of creating a virtual network topology on a single computer. For this testbed, Mininet was setup on a laptop with Ubuntu 16.04. Two separate SDN-based domains for an ISP and a CDN are implemented and connected with each other over a high speed virtual

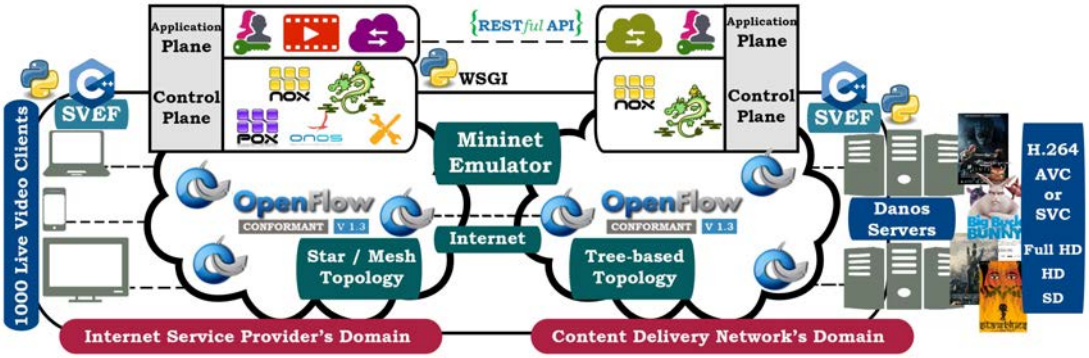



Figure 3.4: eSMAL setup and components. *The modular setup allows for an easy replacement or installment of custom components.*

link. For the CDN, a tree topology with a depth and fan-out equal to one is used. Tree topologies are commonly used in database centers to provide connectivity for content servers [36]. For the ISP domain, a sample star, mesh and tree topology are provided that can be chosen to monitor the behavior of the architecture with different underlying network topologies. In addition, real residential ISP topologies are also provided, from Topology Zoo database [94] including KREONET (approximately a STAR topology) and AT&T network (a MESH topology).

#### 3.3.1.2 Application and control plane

Each domain is managed by a separate SDN controller. As an example, The Ryu controller [49] was chosen for the testbed, over which control and application planes are implemented and OpenFlow [44] protocol v1.3 was used for the SDN southbound interface. Ryu controller provides a mechanism to implement application plane entities which were used to implement various components of mCast and Danos services. The agents in an ISP domain run as web servers with a Representational State Transfer (REST)ful API. The CDN can communicate with the ISP using HTTP requests that are received by the RESTful API. The messages from the CDN can contain details of actions or services that the CDN requires. To configure the network in response, services at the application plane of the ISP construct instructions and pass them down to the SDN controller using Web Server Gateway Interface (WSGI). The components in the control plane can then configure the nodes in the data plane accordingly.

#### 3.3.1.3 Video clients and servers

Although eSMAL is designed to evaluate live streaming services, the focus is on the delivery mechanism and network management and not the live encoding process. Therefore, to enable large-scale experiments, eSMAL pre-encodes videos and when streaming, transmits the videos as live content. Furthermore, when the video clients receive the transmitted stream, rather than live decoding, they save a log of the recieved content for post-processing. This allows scaling the number of clients that can run simultaneously on a single virtual machine-based emulated testbed e.g. when evaluating mCast, 1000 users were tested with each user running a video client application. eSMAL includes two different implementations for video clients and content servers.

The first implementation is based on C++ and adapted from Scalable Video Evaluation Framework (SVEF) [95]. SVEF provides servers and client implementation for evaluating Scalable Video Coding (SVC). For the client, the default SVEF implementation is used, however for the server SVEF implementation is modified to act as a live streaming server that is capable of dynamically streaming the content to more than one client as it receives content requests. Additionally, an API is implemented for interaction between content servers and the CDN application-plane agents. For encoding videos in SVC, eSMAL uses Joint Scalable Video Model (JSVM) [96]. As an example, two open source videos are provided, Big Buck Bunny (*bbb*) and Sita Sings the Blue (*sstb*). Nine minutes of each video are encoded at 1920x1080 HD resolution with a Group of Picture (GOP) size of 8 and a frame rate of 25 fps, yielding a bitrate of 2Mbps.

The second implementation is based on Python and is developed from scratch. The video clients support switching video bitrates during the stream. Unlike standard multicast-based live video clients, that subscribe to a single video quality, these clients are capable of bitrate adaptive streaming. It is assumed that the rate-adaptation algorithm is running in the network or at the content server and the client accepts the decision made by such algorithms. However, in case of failure or stalls, the client holds the capability to request a bitrate of its own choice, which will usually be a lower bitrate than before. The video streaming servers in this implementation run three threads: a **listening thread** to listen for content requests and establish a connection with clients; a **control thread** that uses an API to communicate with CDN application-plane and takes instructions or provides the requested client information and; a **data thread** that streams a video at different bitrates and serves each client at their maximum allowed

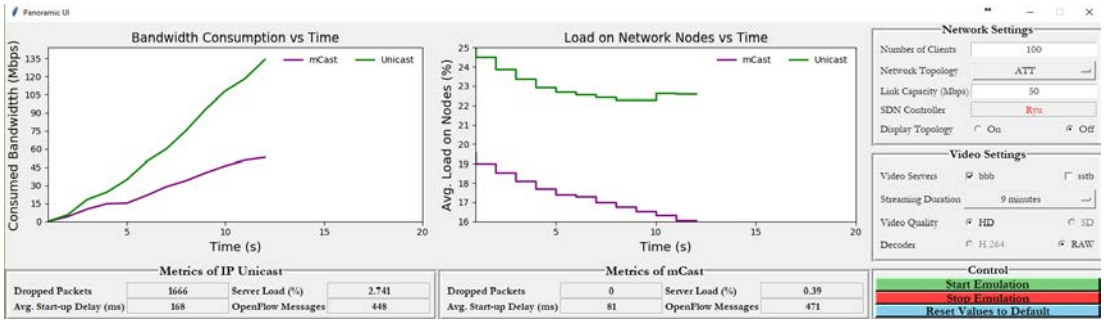



Figure 3.5: Panoramic GUI: *Utilized to modify network and video settings and view performance metrics in real-time.*

or requested bitrate. As an example, three open source raw videos, "Big Buck Bunny" (bbb), "Tears of Steel" (tos) and "Sintel" [21] are encoded using H.264 Advanced Video Coding (AVC) at three resolutions each. The chosen resolutions are Full HD 1920x1080, HD 1280x720 and SD 480x270 with a GOP size of eight and a frame rate of 24fps, yielding approximate bitrates of 4Mbps, 1.5Mbps and 400Kbps respectively.

### 3.3.2 Demonstration overview

The demonstration offers a means of modifying various network and video settings (Section 3.2), and monitoring the effects of these changes on the network and system metrics in real-time. Users can modify these settings through a Panoramic GUI (Figure 3.5). The GUI also offers a means of viewing performance metrics mandated by the input settings. These metrics are displayed in the form of graphs and statistics. The information shown includes bandwidth consumption in the ISP network plotted over time, average load on network nodes including switches and controllers plotted over time, number of dropped packets, average load on content servers in CDN, average start-up delays for the clients and total number of OpenFlow messages in the ISP network.

In addition, a second GUI (Figure 3.6) displays the chosen network topology including clients, servers and the SDN controllers. The traffic flows can be viewed in real-time as the packets travel from servers to the clients. Mininet Network Animation Tool (MiniNAM) is used for this purpose (Section 3.2). MiniNAM provides real-time animation of networks created by the Mininet emulator. A user can view selective flows with options to color code packets based on packet type and source node. For simultaneous comparison of a multicast-based live streaming service and IP unicast, an instance of MiniNAM is created for each

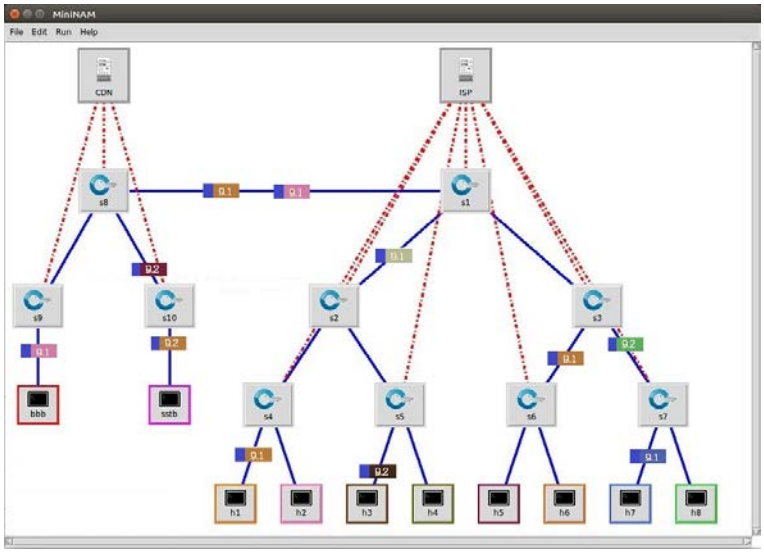



Figure 3.6: A snapshot of MiniNAM GUI displaying live video streaming: *A tree topology is used for both CDN and ISPs domain.*

approach in a separate virtual machine. With the packet level information provided by MiniNAM, users can visually identify patterns in communication and better understand the working of the streaming service.

## 3.4 eMBMS simulator and animator

The open-source NS3 [87] or the commercial OPNET [85] simulators provide LTE modules and include entities that Evolved Multimedia Broadcast Multicast Service (eMBMS) introduces in the LTE Evolved Packet Core (EPC). However, they lack various key features, such as ability to create clusters of Single Frequency Networks (SFN) or groups of users. As these features are essential for evaluating the systems proposed in Chapter 6 and Chapter 7, a custom discrete event-based simulator is proposed in this section. Furthermore, an offline animator, similar to *NetAnim* [89] in NS3, is developed for demonstration purposes.

### 3.4.1 Network topology

An eMBMS service area is simulated with Evolved-Node Base Stations (eNB) arranged in a hexagonal grid layout. The testbed can create a topology with up to five neighboring eNBs. This layout can represent a large shopping mall, e.g. as shown in Figure 3.7. Users can be spread across the service area with either uniform distribution or normal distribution. A normal distribution represents

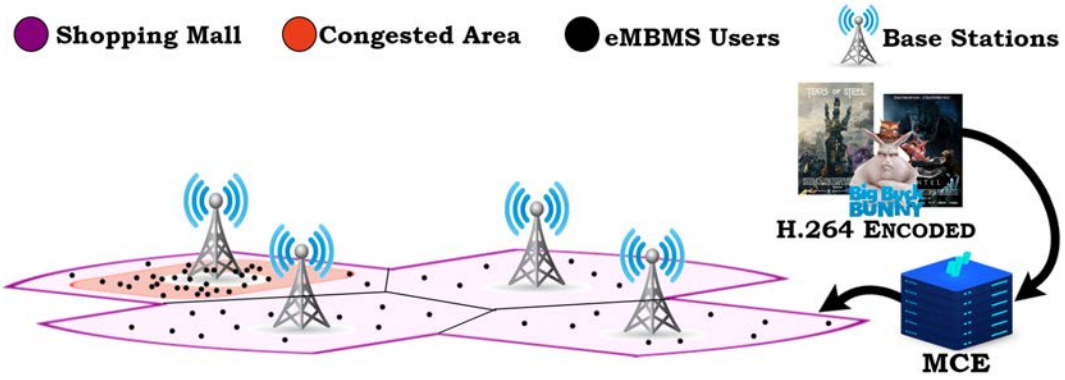



Figure 3.7: An example of eMBMS hexagonal grid layout. *The area represents a shopping mall with four eNBs.*

scenarios, e.g. stadiums, where most of the users are concentrated around one area. A uniform distribution represents scenarios, e.g. shopping mall, where users are more likely to be located evenly in the service area. For user mobility, a Random Way Point (RWP) model [97] is included, which can be tuned to replicate the behavior users walking or resting in a mall.

### 3.4.2 Physical layer parameters

On the user device, Reference Signals Received Power (RSRP) from each eNB is measured and Signal to Interference Noise Ratio (SINR) is calculated. Users then report their achievable Modulation and Coding Scheme (MCS) from different possible SFN clusters. To simulate RSRP, a Log-normal shadowing path-loss model is considered with path-loss exponent value denoted by $n$. eNBs are assumed to transmit signals at power $P$ and based on the user's distance ($d$) from an eNB, the RSRP, denoted by $R$, is calculated as [98]:

$$R_{dB}(d) = P_{dB} - L_{dB}(d_o) + 10 \cdot n \cdot log_{10}\left(\frac{d}{d_o}\right) + \chi, \tag{3.1}$$

where $\chi$ is a zero-mean Gaussian distributed random variable. $d_o$, the reference distance, is set to 0.01m and $L$ is the free-space path loss at $d_o$. The transmit power $P$ of eNB is in dB and the RSRP produced is also in dB. The SINR that a user can achieve from an SFN cluster $c$ can then be calculated as [98]:

$$SINR(c) = 10 \cdot log_{10}\left(\frac{\sum_{b \in c} R_{Watts}(b_d)}{\sum_{b \notin c} R_{Watts}(b_d) + Noise}\right), \tag{3.2}$$

where: $b \in c$ represents the eNBs in the eMBMS service area that are part of the SFN cluster $c$ and add up to the user's signal quality; $b \notin c$ represents the eNBs in the service area that are not part of the SFN cluster $c$ and cause interference for the users; $b_d$ is the distance of user from eNB $b$; $Noise$ is the white noise power density that the user experiences and; RSRP $R$ is measured in Watts.

Based on the user's attainable SINR from each cluster the MCS is calculated using Additive White Gaussian Noise (AWGN)-based Block Error Rate (BLER) vs SINR curves from LTE System-Level Simulator [99]. The AWGN curves from the ns-3 simulator [87] are also included. The target BLER is set to 1% in eMBMS [11], contrary to 10% in unicast, as there are no physical layer retransmissions and a lower percentage ensures higher probability (99% in this case) of successful delivery. The Multicast Coordination Entity (MCE) can then use these MCS values to choose the best bitrate or network configuration for users. In the simulation testbed, these values are passed as an input to the network configuration function, that computes the best solution and logs the output.

For video content, traces of multiple videos are generated based on H.264 encoder at multiple bitrates, and a Forward Error Correction (FEC) of 10% is assumed. The trace files consist of frame-level details with each entry representing the frame number, timestamp, size in bytes and the frame type (e.g. I, P or B). Based on the solution generated by the network configuration function, each user runs a greedy algorithm as the video client, and determines the best bitrates that it can receive at each instance of the streaming duration. It then creates a receiver-specific video log file for post-processing. When switching between bitrates, to ensure a smooth switching experience, the client waits until the end of a GOP to subscribe to the new bitrate. Finally, the user logs are processed to calculate and evaluate various performance metrics (Section 2.5).

### 3.4.3 Animation GUI

The Graphical User Interface (GUI) for the animator is built on Python's Tkinter library [5]. The configurable settings provided in the GUI are: a play button to pause or resume the animation: a time-slider that can be placed at any particular time of the simulation and; a speed knob to adjust the playback speed of the animation. In addition, information boxes are included to display various network and user statistics. The inputs to the animator are the network configuration log

[5] `https://docs.python.org/2/library/tkinter.html`

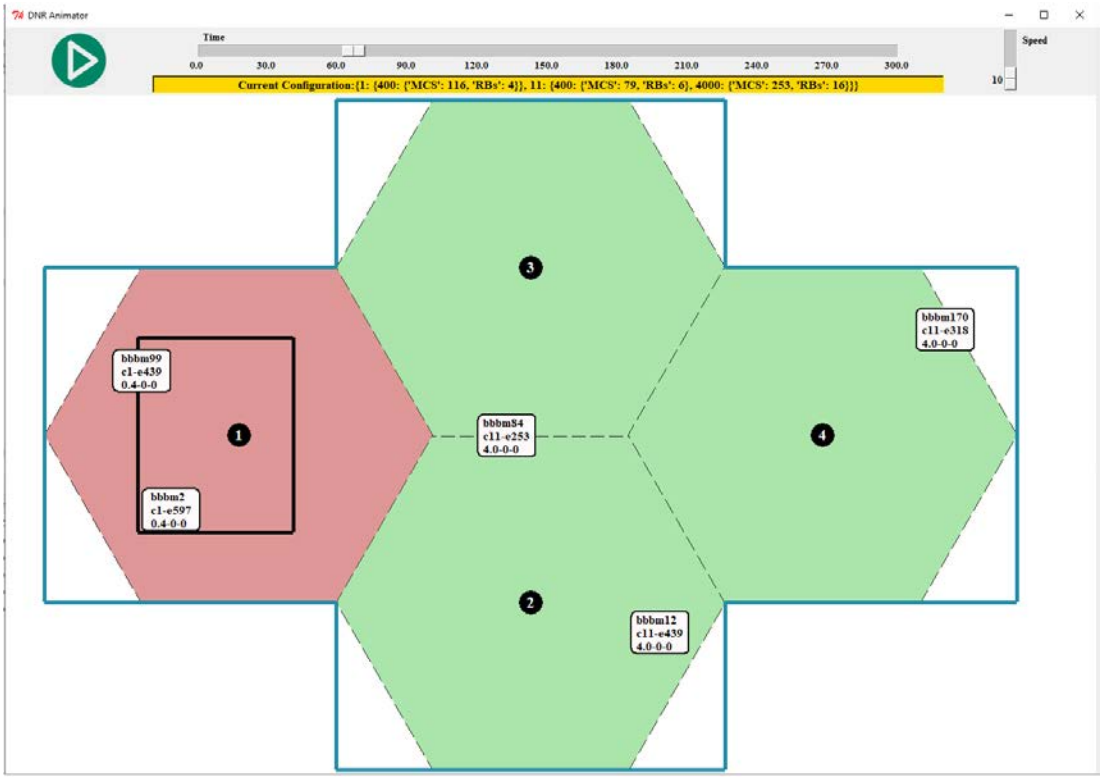



Figure 3.8: A snapshot of eMBMS animator GUI for shopping mall. *5 users are displayed and eNBs are divided into two clusters.*

files and user-specific video trace files.

Figure 3.8 shows a snapshot of the animator. The topology displayed in the figure is of a shopping mall (Figure 3.7) with four eNBs. The GUI helps in visualizing the network topology, user mobility pattern, SFN cluster configuration, bitrates being transmitted, the MCS used to transmit each bitrate and number of Resource Blocks (RB) allocated to each bitrate. In addition to network details, user-specific information is also displayed, including, user ID, the current cluster and bitrate of user, current channel condition (MCS), current streaming bitrate, total lost frames and total switches experienced by the user so far.

## Summary

This chapter explained various testbeds and tools that were developed for performance evaluation of the proposed algorithms that are presented in the following chapters. For evaluating eMBMS, a discrete event-based simulator was designed with commonly used physical layer LTE parameters [100] and traces of real video content. For evaluating SDN-based proposals, a modular and scalable emulator

was built with all the essential components in both CDN and ISP domains. Multiple videos were encoded at multiple bitrates that can be streamed from content servers located in a CDN domain to large number of users located in an ISP network.

# Chapter 4

# mCast: Enabling inter-domain network-layer multicast for live streaming services

## 4.1 Introduction

Content providers usually have limited resources and rely on Content Delivery Networks (CDN), to distribute streams globally. More than 60% of video traffic on the Internet passes through CDNs [9]. To save network and system resources, instead of IP unicast, a CDN can use either IP multicast or Application-Layer Multicast (ALM) to deliver streams to clients located in the networks of Internet Service Providers (ISP). However, multicast is rarely used in the Internet due to its rigid and static nature and other limitations specific to the aforementioned multicast approaches [6, 40].

Software-Defined Networking (SDN) possesses features that are non-existent in a traditional IP network and can be utilized to deploy multicast over the Internet. In this chapter, a novel architecture, mCast, is proposed for live streaming, that merges the flexibility and control of SDN with resource efficiency of multicast to reduce inter-domain and intra-domain traffic for both ISPs and CDNs. The modular architecture of mCast reduces the cost and complexity to implement network-layer multicast for ISPs and provides a dynamic and scalable mechanism for multicast tree construction in real-time. CDNs are given full control of their clients and all the information necessary for management and billing.

The clients do not need to be modified because mCast installs rules on the last hop to convert the stream back to IP unicast, making the delivery transparent to client devices. These rules also help measure the amount of traffic that goes to each user. The ISP can use this information to charge users based on their data plans. As such, mCast solves another main problem of IP multicast i.e. inability to bill users in a multicast group based on their individual billing plans.

mCast employs agents in both the CDN and the ISP domain. These agents are responsible for communication with each other as well as the control plane of SDN in their respective domains. As ISPs are economically driven and will charge CDNs for availing of the mCast service, it is important for a CDN to know the exact cost to serve a certain video stream. A CDN can choose to switch from IP unicast to mCast when doing so will reduce the overall cost. A decision model is formulated to help CDNs in making the switching decision.

The decision model not only identifies various cost factors but also presents them in quantifiable mathematical equations and if a CDN chooses to use mCast, this model will help the CDN to decide when switching to mCast will reduce the total cost of serving a stream to active clients. Once the decision to switch the transmission mode has been made, the CDN mCast Agent provides the ISP mCast Agent with a list of its active clients. An ISP uses this information to construct a dynamic multicast tree using an extension of Dijkstra's Algorithm (Section 4.2.1.5) and installs forwarding rules in run-time.

For CDNs and ISPs, keeping their network infrastructure private may be of commercial importance and they tend not to reveal information such as network topology, available bandwidth, or routing paths. The communication framework of mCast is designed in a way that no such information needs to be disclosed. Both ISPs and CDNs can manage their clients and network in their own way and just share the identity of clients to be served with mCast. This resolves any concern that a CDN might have in terms of user or data privacy.

For evaluation, a prototype of mCast is implemented on a large scale testbed, eSMAL (Section 3.3), with content servers streaming multiple HD videos. An extensive evaluation was performed to show the feasibility, scalability, robustness and gains of mCast. mCast is compared with standard IP unicast and results showed that in similar network conditions, mCast not only can save significant network and system resources for ISPs and CDNs, but also delivers a better quality of video to clients, with lower start-up delays and no dropped packets.

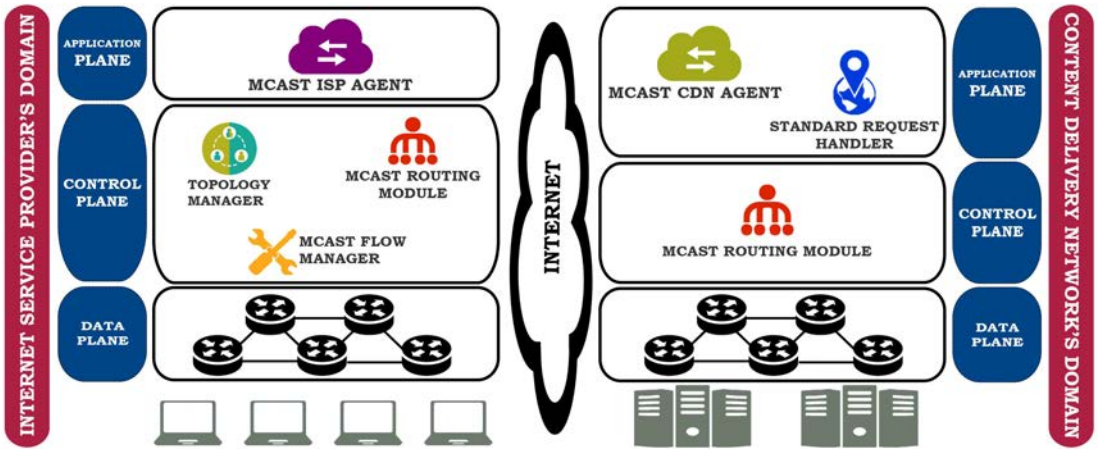



Figure 4.1: Architecture and components of mCast

## 4.2 Architecture and components

The proposed architecture for live streaming focuses on two main goals: reducing the resource consumption in ISPs and CDNs, and providing CDNs with full control over their clients even when using multicast. Both ISP and CDN domains are assumed to be SDN-enabled. This assumption is justified as SDN is already transporting 23% of traffic in data centers, growing to 44% by 2020 [9]. A 2016 survey [101] indicates that 75% of the respondents had either implemented SDN in their network or were planning to do so in the near-future.

Generally, ISPs and CDNs implement SDN for various use cases such as efficient resource utilization and ease of management. In mCast, SDN's flexible data forwarding and in-network editing features are exploited. This is essential for an ISP to take advantage of the routing capabilities of SDN. For a CDN, the SDN controller can be replaced by a server, capable of communicating with an ISP and managing clients on its own. However, the use of SDN for the CDN domain in mCast is to emphasize and show how two controllers can communicate and manage their underlying networks more efficiently.

### 4.2.1 Architecture overview

Figure 4.1 illustrates the key mCast architectural elements in the application, control, and data planes [8]. At the data plane, standard SDN switching nodes are used to forward user data within ISP and CDN networks, according to mCast control plane functions. mCast employs agents in both CDN and ISP domains to carry out application plane functions. The role of these architectural elements

is detailed below.

#### 4.2.1.1 mCast CDN agent

This component has three main functions: monitor the client requests, classify clients and trigger mCast. In a standard IP unicast streaming service, a request handler receives channel requests, authenticates clients and responds with the IP and port address of the streaming server. A client can then send a content request to the streaming server.

The mCast CDN agent extends a standard request-handler in traditional CDN systems to enable efficient content delivery using multicast. A standard request handler authenticates content requests and forwards legitimate content requests to their corresponding servers according to the CDN policy. mCast extensions include request classifier and multicast management functions. The request classifier identifies users located in a single ISP and watching the same content. Such classification can be performed using geo-location databases. Once a group of clients satisfy a pre-specified criterion (discussed in Section 4.3), the request classifier triggers the mCast handler to start multicast operation.

The multicast management functions include interfacing with: the mCast ISP agent to request mCast service and share client details; mCast CDN routing module to install routing entries for multicast and; mCast streaming server to improve the resource utilization, by aggregating a group of flows into a single flow as detailed below.

#### 4.2.1.2 mCast ISP agent

ISP plays a passive role in mCast, as in, it does not trigger the mCast request. To be the trigger, an ISP would need to perform deep packet inspection and decoding to identify what streams are watched by clients and whether they are served from the same CDN. As CDNs are a better judge of their clients, it is simpler, less resource consuming and less prone to privacy violation, if a CDN triggers mCast request.

The mCast ISP agent represents the application plane module at the ISP and holds key importance in the mCast architecture. It performs two main functions, interfacing with the CDN and orchestrating multicast operations in the ISP. It

receives session aggregation requests from the CDN including source and destination addresses and port numbers for every user session intended for multicast. Note that such information is used to locate the target users in the ISP network and ensure transparency for end clients.

Upon reception of such requests from the CDN, the mCast ISP agent first creates an identifier for the mCast stream composed of an IP address and a port number, denoted as $V_{(IP,\ Port)}$. Then, it instructs the mCast ISP routing module to create a multicast tree for the intended users starting at the ISP gateway. Once a tree is successfully constructed, ISP agent shares $V_{(IP,\ Port)}$ with CDN agent which uses it to configure CDN network for mCast delivery.

As users join or leave the session, the mCast CDN agent informs the ISP agent to dynamically update routing entries in the ISP forwarding nodes. Such interaction is not possible in traditional network nodes due to the lack of SDN-like centralized control.

#### 4.2.1.3 mCast enabled streaming server

This server typically listens for content requests and streams the requested content using RTP UDP/IP unicast sessions. To support mCast, the server implements an API to communicate with the mCast CDN agent. Over such an interface, the server would be instructed to pause transmission for a group of sessions and alternatively send the content to the provided destination address and port $V_{(IP,\ Port)}$. To avoid packet losses, the new connection is activated before terminating the old unicast sessions. While this sequence ensures no packet loss, it may lead to duplicate packets at the client. However, these packets would be ignored by the client after inspecting the RTP headers.

#### 4.2.1.4 mCast CDN routing module

The routing module is programmed to forward content requests to the request handler for authentication. Once authenticated, the routing module is provided a target server to forward connection requests using its predefined routing policy (e.g., shortest path or least loaded path). In the presence of an mCast agent, it is consulted before proceeding with the default routing. If the multicast criteria are satisfied, multicast routing is activated and the new content request is served through the new multicast stream.

#### 4.2.1.5 mCast ISP routing module

This module identifies the routes of different flows from the ISP gateway to their end points. Due to the global view of the SDN controller, a typical topology manager in SDN is aware of all of its network nodes and clients. The mCast ISP Routing module probes the topology manager to obtain a graph representation for the ISP network. It also communicates the estimated routes to the flow manager that interfaces with the switching nodes. The ISP routing module implements the ISP routing policy and is extended by an additional function to support multicast routing, which is triggered by the mCast ISP agent. The mCast ISP agent also dynamically instructs the routing module to update video multicast trees as clients leave and join.

The routing tree construction problem for mCast is a well-known Steiner tree problem in graphs [102]. Although the main focus of the work is to present the architecture and components essential for realizing a live streaming service using inter-domain multicast, an extension of Dijkstra's algorithm is developed to implement the mCast ISP Routing Module. As the architecture of mCast is modular, to implement any other tree construction algorithm [53, 54], the mCast Routing Module can be replaced with that algorithm without modifying rest of the modules in the architecture.

In the extended Dijkstra's algorithm, when the mCast ISP Routing Module receives a request from an mCast ISP agent, it calculates the shortest path for the first client. Then it sets the weight of all involved edges to zero before calculating the shortest path for the next client. This prioritizes the used edges and paths over others and reduces link stress in the network. The process is repeated until a path is determined for all the clients. This information is then passed on to the mCast Flow Manager.

#### 4.2.1.6 mCast flow manager

A Flow Manager in an ISP installs rules on SDN-enabled switches based on the information that it receives from the Routing Module. The mCast Flow Manager installs multicast entries in network nodes with higher priority than IP unicast, ensuring that clients are served with mCast whenever possible. In addition, the mCast Flow Manager installs transparency rules on the egress switch. Before forwarding a packet to the client, this rule modifies the $V_{(IP,\ Port)}$ to the IP and port address of the video client, so the client receives the packet just as it would

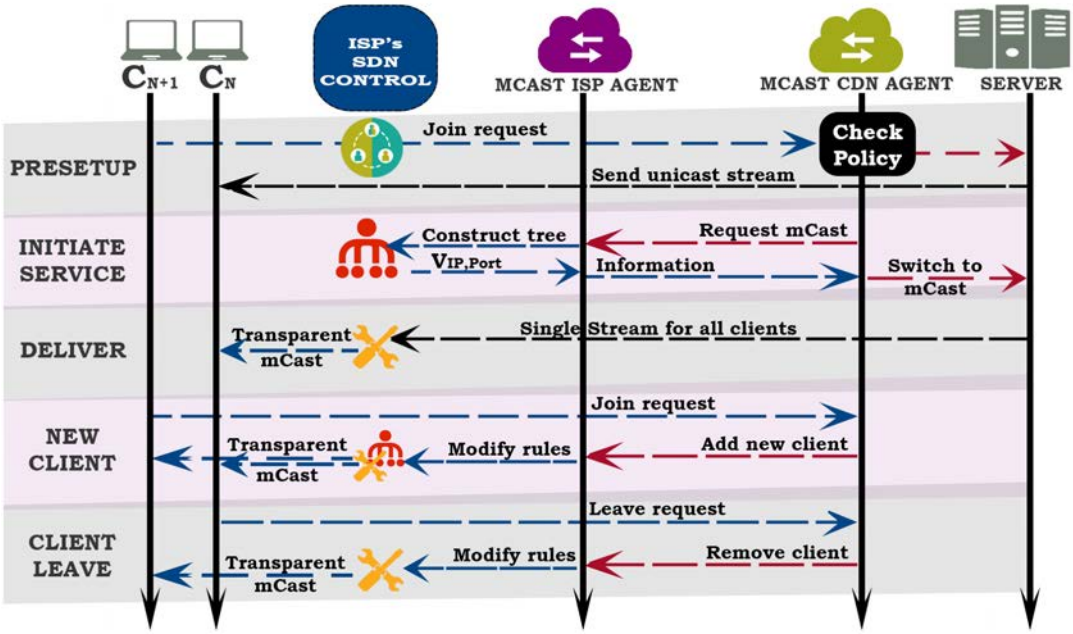



Figure 4.2: Important message exchanges to establish and serve mCast.

in IP unicast, hence the transparent delivery. These rules also help the ISP in measuring the amount of traffic that goes to each user. An ISP can use this information to charge users based on their individual billing plans.

### 4.2.2 Functional description

Figure 4.2 illustrates the exchanged message-sequence between different components to setup mCast and deliver content through multicast. For simplicity of illustration, the figure provides an overview of mCast operations for a single ISP and a single CDN domain. The same operations would be applied to clients belonging to different ISPs or multiple CDNs served by a single ISP.

#### 4.2.2.1 Pre-setup

The decision model (Section 4.3) is used as the criteria to initiate mCast. $C_N$ represents the client that satisfies the decision model criteria. Any client that joins the stream before $C_N$, is served with IP unicast. For these clients, the mCast CDN routing module installs unicast rules for the client and forwards the request to the streaming server which sends a unicast stream to the client.

When a content request is received from $C_N$, along with sending a unicast stream to $C_N$, CDN requests mCast service from ISP and sends a list of clients to be

served with mCast. The list includes full tuples i.e. server's address $S_{(IP,\ Port)}$ and the client's address $C_{(IP,\ Port)}$. The mCast ISP agent receives this request and assigns a virtual IP and port $V_{(IP,\ Port)}$ to the video stream. ISP can choose an address from a pool of IPv4 addresses that it reserves for the mCast service. It can then use port numbers to distinguish streams, allowing up to 65535 streams per address.

#### 4.2.2.2 Registration and delivery

The list of clients and $V_{(IP,\ Port)}$ are passed to the mCast ISP routing module. The mCast Flow Manager then installs rules on the switches to: forward the traffic coming for $V_{(IP,\ Port)}$ towards the switches in mCast tree and; on egress switch, replace $V_{(IP,\ Port)}$ with $C_{(IP,\ Port)}$ and the source with $S_{(IP,\ Port)}$.

The SDN controller then informs the CDN mCast agent which sends a message to the streaming server to terminate IP unicast streams of accepted clients and replace them with a single stream destined for $V_{(IP,\ Port)}$. As the ISP network is all set-up for mCast, when the single stream from the server enters the ISP ingress switch, it is sent only once on every link on the tree until it reaches all the clients.

#### 4.2.2.3 Client join and leave requests

When the CDN receives a new content request from a client, instead of sending that request to the server, it is first sent to the ISP. The port number $S_{(IP,\ Port)}$ is set as the one at which the client made the request i.e. the listening port of the server. The ISP adds this client to the mCast tree by installing or modifying forwarding rules and the client starts receiving the stream instantly and transparently.

When a client decides to switch channel or terminate the service, it sends a leave request to the CDN. The CDN mCast agent receives this request, updates the client state and also informs the ISP. The mCast Routing module in ISP updates the mCast tree by traversing backwards from the egress switch and removing mCast entries, until it reaches a node that serves more than one client. As clients joining and leaving can result in un-optimized multicast paths, a global update of mCast tree can be scheduled, to optimize the routing paths based on the currently joined clients.

### 4.2.3 Performance analysis

The performance of mCast is measured in terms of the consumed network and system resources. For IP unicast, the bandwidth consumed in the network increases linearly with every new client, regardless of the number of shared links in the network. mCast uses network layer multicast and avoids duplication of traffic over any link. As the number of clients increases in the network, the amount of bandwidth consumed at a link stays constant. This reduces the amount of bandwidth consumed when one or more shared links exist in the network. This also avoids creating bottleneck links when a large number of clients share a link, as the bandwidth consumption per link is independent of the number of clients in mCast.

The system resources consumed in an SDN network include the flow entries or rules installed at switches and, the messages shared between SDN controller and the switches to install these flow entries. For a single live stream, mCast installs only one entry per switch. When a new client joins mCast, the SDN controller sends a message to the switches on the path to this client and further actions are added to the flow entry. Depending on the approach used by an ISP, mCast offers savings in system resource consumption, for example when an ISP implements service differentiation [103]. Instead of having one rule at entry and exit point of each tunnel for a live streaming service, mCast will install only one rule per switch in the network, saving the system resources.

Other ISPs will save system resources by mCast, as the number of packets to be processed and forwarded by a switch will be very low in comparison to IP unicast and hence the processing cost and time will be reduced. This is reflected in the evaluation results (Section 4.4), where it can be seen that for a large number of clients, where IP unicast overloads a switch's processing capability, mCast maintains a very low CPU consumption.

## 4.3 Cost-based decision model

As ISPs are economically driven and will charge CDNs for availing of the mCast service, it is important for a CDN to know the exact cost to serve a certain video stream with unicast or mCast. In this section, various cost factors are identified and an optimized mathematical cost model is presented for the decision of switching from unicast to multicast for a specific stream. The model is a

distinct complementary contribution: mCast does not rely on it however if a CDN chooses to use mCast, this model will help the CDN to decide when switching the transmission mode to mCast will reduce the total cost to serve that stream.

Two key factors that add up to the cost for a CDN serving a channel to $N$ clients are: the load on servers or power consumption $P(N)$, and the outgoing traffic or bandwidth consumption $B(N)$. For $P(N)$, a Power model from [104] is adapted, which states a linear increase in power consumption with the increasing number of clients. The maximum power consumed by a server when fully utilized is represented by$P_{\max}$. An idle server consumes approximately 70% of $P_{\max}$ while the remaining 30% increases linearly with the number of clients. From this model an equation is derived for the total power consumed by all the servers to serve $N$ clients that are watching a particular channel. If one CDN server can support $N_o$ number of clients then to serve $N$ number of clients with IP unicast $P(N)$ will be:

$$P(N) = P_{\max}\left(0.7\left\lceil\frac{N}{N_o}\right\rceil + 0.3\frac{N}{N_o}\right) \tag{4.1}$$

As an example, if there are 1200 clients ($N$) watching a channel and one server can support 500 clients ($N_o$), then three servers are needed, where two servers are fully utilized and the third one consumes 70% of $P_{\max}$ for running idle and an additional 12% to serve 200 clients. The total power consumed can be found by substituting $N$ and $N_o$ in Equation 4.1, yielding $P(N) = 2.82P_{\max}$.

$B(N)$ is the Internet transit cost that a CDN will have to pay to the Internet Exchange Point (IXP) for serving a channel to $N$ clients. Transit volume is the amount of traffic that goes out of a network domain. Internet transit is typically metered and priced in $/Mbps. The industry standard to measure transit cost is the 95th Percentile method [105]. In this method, a network can avail of pricing discounts by relying on commit volume. Commit volume is a certain volume of traffic that network domains can agree in advance to pay for, regardless of the actual volume that they consume. Let $V_T$ be the 95th percentile transit volume and $V_C$ be the commit volume that the CDN committed to an IXP, Norton [106] models the transit cost as:

$$\text{Transit cost} = \max(V_T S_T, V_C S_C), \tag{4.2}$$

where $S_C$, the single unit price for commit volume is lower than $S_T$, the single unit price for transit volume. The volume consumed by $N$ clients is represented

by $V(N)$ and hence $B(N) = \max(V(N) \cdot S_T, V(N) \cdot S_C)$.

If a CDN chooses the first method, i.e. transit volume, then $V(N)$ increases linearly with the number of clients and can be calculated as $V(N) = \alpha N B_i$, where $B_i$ is the average bit-rate of channel $i$ and $\alpha > 1$ accommodates for the variable video bit-rate and helps avoiding large queuing delays. With $V_C = 0$, $B(N) = \alpha N B_i S_T$.

If a CDN uses chooses the second method i.e. commit volume to an IXP, then the cost for $V_C$ is a fixed value and does not vary with the number of active clients. To get an estimate of the cost for $N$ clients watching a single channel, the cost is divided equally among all active clients. Let $X_i$ be a client watching a channel $i$ at the average bit-rate $B_i$, then $V(N) = \frac{NB_i}{\sum X_i B_i}$. This represents the portion of the cost for $N$ clients. With $V_T = 0$, $B(N) = \frac{V_C S_C N B_i}{\sum X_i B_i}$. Combining these two results and substituting in Equation 4.2 gives the bandwidth consumption:

$$B(N) = \max\left(\alpha N B_i S_T, \frac{V_C S_C N B_i}{\sum X_i B_i}\right). \tag{4.3}$$

In general, CDNs use the second method i.e. commit volume to an IXP as this is more predictable. The total cost that a CDN will incur to stream a channel to $N$ clients using IP unicast is denoted by $U(N)$ and equals to:

$$U(N) = P(N) + B(N).$$

Substituting Equations 4.1 and Equation 4.3 gives the total cost incurred by a CDN to serve $N$ clients using IP unicast:

$$U(N) = P_{\max}\left(0.7\left\lceil\frac{N}{N_o}\right\rceil + 0.3\frac{N}{N_o}\right) + \frac{V_C S_C N B_i}{\sum X_i B_i}. \tag{4.4}$$

Now the cost to serve a single video to $N$ clients using mCast is calculated. For mCast, the outgoing traffic from a CDN server and the load on it is independent of the number of clients and is equal to the cost of one client i.e. $U(1)$. In addition, a CDN will have to pay a certain charge to the ISP for availing of the mCast service. An ISP can charge a CDN with either a variable-rate based on the number of clients that joined a particular stream or a flat-rate based on an estimated value. This charge provides an extra incentive for ISPs to use mCast

because in addition to saving resources, an ISP can increase its revenue earned through mCast. This charge should represent the cost that an ISP incurs to provide mCast service and is mainly based on forwarding data to the clients that are in the multicast tree.

Because mCast avoids packet duplication on any link in the ISP topology, once a link is added to the tree to serve a client, the cost to serve any additional clients over that link is essentially zero. This means that once all the links in the ISP topology have been traversed and added to the multicast tree, an ISP will incur no additional cost for the increasing number of clients. Probabilistically, such a situation occurs with just a fraction of the total number of clients that an ISP can serve. As mentioned in [107], an ISP that can serve 100,000 clients can have all of its links, added in the multicast tree, with just 500 randomly distributed clients. Hence, an ISP can charge CDN a geometrically decreasing cost for every client that joins the stream. Therefore the total cost $M(N)$ that a CDN will have to pay for $N$ clients using mCast will be:

$$M(N) = U(1) + C\frac{1 - r^N}{1 - r}, \tag{4.5}$$

where C is an initial cost that an ISP charges CDN and $r$ is the ratio for decreasing cost of every new client. Note that the small increment for every new client, regardless of no increase in the link cost, is justified by other minor costs that an ISP incurs such as number of forwarding entries and actions taken at the network nodes; managing clients and multicast trees and; interacting with the CDN.

A business model of a CDN for live streaming involves individual clients where serving each client incurs some cost on the CDN as discussed above. To minimize the total cost for the duration of the stream, a CDN should switch the transmission mode to mCast when the cost for mCast becomes lower than IP unicast. i.e.

$$M(N) < U(N). \tag{4.6}$$

For stability, only those clients should be considered for initiating mCast that have been watching a particular channel for a certain amount of time. The clients with very dynamic behavior such as the ones which are switching channels should be ignored. This will also keep the cost to minimum when an ISP is charging CDN with variable-rate based on the number of clients that joined a particular stream.

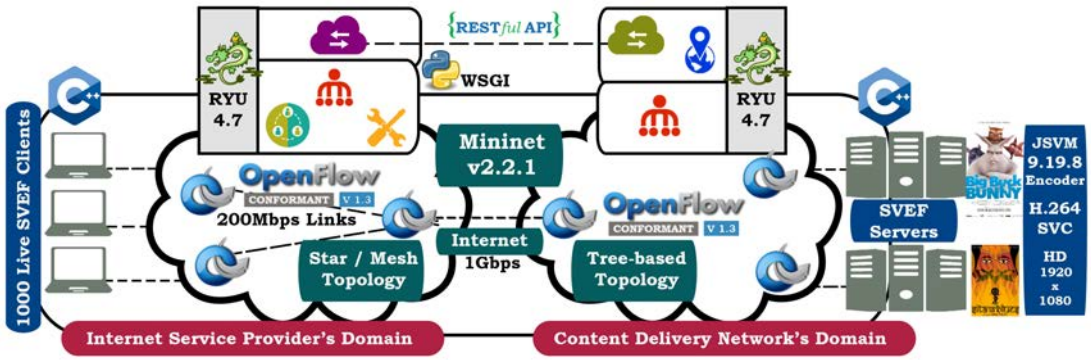



Figure 4.3: Experimental testbed for evaluation of mCast

## 4.4 Performance evaluation and comparison

The performance evaluation is based on a prototype implementation of mCast on top of the eSMAL testbed (Section 3.3). Figure 4.3 illustrates the experimental setup. The two separate SDN-based domains of CDN and ISP are connected over a high speed 1Gbps Internet link. For the CDN, a tree topology is used, with a depth and fan-out equal to one. For the ISP domain, two different residential ISP topologies from the Topology Zoo database [94] are used, including KREONET (approximately a STAR topology) and AT&T network (a MESH topology), to show that the potential of mCast is independent of the underlying network. Since Topology Zoo information does not include link rate, the internal link bandwidth is set to 200 Mbps.

Ryu, a Python-based SDN controller, is used to implement the mCast components in the control and application plane. Two open source videos "Big Buck Bunny" (*bbb*) and "Sita Sings the Blue" (*sstb*) are streamed from the CDN using the C++ based modified-SVEF live servers. Nine minutes of each raw video is encoded using the JSVM framework [96] at 1920x1080 HD resolution with a GOP size of eight at a frame rate of 25 fps, yielding an approximate bitrate of 2 Mbps. The SVEF clients, postpone video decoding to a post processing phase, which helped increase the number of clients to 1000. Each client was randomly attached to one of the ISP switches and requested one of the two available streams at a random time according to a uniform distribution.

mCast is compared with IP unicast to set a benchmark for various performance metrics. The key performance metrics include link utilization and dropped network packets. The percentage of decodable frames is also calculated along with start-up delays as an application layer metric. The number of additional OpenFlow rules and messages generated are captured, to analyze the cost of using

Table 4.1: Results of mCast experiments for STAR topology

| Metrics | Clients | 200 | 400 | 600 | 800 | 1000 |
|---|---|---|---|---|---|---|
| Failed Client Connections | Unicast | 0 | 0 | 56 | 470 | 844 |
| | mCast | 0 | 0 | 0 | 0 | 0 |
| Network Packet Losses (%) | Unicast | 0.33 | 3.41 | 23.56 | 1.33 | 0.64 |
| | mCast | 0 | 0 | 0 | 0 | 0 |
| Video Frame Losses (%) | Unicast | 1.22 | 5.98 | 38.78 | 2.52 | 1.75 |
| | mCast | 0 | 0 | 0 | 0 | 0 |
| Open vSwitch CPU Util. (%) | Unicast | 16.67 | 31.42 | 68.14 | 86.45 | 96.58 |
| | mCast | 4.92 | 6.68 | 10.15 | 11.88 | 14.72 |

mCast instead of IP unicast. The shown results represent the average of five runs of experiments.

### 4.4.1 Dropped network packets and video frames

Table 4.1 presents results for STAR topology and Table 4.2 shows the results for MESH topology. The results can be understood better by splitting them in two parts: when the system is not overloaded i.e. less than 600 clients and when it gets overloaded i.e. more than 600 clients. In the first case, IP unicast resulted in congestion in the network due to high bandwidth consumption with increasing number of clients. mCast reduced the consumed bandwidth by avoiding packet duplication and hence avoided congestion. This can be seen with zero packet loss when using mCast.

For more than 600 clients, the network nodes and streaming servers became overloaded and many clients failed to connect to the server. These failures were due to overloaded Mininet and Open vSwitch as shown by the CPU Utilization in Table 4.1 and Table 4.2. Real network nodes may have far more capacity than Mininet but the number of clients are also far higher. This testing scenario is shown to represent situations where the number of clients is high enough to surpass system resources.

In the case of mCast, the load on network nodes decreased significantly due to no packet duplication. Similarly, the load on servers reduced, as only one stream was transmitted for one video channel regardless of the number of clients. Consequently, all the client-requests and traffic was handled perfectly with all the clients connecting to the servers and no packet loss in the network. This shows the robustness of mCast when the system resources are limited.

Similar trends can be seen with video packets at the application layer of the receiv-

Table 4.2: Results of mCast experiments for MESH topology

| Metrics | Clients | 200 | 400 | 600 | 800 | 1000 |
|---|---|---|---|---|---|---|
| Failed Client Connections | Unicast | 0 | 0 | 191 | 677 | 938 |
| | mCast | 0 | 0 | 0 | 0 | 0 |
| Network Packet Losses (%) | Unicast | 0.37 | 5.62 | 13.02 | 5.61 | 1.15 |
| | mCast | 0 | 0 | 0 | 0 | 0 |
| Video Frame Losses (%) | Unicast | 1.38 | 9.108 | 22.91 | 9.01 | 3.88 |
| | mCast | 0 | 0 | 0 | 0 | 0 |
| Open vSwitch CPU Util. (%) | Unicast | 15.93 | 31.51 | 72.91 | 94.11 | 99.63 |
| | mCast | 6.24 | 9.55 | 12.04 | 13.94 | 18.17 |

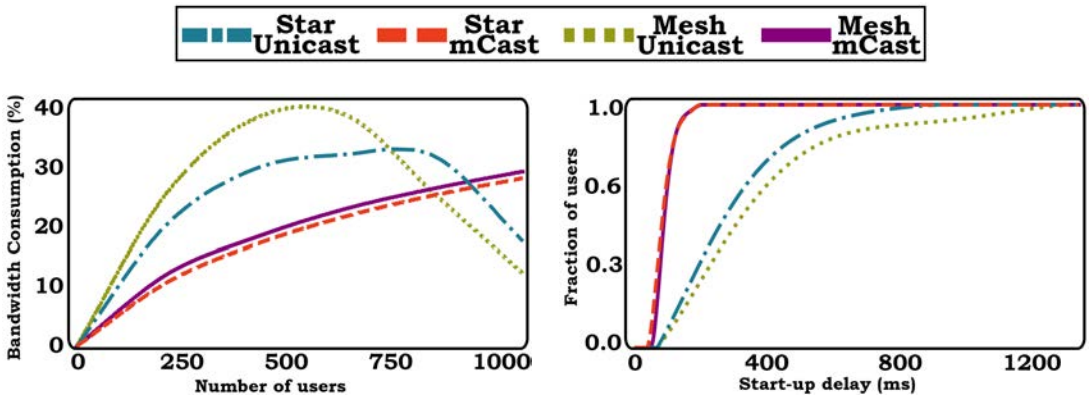
ing clients. The total number of video frames dropped by all the clients throughout the streaming duration was calculated. The frames that were received by a client but had errors or lost dependencies were also considered dropped. These packets are actual representation of the number of frames that are not decodable and will result in a decreased quality for the end-user. As the results show, mCast provides a better video to users by avoiding congestion in the network and eliminating the dropped video packets and un-decodable frames.

### 4.4.2 Link utilization and bandwidth consumption

The link utilization is measured as the percentage of link capacity of all the links in an ISP network, utilized over a given amount of time. The link utilization was calculated against the number of clients that were actively receiving a video stream (Figure 4.4a). As expected, in case of IP unicast the link utilization increased linearly with the number of clients. Duplicate packets passed through same link for each client, increasing the link utilization almost linearly, until congestion occurs and the links were saturated.

As in MESH topology, a stream has to traverse more links to reach the client, link utilization in MESH was higher than that of STAR topology. For mCast, the amount of traffic generated in ISP over a certain link remained constant avoiding any bottlenecks in the network. For a large number of clients, IP unicast overloaded network nodes and content servers, resulting in clients failing to connect to the servers. This adds unreliability to both the network and the streaming service. In mCast, such a situation did not occur as the content server was not overloaded even when the number of clients was very large i.e. 1000 active clients.

In addition to the decrease in intra-domain bandwidth consumption and link utilization, it is also important to notice that in mCast the stream enters the ISP network only once. Inter-domain traffic is usually more expensive and valuable




(a) Link Utilization (%) vs Number of clients (b) CDF of Start-up Delays

Figure 4.4: Comparison of link utilization and start-up delays.

for both ISPs and CDNs. Using mCast for a 1000 clients, this traffic got reduced to just one stream in mCast from 1000 streams in IP unicast.

### 4.4.3 Start-up delays

Start-up delay is the time a client has to wait from sending a content request until it starts receiving the stream. For fair comparison, the start-up delays were measured for 600 clients in IP unicast and mCast, as for more than 600 clients, the system resources are insufficient for unicast to server without failures. For unicast, the client request goes to the streaming server, which establishes a connection and starts streaming. For mCast, this request is intercepted by the mCast CDN agent which requests the ISP to deliver stream to the client and only if ISP fails to do so, does the request goes to the streaming server.

The results of delays are plotted (Figure 4.4b) as Cumulative Distributive Function (CDF) with delays in milliseconds (ms). As results show, the delay for mCast stayed below 200ms for all the users, while for unicast, it went up to 1140ms in the STAR topology and around 1300ms in the MESH topology. This is due to the extra load on servers and network nodes in the case of unicast. As mCast reduces these loads, the requests get responded to very quickly as shown in the results. The decreased start-up delays, along with no dropped packets, improves the video quality significantly and enhances the overall user experience.

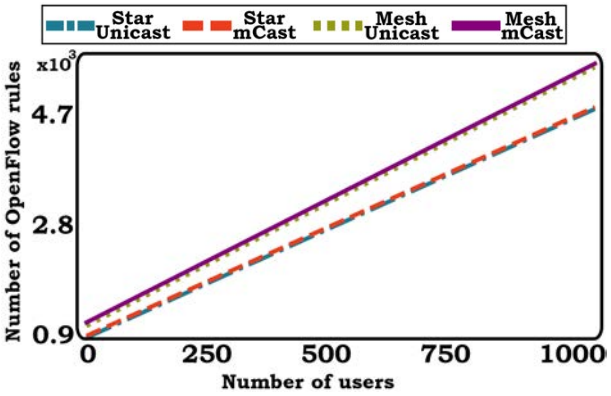



Figure 4.5: Number of OpenFlow Rules vs number of clients

### 4.4.4 Overhead cost of mCast

Results show that mCast can drastically decrease the network resource consumption for both ISPs and CDNs. However to setup mCast extra OpenFlow rules and messages are needed in the network. The goal of mCast is to minimize all types of resource consumption, therefore mCast is designed in a way that minimizes the number of OpenFlow rules and messages needed. The number of OpenFlow rules (Figure 4.5) depends on the number of switches involved in mCast. For every switch, mCast needs only one extra rule per stream to match the incoming packet's destination IP and port with $V_{(IP,\ Port)}$ and forward it on the relevant physical ports. The mCast rule at egress switch has additional actions to modify the destination IP and port to the client's address before forwarding. In OpenFlow v1.3 [44] these actions can be merged in one OpenFlow rule.

To add every client, the SDN controller calculates the path for that client based on the mCast tree and then sends an OpenFlow message to all the switches that need to add or modify their rules. While these control messages cause an overhead for the mCast service, these packets are usually very small and the overhead caused is negligible in comparison to the amount of bandwidth saved due to the reduced number of data packets Figure 4.4a.

## Summary

In this chapter, mCast, a novel scalable architecture for live streaming was proposed. With mCast, a CDN sends only one copy for all the clients of a video stream located in an ISP, leading to a significant reduction in the CDN egress link. Additionally, an ISP reduces the bandwidth utilization by eliminating redundant

transmission in its network. mCast is transparent to the client and maintains the CDN control on the content distribution. A decision model is presented that can be used by CDNs to determine when switching to mCast can be profitable for them. A large scale evaluation was performed with up to 1000 clients. Results showed that mCast decreased link utilization by more than 50% in comparison to IP unicast and reduced start-up delays to less than 200 ms for all users.

# Chapter 5

# Danos: Device-aware network-assisted optimal streaming service

## 5.1 Introduction

The latest social media trends have increased the amount of user-generated live video content over the Internet through various streaming platforms such as Periscope, YouTube Live and Twitch, giving rise to frequent flash crowd events [2]. Additionally, more traditional TV programs such as news, sports and political events are now streamed online at High Definition (HD) qualities to large and ephemeral audiences, making it challenging to provision streaming resources in an efficient and flexible manner. The heterogeneous device capabilities of users, ranging from smart-phones and tablets to ultra-HD 4K TVs, further elevate the challenge to dynamically deliver the video streams at multiple bitrates that match the specific needs and requirements of each user [4].

When delivering live video streams to clients located in Internet Service Providers (ISP), the Content Delivery Networks (CDN) rely on either IP unicast as in DASH-based systems, or overlay multicast as in P2P-based systems [5]. These approaches enable good control and management of end-devices by establishing end-to-end connections, but result in redundant transmissions in the network layer and waste system resources for both ISPs and CDNs. On the other hand, native IP multicast can help reduce resource consumption by eliminating packet duplication over network links and at content servers, but its adoption is stymied

by lack of desirable features such as management, authorization and accounting [6]. Today, IP multicast is limited to intra-domain pre-provisioned services such as ISP-oriented IPTV [7].

Software-Defined Networking (SDN) significantly enhances the degree of control in IP networks [8] and can enable dynamically manageable network-layer multicast over the Internet. An increasing number of ISPs and CDNs are incorporating SDN in their domains [9], which serves as a motivation to rethink the design of live video streaming services. mCast (Section 4) has proposed services and architectures that enable efficient inter-domain content delivery using network-layer multicast. However, it has salient limitations that would restrict a practical deployment. Specifically, a practical solution must account for the heterogeneity and capability of end-user devices, support multiple video qualities and not ignore the adaptive bitrate aspect of live streaming. It must also handle network congestion while offering CDNs the control and privacy essential for their businesses.

In this chapter, Danos, a Device-Aware Network-assisted Optimal Streaming service, is proposed, that merges the flexibility and control of SDN with resource efficiency of network-layer multicast to enable inter-domain adaptive bitrate streaming. Danos involves minimal modification at the server and client side and supports transparent multicast for end-nodes. Similar to mCast, Danos is designed to maintain user and CDN data privacy by avoiding any deep packet inspection and provides CDNs with full control over their clients so they can perform authentication, authorization and accounting (AAA) functions. Through Danos, four contributions have been made:

- An SDN-based architecture that enables adaptive bitrate live streaming service for network-layer multicast transmission. The essential components and various architectural design choices that can be made by CDNs or ISPs are addressed.
- The formulation of a novel multi-objective optimization problem to maximize the perceived video quality of all the users and minimize the utilization of the ISP network. The formulation accommodates ISP or CDN operation constraints and scales with the number of users. The performance analysis of the model shows that it can be solved in the order of milliseconds for millions of users in the network, and can improve video quality for users in comparison to mCast.

- Design features to address key challenges that arise by using optimized multicast and considering device-specific requirements when serving bitrate adaptive live video streams. Multicast sessions are UDP-based and, unlike TCP, there is no feedback loop or constant interaction between streaming servers and clients. An approach is proposed that presents how the bitrate of users can still be switched smoothly without causing frame jitter. Furthermore, a mechanism is devised to synchronize various components in the parallelized architecture when optimizing the network and minimizing the start-up delays for clients.
- A prototype of Danos is implemented for demonstration and evaluation, and large-scale experiments with up to 500 clients streaming multiple videos at multiple bitrates. Danos is tested for real-world scenarios including flash crowd events and cross-traffic and the performance of Danos is evaluated against mCast.

## 5.2 Architecture and components

Danos enables efficient delivery of bitrate-adaptive video streams in both CDN and ISP networks by leveraging SDN capabilities. Specifically SDN facilitates global network state, flexible data forwarding and standardized real-time route modification and packet editing. Danos deploys agents in both CDN and ISP domains to allow unicast delivery of different streams from a CDN to an ISP network and network-layer multicast inside the ISP network. Such sessions are denoted as Danos *sessions*. It is assumed that the ISP offers such sessions as a service for interested CDNs. Based on their business requirements and decision models (Section 4.3), CDNs may request such services, and ISPs would accordingly set up and manage multicast trees for Danos video streams.

Figure 5.1 illustrates the architecture of the Danos system and its various essential components. The remainder of this section, first describes the functionality of each individual component. Then, it presents the overall workflow of Danos (Section 5.2.9) using examples of important messages exchanged between the different components to initiate and maintain a Danos session.

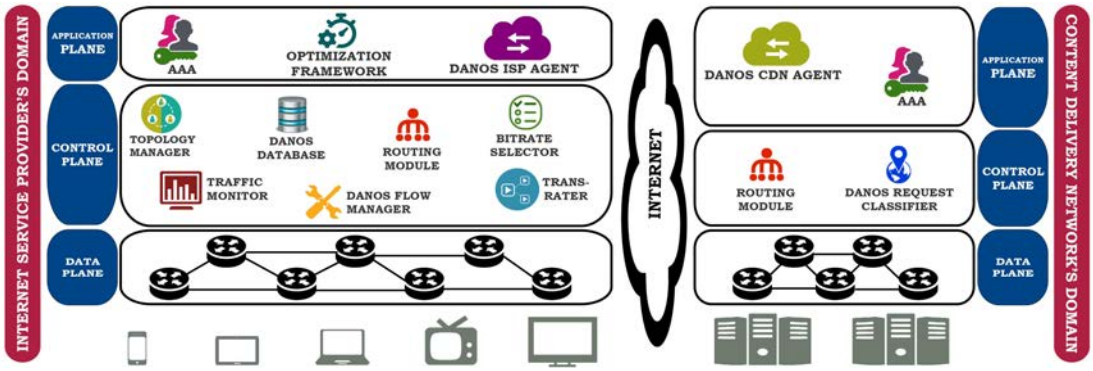



Figure 5.1: Architecture and components of Danos

### 5.2.1 Request classifier

When a client wishes to watch a video stream served by a CDN it sends a session join request, which is typically delivered to AAA modules that authenticate requests and handle legitimate requests. In Danos, after handling the request, AAA forwards it to a Request classifier module, which is located in the control plane of the CDN. The classifier identifies the client's ISP network and forwards the request to the Danos CDN agent. Such classification can be performed using IP address ISP mapping or geo-location databases.

### 5.2.2 Danos CDN agent

The Danos CDN agent is implemented at the application plane of the CDN's SDN-based architecture and decides how content is served. On reception of the user request, the Danos CDN agent determines whether it would be served as a unicast stream or would be part of a Danos session. Generally, the request would be served as unicast if it is not from a Danos-based ISP or if the CDN policies to initiate a Danos session have not yet been met.

**Unicast-case**: The Danos CDN agent prepares and sends a standard content request message to the selected server and configures the CDN forwarding nodes for the stream delivery. The configuration is done by sending messages from the SDN controller to the forwarding nodes, e.g., *flow_mod* messages in OpenFlow. The server starts streaming the content to the client device. The routing of this flow is handled normally by the CDN routing module.

**Multicast-case**: If the requested stream is already served by a Danos session, then the CDN agent will ask the ISP agent to add the new client to the multicast

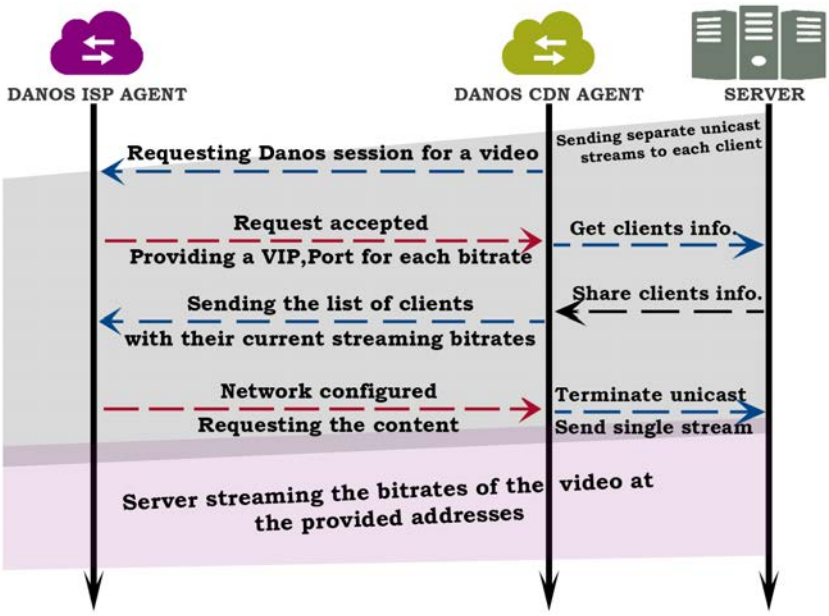



Figure 5.2: Message exchanges between a CDN and an ISP to initiate a Danos session for a live video content

group of the stream. This request includes the client's IP address, port number, the content server's IP address and a randomly generated source port number which will be used for smooth switch-over if reverting back to a unicast stream for that client. This *add-client* request also includes the device specification and the user's CDN subscription level details, such as the maximum allowed bitrate or the minimum bitrate guarantees.

The Danos CDN agent implements multicast policies that determine when a stream should be transmitted over multicast in an ISP. In this case, the Danos CDN and ISP agents interact to initiate the transition process as presented below. Typically, a multicast policy depends on the economics of bandwidth utilization and the ISP charges for multicast delivery in the latter network.

### 5.2.3 Danos ISP agent

The Danos ISP agent is implemented at the application plane of the ISP network and handles the requests coming from CDNs. It performs two main functions: interfacing with the Danos CDN agent using an SDN east-westbound interface and orchestrating multicast operations in the ISP network using the northbound interface.

Figure 5.2 illustrates the message exchanges that take place between the Danos

ISP and CDN agents. When a CDN wants to trigger multicast for a video stream, it sends a session aggregation request to the ISP agent along with the details of the transmission bitrates of the stream. After analyzing the request and performing initial assessment, the ISP agent responds by creating an identifier for each bitrate of the stream. The identifier is composed of an IP address and a port number, denoted as $V_{(IP,\ Port)}$. An ISP can choose an address from a pool of IPv4 addresses that it reserves for the Danos service. It can then use IP addresses and port numbers to distinguish video streams or bitrates.

Upon reception of $V_{(IP,\ Port)}$, a CDN sends *add-client* requests with details of the clients located in the ISP network and watching that video. These details include the header fields, their current streaming bitrates, and maximum and minimum allowed bitrates for each user. Such information is used to identify the target users in the ISP network and ensure a good video quality for users.

The Danos ISP agent then instructs the modules, discussed in the following sections, to create multicast trees, configure the network, and ensure transparent delivery to all the clients. To provide complete control of clients to a CDN, the ISP does not intercept any *session-join* or *session-leave* requests from clients, even after a multicast stream has been initiated using Danos. These requests are instead delivered to the CDN to be handled as described above. The Danos CDN agent may ask the Danos ISP agent to remove a user from a Danos session when needed (e.g., on channel switching or application termination).

Note that the propagation delay for the control is comparable for unicast clients and clients added to Danos sessions. Moreover for data delivery, Danos offers lower start-up delays than a conventional streaming service, as the content is already available near the client through Danos multicast streams. Such interaction and dynamic updating of the multicast trees is facilitated by the centralized control and global view of the SDN and is not achievable in a traditional non-programmable Internet architecture.

### 5.2.4 Danos database and bitrate selector

Once a Danos session has been initiated for a video stream, the CDN agent sends details of clients to the ISP agent to add to the stream. In a Danos session, the best streaming bitrate for each client should be served. This rate depends on different factors:

- The receiving devices can have different specifications and capabilities resulting in different bitrate limitations.
- CDNs may offer different subscription levels to users which may limit the maximum bitrate that a user can be served with.
- ISPs may offer different packages to users with different bandwidth limitations.
- Based on the current cross-traffic being generated or received by a user, the bitrate that a user can be currently served may vary.

The proposed architecture gathers the information provided by the CDN and collected by different SDN modules and stores it in a database. The database saves the multicast trees for all the bitrates of each video stream in the form of network graphs. When an *add-client* request is received by the Danos ISP agent, it updates the database with any new received information, instructs the bitrate selector to choose a bitrate for the user, and updates the graph of the chosen bitrate accordingly. For clients switching from unicast to multicast, the Danos CDN agent provides their current streaming bitrates to the ISP agent and to enable smooth switch-over these clients are initially served with their current bitrates and are optimized later as explained in Section 5.4.

For any new clients, the bitrate selector accesses the Danos database and uses all the available information to find the highest bitrate that the user can support and passes on that information to the Danos ISP routing module for further processing.

### 5.2.5 Danos routing modules

In the CDN network, the CDN routing module implements the CDN routing policies and installs relevant flow rules for both unicast and Danos streams. In the ISP network, the ISP routing module receives a message from Bitrate Selector that includes the highest bitrate for a client. The ISP routing module accesses the information saved in the database by the Topology manager module and finds a path from the ISP entry point to the end-user and assigns the highest bitrate that is available on the path and can be assigned to the user. It then adds the user to the multicast tree of that bitrate and passes on that information to the Danos flow manager to configure network nodes and deliver the stream to the client.

### 5.2.6 Danos optimization engine

To minimize the start-up delays for users, the Danos ISP routing module finds paths and configures the network for clients as soon as an *add-client* request is received from the CDN agent. However, the bitrate chosen for a client might not be feasible or optimal depending on the current network state. The Danos Optimization framework runs as a network application on the application plane of the ISP network and uses the global knowledge acquired by the Danos database to find the optimal solution for all the videos and clients that are currently being served in the network using Danos. This framework also considers any cross-traffic on the links on paths used for multicast. Such information is stored in the database by the Traffic Monitor module of the ISP. The Danos optimization engine can function in two different modes:

**Event-based**: The network and client states can be monitored and the optimization module initiated when certain events occur; e.g., a new Danos session being added to the network, the amount of cross-traffic on the multicast paths exceeds a certain threshold, and/or a certain number of clients have been added to or removed from a video stream.

**Periodic**: The optimization framework can run at regular intervals, making the optimal decisions and updating the network paths and user bitrates. Re-configuring the network too often can result in excessive signaling overheads, therefore when running the framework periodically, the network and user devices may only be re-configured if the gains of doing so increase beyond a certain threshold.

### 5.2.7 Danos flow manager

The ISP Routing Module and Optimization Engine choose bitrates for clients, decide the paths to be used and instruct the Danos flow manager to configure the forwarding nodes. The flow manager uses the south-bound interface to install rules on SDN-enabled switches. The multicast entries in network nodes are installed with higher priority than IP unicast, ensuring that clients are served with Danos streams whenever possible. In addition, the Danos flow manager installs transparency rules on the egress switch. Before forwarding a packet to the client, the transparency rule modifies the source and destination IP address and port number to match the client-specific details, so the client receives the packet just

as it would in IP unicast, hence the delivery is transparent. These rules ensure that the clients do not need to be modified to support Danos streams. These rules also help the ISP to identify the amount of traffic that goes to each user. An ISP can use this information to charge or limit users based on their individual billing plans.

### 5.2.8 Danos streaming server and client

A typical live streaming server streams the requested content via unicast-based RTP/UDP sessions. To support Danos, an API is implemented in the server to communicate with the Danos CDN agent. When the CDN agent wants to switch the transmission mode between unicast and multicast, it uses this interface to perform the following tasks:

**Acquire the current state of the clients**: This information is helpful for a smooth transition from unicast to multicast and providing a seamless video experience to active clients.

**Terminate individual client transmissions and initiate a single transmission instead**: Along with this instruction, the CDN agent provides a destination IP and port, $V_{(IP,\ Port)}$ to the server for each bitrate and the server establishes connections. To avoid any lost or skipped frames at the client device, the new connection is activated before terminating the old unicast sessions. This sequence avoids any skipped frames; however, it may lead to duplicate packets at the client, which can be ignored by the client after inspecting the RTP headers.

**Terminate single transmission per bitrate and initiate individual client transmissions**: When switching back from multicast to unicast, the CDN agent will send a list of clients (IP,port) to the server with an instruction to terminate the connection with $V_{(IP,\ Port)}$ and the server will act accordingly.

Due to the transparent delivery mechanism of Danos, any standard UDP-based live video client can be used with a minor consideration at the application layer. To enable smooth switch-over between different video bitrates, a design is proposed in Section 5.3.1 and a Danos client should incorporate this design to correctly handle and decode video frames.

Figure 5.3: An example workflow of adding clients to an established Danos session.

### 5.2.9 Danos workflow

An example (Figure 5.3) is provided to explain how different components in the ISP domain interact with each other.

The Danos CDN agent decides to deliver a video stream S, available at bitrates 1-Mbps and 2-Mbps, through a Danos session. It initiates this session by following the steps shown in Figure 5.2 and starts delivering both bitrates to the ISP using single streams. Client C-1 sends a *session-join* request to the CDN and receives an IP address of a content server after AAA functions have been performed. When C-1 sends a *content-request* at the provided address, the Danos CDN agent receives it and sends an *add-client* request to the Danos ISP agent ①. Along with C-1's IP address and port number, this request includes information such as the maximum bitrate that the client can support and any CDN user subscription details. The ISP agent stores the newly learned information in the Danos database ② to be accessed by other modules and sends an instruction ③ to the Bitrate Selector to find an appropriate bitrate for C-1.

**The bitrate selector** accesses the information stored in the database ④ by the Traffic Monitor, Danos ISP agent and AAA modules of ISP and chooses 2-Mbps as

the best bitrate for C-1. It instructs the routing module ⑤ to find a suitable path for C-1. To locate the client in the ISP network, the routing module retrieves the information stored in the Danos database ⑥ by the Topology manager module. It then analyzes the graph representation of the multicast tree for the chosen bitrate, to find the least expensive path to add C-1 to the tree and decides to deliver the stream to C-1 through Switch-1. The routing module updates the graphs and saves them back to the database, to be used later by the optimization framework and sends a request to the flow manager ⑦ to configure the forwarding nodes and enable stream-delivery for the client.

**The Danos flow manager** installs or updates forwarding entries on the nodes in the path to deliver the bitrate to Switch-1 and installs or updates transparency rules on Switch-1 ⑧ to modify the packet headers before forwarding them to C-1. At this point, C-1 will start receiving the requested video quality at 2-Mbps. The process is repeated when the Danos ISP agent receives a new *add-client* request from the CDN agent for C-2 and C-2 starts receiving 2-Mbps bitrate through Switch-2.

**The optimization framework** runs as a separate application than the bitrate selector. It gains the global information of clients and network state from the Danos database ⑨ and solves the optimization problem to maximize a system utility while respecting network, user and device constraints. In this example, the optimization framework detects congestion on the path to C-2 and realizes that C-2 will experience frame losses if it kept on receiving 2-Mbps and decides to stream 1-Mbps to C-2 instead. It instructs the flow manager ⑩ to reconfigure the forwarding nodes and Switch-2 accordingly. The flow manager updates the forwarding entries ⑪, using the smooth switching technique discussed in the next section, and switches the bitrate of C-2 to 1-Mbps which now receives a better quality of video by eliminating frame drops.

## 5.3 System design

Three key design challenges are addressed that arise by using optimized multicast and considering device-specific requirements when serving bitrate adaptive live video streams.

### 5.3.1 Smooth switching of client bitrates

Video streams are generally encoded in the form of a Group of Picture (GOP). A GOP is a collection of successive inter-frames (I-frames), that are encoded independently of all other frames, and intra-frames (P or B-frames), that are dependent on other frames for successful decoding. Receiving an incomplete GOP may cause the user to experience glitches or jitters in the video, based on the type or number of lost frames. To avoid video glitches and jitters, the bitrate of a client should only be changed after it has received a complete GOP i.e. at the next I-frame.

In unicast transmissions detecting an end of a GOP is not a problem because each client has its own separate connection with the streaming server and the server would know when a GOP has ended. As existing multicast-based solutions only consider single bitrate schemes and do not change the quality, thus they do not face or address this issue either. In the proposed adaptive bitrate multicast-based solution, when the ISP optimizer module decides to switch the bitrate of a client based on the current network state, it must prevent the client from losing a GOP and experiencing jitter.

The decisions made by the optimization framework translates into three types of actions:

- Change the network path for one or more bitrates of a stream;
- Serve one or more clients with a new bitrate;
- Stop serving those clients their current bitrate.

These actions are then translated into flow-table entries and are installed on the forwarding nodes by the Danos flow manager. To ensure a smooth bitrate switch for clients, each action is handled in different ways. If a decision only involves changing the path and re-routing the traffic, it is installed on the forwarding node immediately. However, if a decision involves switching the bitrate for a client, the action of delivering the new bitrate to the client is installed immediately but the action to stop serving the current bitrate is taken with a delay of a duration of one GOP.

Figure 5.4 shows an example to demonstrate a switching action. A GOP of 8 frames is considered with the video encoded at 24 frames per second (fps). This implies a GOP duration of $333ms$. For a client, the video is delivered at 1-Mbps until time $t_0$. At time $t_0$, the optimizer decides to switch this rate to 2-Mbps and

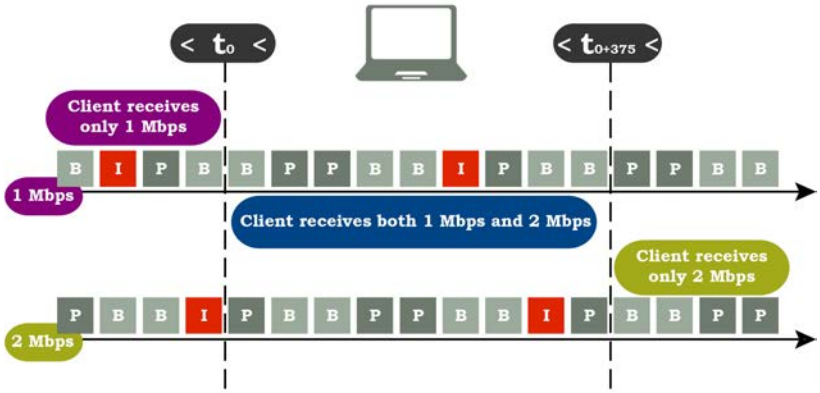



Figure 5.4: An example of bitrate switching process. *A smooth switch is ensured by delivering both rates for a duration of a GOP.*

initiates the switching process. The action to deliver 2-Mbps is taken immediately and the client starts receiving frames at both bitrates.

In this example, Danos waits for a duration of 375$ms$ during which time the client receives 9 frames at both bitrates. This will ensure that the previous GOP has been completely received by the client and the user does not experience any jitter or glitch in the video. The event where a client receives frames from two distinct bitrates can serve as an indication that a bitrate switch is taking place and the client can decode the correct frames by inspecting the RTP headers while discarding the additional unnecessary frames from both bitrates.

This design choice avoids the need for deep-packet inspection to detect the start and end of a GOP in the ISP domain and maintains content provider's and user's data privacy. It also avoids the need of constant signaling between ISP and CDN Agents to share the start and end of each GOP. The CDN Agent will instead share the duration of a GOP when initiating a Danos session for a stream and the Danos flow manager will use that information for a smooth transition of bitrates for clients.

Multiple rates of an adaptive bitrate live video are always transmitted synchronously. This prevents users from experiencing frame skipping or playout delays when switching between bitrates. A minor time difference, due to the dynamic nature of network or different amount of data for different bitrates, can be handled by the client's playout buffer.

During the smooth switching process, the forwarding nodes will be transmitting two distinct bitrates ($R_1$ and $R_2$) of the same video stream. If any of the links on the path are congested, it might result in an input rate ($R_{in}$) higher than the out

rate ($R_{out}$) of the network buffer in the respective forwarding node. An ISP must plan ahead for such events to avoid buffer overflows and ensure reliable packet delivery. A simplified example is provided here, on how to calculate the extra amount of buffer length needed for a stream.

Assume that a video is being streamed at full-HD with a bitrate of $R_1 = 8Mbps$ and 4K at $R_2 = 20Mbps$, and on the path to a client, one of the network buffer can support $R_{out}$ up to $20Mbps$. The optimization framework chooses to switch the client to 4K and instructs the Danos flow manager to initiate smooth switching. The total $R_{in}$ during the switching period will be $R_1 + R_2 = 28Mbps$ which is $8Mbps$ more than $R_{out}$ of the bottleneck buffer and will result in packet drops. Assuming that the video is encoded at 24fps and 8 frames per GOP, the additional bytes that will arrive in one GOP duration ($8/24 * 1000 = 333ms$) will be $8Mbps/8/334ms = 333KBytes$. Considering a Maximum Transmission Unit (MTU) of 1500 bytes, $333KBytes/1500 = 222$ additional packets will arrive at the network buffer during the switching period. A similar exercise can be conducted for each video with its respective highest bitrates to get the total size of extra queue length needed.

An additional advantage of an SDN-based architecture is the centralized control provided by the SDN control plane, that can be used to perform these calculations dynamically and reconfigure buffer lengths based on the policies and priorities of the ISP network.

### 5.3.2 Synchronizing optimization and routing modules

The optimization framework is activated by the Danos ISP Agent to run periodically or based on some events. It runs as a separate application and optimizes a system utility and the network link utilization by solving an optimization problem. One key challenge for running the framework as a separate application is handling the clients that arrive during the computation span i.e. after the problem starts computing until the optimal solution is found and the framework is ready to re-configure the network.

If any new clients arrive during the computation span, the solution found by the optimization framework may not be optimal anymore, depending on where the new clients are located in the network and their specifications. Although this problem is natural for real-time systems and cannot be avoided, the decision on paths or bitrates to choose for these clients impacts the system performance.

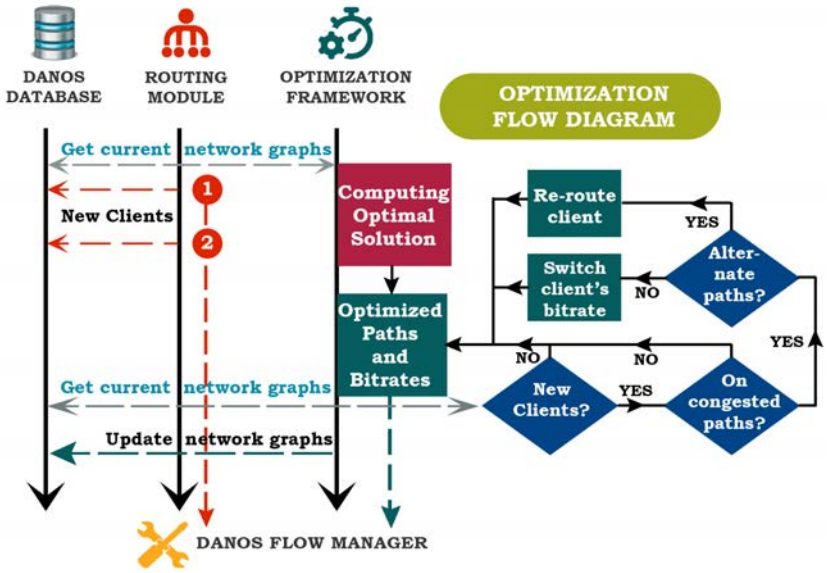



Figure 5.5: Flow diagram of optimization framework: *Clients that arrive during computation will be switched to lower bitrates only if their current rates cannot be served.*

There are two ways to handle these clients:

- Add the new clients to the multicast paths as soon as the request arrives
- Delay serving new clients until the end of computation span of the optimization framework

As mentioned in Section 5.2, to minimize start-up delays, the Danos ISP Routing Module finds paths and requests network configuration as soon as an *add-client* request is received. Delaying clients until the end of computation span would imply pausing the routing module and increasing start-up delays which has a negative impact on user experience. On the other hand, adding the clients as soon as they arrive might lead to the routing module choosing paths with congested links, which the optimization framework would have resolved by solving the constrained-problem globally.

This issue is resolved by proposing a hybrid approach as illustrated in Figure 5.5. At the start of the computation span, the optimization framework accesses network paths and graphs from the Danos database and saves a copy of it. It then proceeds with solving the problem and making decisions which translate to either re-routing or switching bitrates for some clients. During the computation span, the routing module keeps adding new clients to the session.

After solving the optimization problem, the framework accesses the database

again and gets the current network paths and graphs to determine whether any new clients have been added during the computation span and what paths and bitrates have been assigned to them. If the new clients happen to be on the optimal paths and receiving the optimal bitrate already, then no further action is needed and the optimization framework can configure the network based on its solution. Similarly, the optimization framework leaves the clients on un-congested paths untouched and does not update its solution to alter their paths.

If a client is assigned a certain bitrate on a congested path and the paths decided by optimization framework are able to deliver the same bitrate to the client through another path, then the action to reroute that client is added to the optimal solution. However, if no alternate paths are available then the optimal solution is updated with actions to initiate smooth switching for the client.

This approach avoids additional start-up delays for clients and handles any network congestion that might occur due to an un-optimized path. The final solution is then sent to the Danos flow manager and the network is re-configured accordingly.

### 5.3.3 ISP transrating services

Danos provides a resource efficient delivery mechanism for CDNs using multicast and the CDNs can save their energy cost and Internet transit cost by serving all the clients in an ISP with one stream per bitrate of a video. To further maximize on the savings, a transrater module is proposed for the ISP network, that can be rented by the CDNs. After establishing multicast for a live video session, if a CDN opts for transrating services then it will deliver one stream per video through the Internet Exchange Point (IXP) into the ISP network where it will be received at the transrater module. This stream can then be requested by the forwarding modules of the ISP as needed.

This design is beneficial for both ISPs and CDNs. For a CDN the costs for serving a stream at multiple bitrates will be reduced. For an ISP, in addition to the monetary gains, the ISP network will be capable of accessing the streams in a more dynamic fashion by requesting the transrater to generate only the bitrates that are currently being served to the active clients in the network.

If an ISP provides Transrating services, the Danos flow manager will interface with the Transrater module to facilitate efficient resource utilization in the ISP

network for the CDNs that are renting transrating services. When installing flow entries on the switches, the flow manager will check if a new bitrate is to be served to any clients in the network. For such cases, it will request the Transrater to make that bitrate available in the network. Similarly, when removing entries from switches, the flow manager will check if a certain bitrate is no longer served to any client in the network. In this case, the transrater will be informed to stop transrating the video on that bitrate.

## 5.4 Global and real-time optimization

For the Danos Optimization Framework, a novel multi-objective optimization problem is formulated, with the primary goal to maximize the system utility of all the Danos clients in the network. The model can solve the problem for any system utility that is a function of user-bitrates or bitrate-switches or both. This makes the model generic and independent of any particular Quality of Experience (QoE) or fairness function. A secondary goal is defined to minimize the network link utilization which will allow more resources for non-Danos users and hence improve the overall resource utilization.

As discussed in Section 5.3.2, a longer computation time for the model can result in a sub-optimally configured network. Therefore, a scalable guided-optimization approach is also proposed for the Danos Optimization Engine that can solve the problem in real-time regardless of the number of users in the network.

### 5.4.1 System model

An ISP network is considered, with a number of forwarding nodes used for Danos to serve a set of videos $V$ to multicast users in set $M$. Each video $v$ is encoded and streamed at a set of bitrates, denoted by $R_v$ and a node $i$ may receive one or more bitrates of $v$ from any of its previous hop node $j \in L_i$. The capacity on the link from node $i$ to $j$ is denoted by $c_{ij}$ and for a user $m$ this capacity will be limited based on the user's ISP subscription level. Each link may have some cross-traffic and a constant $\alpha_{ij}$ represents the percentage of the capacity available or allowed for multicast sessions. Note that, even for links with no cross-traffic a small percentage of total link capacity can be made inaccessible to multicast sessions, to avoid large network queuing delays.

Table 5.1: Notation for the Danos optimization model

| Symbol | Description |
|---|---|
| | **INPUTS** |
| $V$ | Set of active video sessions served by Danos |
| $M$, $M_v$ | Set of multicast users ($M$) subscribed to video $v$ ($M_v$) |
| $R_v$ | Set of bitrates available for video $v$ (in ascending order) |
| $m_{v_{max}}$ | Index of maximum bitrate allowed or possible for user $m$ watching video $v$ |
| $m_{v_{min}}$ | Index of minimum bitrate guarantee for user $m$ watching video $v$ (Set to lowest if not specified) |
| $N$ | Set of all the network and user nodes |
| $L_i$ | Set of previous hop nodes to reach node i |
| $c_{ij}$ | Capacity on the link between node $i$ and $j$ |
| $\alpha_{ij}$ | Maximum fraction of link capacity allowed for Danos between node $i$ and $j$ |
| $\beta_{ij}$ | Priority weight for the link between node $i$ and $j$ |
| $f(m, r)$ | A system utility function with user $m$ and bitrate $r$ as inputs |
| | **VARIABLES** |
| $B_{ijvr}$ | Binary variable to determine if bitrate $r$ of $v$ has been chosen to be transmitted from node $j$ to node $i$ |

A video $v$ enters the ISP domain at a forwarding node $e_v$. A user $m$ may request one or more videos at the same time. For $m$ watching a video $v$, a maximum bitrate $m_{v_{max}}$ that can be assigned to the user is defined based on its device specification and CDN user subscription level. The model also accepts a minimum bitrate guarantee $m_{v_{min}}$ that might be offered to $m$ for video $v$. Table 5.1 summarizes all the notation used for the model.

### 5.4.2 Problem formulation

The optimization problem is an Integer Linear Program (ILP) with the primary objective of maximizing the cumulative system utility for all users. The problem is agnostic to the utility function being used and accepts any metric as a utility function that takes a bitrate as an input and returns the achievable utility value for a user. The goal is to maximize the sum of the utility function for all users of all videos.

**Problem 1: Optimal user bitrate assignment.**

$$max \sum_{v \in V} \sum_{m \in M_v} \sum_{r \in R_v} \sum_{j \in L_m} f(m,r) \cdot B_{mjvr} \tag{5.1a}$$

*subject to*

$$\sum_{r \in R_v} \sum_{j \in L_m} B_{mjvr} = 1, \ \forall v \in V, m \in M_v \tag{5.1b}$$

$$\sum_{r=R_v[m_{v_{min}}]}^{R_v[m_{v_{max}}]} \sum_{j \in L_m} B_{mjvr} = 1, \ \forall v \in V, m \in M_v \tag{5.1c}$$

$$\sum_{v \in V} \sum_{r \in R_v} r \cdot B_{ijvr} + \sum_{v \in V} \sum_{r \in R_v} r \cdot B_{jivr} \leq \alpha_{ij} \cdot c_{ij} \ \forall i \in N, j \in L_i \tag{5.1d}$$

$$B_{ijvr} - \sum_{k \in L_j} B_{jkvr} \leq 0 \ \forall v \in V, i \in N - e_v, j \in L_i, \tag{5.1e}$$

where $B_{ijvr}$ is a binary variable that determines if bitrate $r$ of video $v$ has been chosen to be transmitted from node $j$ to node $i$. The objective function 5.1a only considers the links between users and their egress switches to check the bitrates that will be delivered to the end-devices and ignores the links between forwarding nodes as these do not add any value to the user experience.

Constraint 5.1b ensures that a user only receives one bitrate of a video stream and Constraint 5.1c ensures that the assigned rate meets the minimum guarantee criteria and does not exceed maximum allowed bitrate by CDN or the maximum bitrate stream-able by the user device. Constraint 5.1d limits the total multicast traffic flowing through a link to the maximum capacity allowed for the Danos service. For bi-directional links, the traffic flowing in both directions i.e. $i$ to $j$ and $j$ to $i$ is considered and the sum should not exceed the allowed capacity.

Finally, Constraint 5.1e establishes paths in the network by guaranteeing that a node only transmits the traffic that it receives. This is achieved by ensuring that if a bitrate of a video is transmitted on a link i.e. $B_{ijvr} = 1$, then at least one of the previous hop link of node $i$ should be 1. This is checked for all the forwarding nodes in the network except the entry point. The model assumes that the entry point always has all the bitrates and videos available. If a CDN is using the ISP transrating service then the entry point will be the node attached to the Transrater module or else it will be the ingress node between ISP and CDN.

A secondary objective of the model is to minimize the resource utilization in the network. If more than one solution exists to *Problem 1*, where users would receive the same optimal bitrates, then this objective will find the solution with

the least redundant transmissions on network links. Depending on the planned infrastructure of the network and how an ISP wants to shape the traffic, the ISP may have different priorities set for each network link, e.g. to implement load balancing or fail-over mechanism. $\beta_{ij}$ represents the priority weight for a link between node $i$ and node $j$. Note that if no such priorities exist then this constant will be set to 1 for all links.

**Problem 2: Minimizing the network load.**

$$min \sum_{v \in V} \sum_{r \in R_v} \sum_{i \in N} \sum_{j \in L_i} r \cdot \beta_{ij} \cdot B_{ijvr} \tag{5.2a}$$

*subject to*

$$\sum_{r \in R_v} \sum_{j \in L_m} B_{mjvr} = 1, \ \forall v \in V, m \in M_v \tag{5.2b}$$

$$\sum_{r=R_v[m_{v_{min}}]}^{R_v[m_{v_{max}}]} \sum_{j \in L_m} B_{mjvr} = 1, \ \forall v \in V, m \in M_v \tag{5.2c}$$

$$B_{ijvr} - \sum_{k \in L_j} B_{jkvr} \leq 0 \ \forall v \in V, i \in N - e_v, j \in L_i \tag{5.2d}$$

If solved without *Problem 1*, the objective function 5.2a would choose the minimum bitrate for each user. However, with 5.1a as primary objective, the link utilization will only be minimized when it does not impact user video quality defined by the system utility function. Constraints 5.2b, 5.2c and 5.2d are the same as Constraints 5.1b, 5.1c and 5.1e.

### 5.4.3 Real-time guided optimization

The problem is an ILP with binary variables and belongs to Non-deterministic Polynomial Time (NP)-complete [108] class. However, depending on the number of video streams, the number of users and the size of the network, it might take too long to solve the problem in real-time and fast enough to be implemented in the dynamic optimization framework. The envisioned use cases of multicast streaming involve large numbers of users and the time span of computation is a crucial factor to the overall efficiency and performance of the system.

To make the model scalable, the dependence on the number of users is eliminated, by relying on the fact that there are only a limited number of distinct user classifications. The users are classified based on their ISP or CDN subscription and device specification. Each of this classifications results in a minimum and

a maximum bitrate that the user can be delivered. These classes are combined together at each egress node, to form a group and identify users that belong to a certain group.

For example, user devices attached to an egress node will form a group, if: they are allowed full-HD quality by the CDN with a maximum bitrate of $8Mbps$; have an ISP bandwidth capacity between $8Mbps$ and the next higher bitrate served by the CDN (say $20Mbps$ for 4K) and; are using a device that can seamlessly support $8Mbps$ streaming. Similarly all the user devices at each egress switch are grouped together based on the available bitrates for the video and their ISP, CDN or device capabilities.

The optimization model is re-defined to take the groups as an input instead of individual users. An additional weight factor $w_{gj}$ is introduced in the objective function of *Problem 1*, which is equal to the number of users in a group $g$ attached to an egress switch $j$. The remaining constraints and **Problem 2** are applied in the same way but by replacing the set $M$ of all the multicast users by set $G$ which represents all the groups. The objective function is as follows:

$$max \sum_{v \in V} \sum_{g \in G} \sum_{r \in R_v} \sum_{j \in L_g} w_{gj} \cdot f(g,r) \cdot B_{gjvr}, \tag{5.3}$$

where each $g \in G$ is defined by the connecting egress switch, ISP and CDN user subscription level, and device specification. As the number of elements in $G$ are limited and will not increase with the number of users, this modelling technique makes the optimization model very scalable and helps solving the problem in real-time.

### 5.4.4 Performance analysis

**Scalability and computation time**: There are three main factors that can impact the scalability of the optimization model: number of users, number of video streams and the size of the network. The model is implemented in the Gurobi solver [109] and the mathematically optimal solution is computed for two ISP topologies from the Topology Zoo database [94] including the AT&T network (a MESH topology) with 25 forwarding nodes and KREONET (approximately a STAR topology) with 13 nodes. The capacity of all the network links is set to $40Mbps$.

The computation time of the model is determined by varying the number of users

that have characteristics as described in Section 5.5, and changing the number of video streams, each served at average bitrates of $400kbps$, $1.5Mbps$ and $4Mbps$. For fair comparison among different number of users, this analysis ignores the time spent to load the users and variables into the model and only focuses on the time taken to solve the model and find the optimal solution. Figure 5.7 shows the results averaged over 20 runs.

For the smaller STAR topology, the solution was computed within 100ms for all cases. As the model implements a scalable pre-grouping approach, the number of users did not increase the computation time significantly. Even for 1 million users in the larger MESH topology, the optimal solution for a single video stream was found within $2ms$. The computation time increased with the number of video streams and for a fairly congested network the solution could be found in less than 1 second for up to 10 videos, which is a reasonable computation span for real-time systems [84].

Increasing the number of streams further resulted in higher computation times, that might not be suitable to efficiently implement the model in a dynamic network. Based on the observed trends in the results, two solutions are proposed for scenarios where an ISP intends to serve more than 10 video streams using multicast.

For very large networks, a **locally optimal solution** can be found by considering smaller slices of the network and solving the problem for each slice separately, which will result in lower computation times, as seen in Figure 5.7 for the STAR topology. This is practical for large ISPs with national coverage where they manage local areas rather than finding a globally optimal solution for users located all over the country.

A second approach is to find a **near-optimal solution** by solving batches of videos in parallel, instead of all the videos simultaneously. The fraction of resources for each batch of videos will have to be determined prior to solving the parallelized problem and the nearness to the optimal solution will depend on the efficacy of the heuristics used for resource sharing. Such a heuristic algorithm can be designed in a similar way to approaches for other network architectures, such as [12], where a multicast weighting function is proposed for multicast users in LTE networks.

**Handling network congestion**: The performance of Danos optimization model is analyzed by comparing it with mCast (Chapter 4), an approach that enables

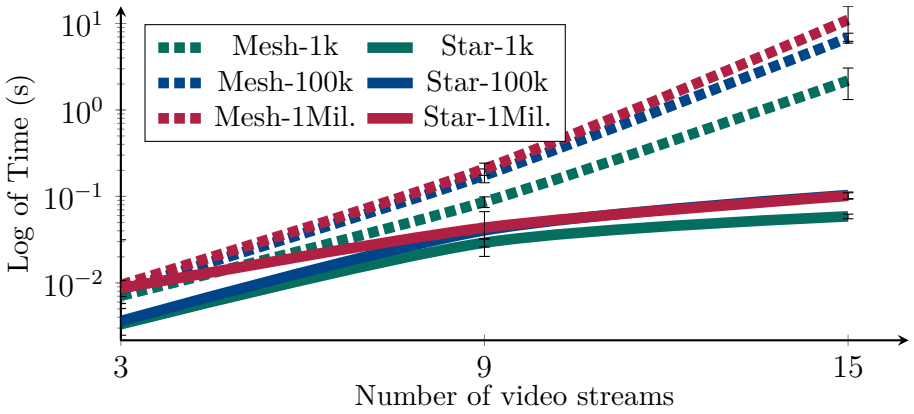



Figure 5.6: Time taken to compute optimal solution by Danos for different topologies, number of video streams and users.

inter-domain network-layer multicast but does not consider network or user details. mCast has a broadly similar architectural view to Danos, thus easing the comparison. The metric for this analysis is the number of users that were assigned the best bitrate based on their specifications i.e. the highest rate which would result in no frame losses due to network congestion or device capabilities. Note that the metrics such as actual lost frames or average goodput cannot be measured for analyzing the optimization model as the model solves the problem at a particular network instance. Such metrics are instead studied in Section 5.5, where an end-to-end streaming service is evaluated over a duration of time.

Even with no congestion in the network, users may experience frame losses if they request a bitrate that exceeds the bandwidth allowed by their ISPs or is higher than what their devices can support. While the Danos optimization model is aware of such details and handles them to ensure no losses at the user-end, other approaches do not. For a fair comparison of mCast with Danos, smart users are considered for the mCast service i.e. users never request a bitrate higher than their device capabilities and also consider any cross-traffic they are receiving to ensure that they do not exceed their allowed ISP bandwidth. With no drops at the user-end, network congestion in the ISP network remains a cause of frame losses for users.

Figure 5.6 shows average results of 20 runs for 1 million users requesting one of 15 available video streams, each served at three different bitrates. To analyze the behavior of Danos and mCast in network congestion, the available capacity is decreased at all the links in the network from $40Mbps$ to $10Mbps$ and the percentage of users that were assigned a bitrate correctly is measured in both STAR and MESH topology. At $40Mbps$ there was little congestion in the network and

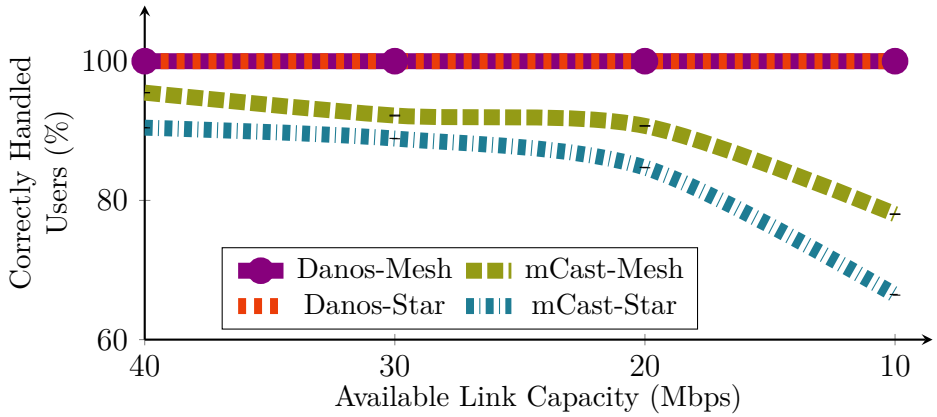



Figure 5.7: Percentage of correctly assigned users vs available link capacity for 1 million users and 15 video streams

mCast served most users correctly (95% for MESH and 90% for STAR topology).

As the link capacities started decreasing and congestion occurred, mCast failed to respond properly and the percentage of correctly handled users kept on decreasing. The users in the STAR topology, with no redundant paths, suffered more than users in the MESH topology. Because of its awareness of device specifications, ISP bandwidth limitations and congestion in the network, Danos was able to handle all such scenarios and avoided any frame losses for users, by assigning the best bitrate that could be delivered to them. Hence, in both MESH and STAR topologies, Danos assigned 100% users with their best bitrates.

In addition to showing the gains of Danos over mCast, these results present another key message. An efficient and well-designed network-layer multicast approach can serve potentially unlimited numbers of users (1 million in this analysis) with very low link capacities in the network. As Figure 5.6 shows, when network links had only $10Mbps$ capacity available for Danos, it still managed to find a solution where none of the million users suffered any frame losses.

## 5.5 Performance evaluation and comparison

A prototype of Danos was implemented on eSMAL (Section 3.3) for demonstration and evaluation. Danos was evaluated and compared with mCast in real-world scenarios including flash crowds and cross-traffic.

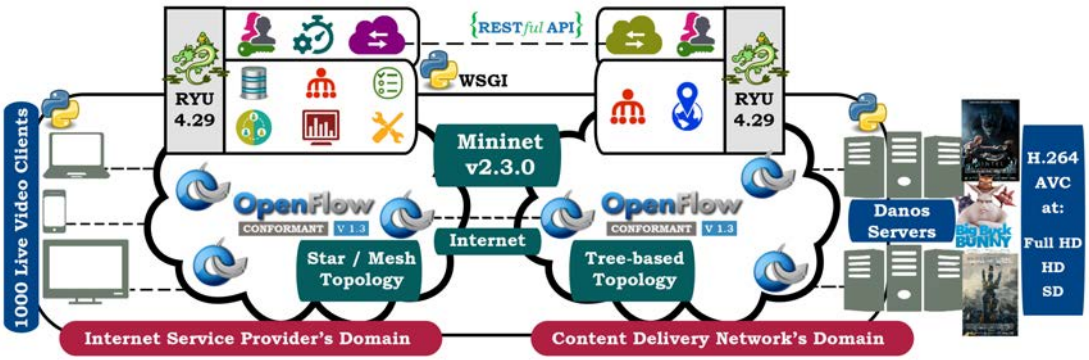



Figure 5.8: Prototype implementation of Danos over an emulated testbed.

### 5.5.1 Prototype implementation

Figure 5.8 shows the prototype implementation of Danos. An SDN-based domain is assumed for both CDN and ISP networks. The Mininet Emulator v2.3.0 [92] is used to emulate network topologies and client devices. For the CDN domain, a simple tree-based topology is used, which is the most common approach for data-centers [110]. For the ISP domain, two residential ISP topologies from the Topology Zoo database [94] are considered, as mentioned in Section 5.4.4.

One ISP ingress switch provides a gateway to the CDN, creating one entry point for all the video streams. The capacity of this link is set to 100Mbps, allowing enough bandwidth for all the videos and bitrates to be delivered to the ISP network with multicast. A scenario is considered where the CDN does not opt to use ISP transrating services and hence the Transrater module is not implemented. If used, it would reduce the traffic flow from the CDN to the ISP.

To implement all the remaining modules of Danos in the application and control plane (Figure 5.1), the Ryu controller [49] is used and the OpenFlow v1.3 protocol is used for the SDN southbound interface. The network forwarding nodes are chosen to be instances of Open vSwitch v2.10[1] switches in both domains with default queue sizes for each link, which was found to be sufficient with $\alpha = 0.8$ (Equation 5.1a), for smooth switching as discussed in Section 5.3.1. The Danos ISP Agent runs as a web server built over the Python Web Server Gateway Interface (WSGI) and a RESTful API is defined for communication between the Danos CDN Agent and the ISP Agent.

The optimization framework runs periodically every 5 seconds and solves the guided optimization model (Section 5.4.3) using the Gurobi solver [109]. For

[1] `https://www.openvswitch.org/`

fairness among users when allocating resources, the Proportional Fairness (PF) metric is calculated as the system utility. PF is defined as the sum-log of bitrates assigned to users i.e. $f(m, r) = log(r_m)$ in Equation 5.1a, where $r_m$ is the bitrate assigned to user $m$.

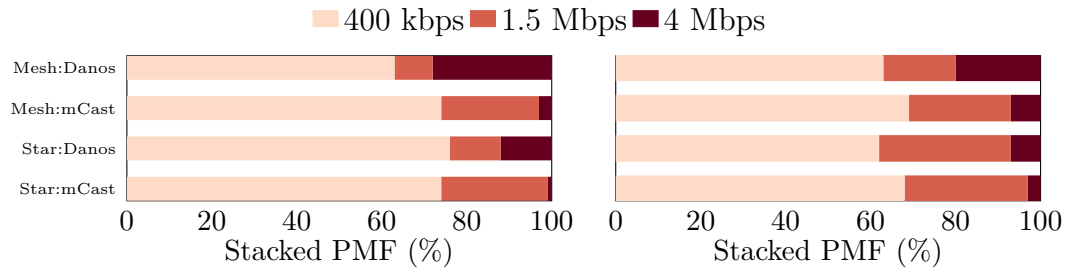
Three raw open-source videos [21] are encoded, "Big Buck Bunny" (bbb), "Tears of Steel" (tos) and "Sintel", using H.264 AVC at three resolutions each, 1920x1080, 1280x720 and 480x270 with a GOP size of eight and a frame rate of $25fps$, yielding approximate bitrates of $4Mbps$, $1.5Mbps$ and $400kbps$ respectively. Three Danos streaming servers are added to the CDN, each streaming one of the video at the three mentioned bitrates.

The Danos servers and video clients are built on Python2.7. The clients send requests to join a video stream at a specific bitrate and rather than live decoding, save a log of the recieved content for post-processing. This allows scaling the number of clients that can run simultaneously in the emulated testbed. One live video client runs on each user created in the Mininet topology and up to 500 clients are tested for evaluation of Danos.

The Danos streaming servers run three threads: a **listening thread** to listen for content requests and establish a connection with clients; a **control thread** that uses an API to communicate with the Danos CDN Agent and takes instructions or provides the requested client information and; a **data thread** that streams a video at different bitrates and serves each client at their maximum allowed or requested bitrate.

The performance of Danos is evaluated in two key scenarios: flash crowd and cross traffic and the results are compared with mCast (Chapter 4) that has a broadly similar architectural view to Danos. The performance metrics include:

- **Lost frames** defined as the number of video frames that were partially or completely dropped in the network
- **Average goodput** which is the rate at which useful data arrived at client devices. Useful data includes the number of bytes in GOPs received completely by a client.
- **Probability Mass Function (PMF)** of the bitrates assigned to users. The PMF serves as an indicator of overall system performance by providing an idea of how many users can receive a certain bitrate in given conditions.
- Regarding the overhead of Danos, the number of **signaling messages** are




(a) Flash-Crowd scenario (b) Cross-Traffic scenario

Figure 5.9: PMF of average user-bitrates for Danos and mCast in two topologies

measured that are shared between the SDN controller and the OpenFlow switches in the ISP network.

### 5.5.2 Flash crowd scenario

The nature of live video streaming, and in particular, rising popularity of user-generated live content has resulted in frequent occurrence of flash crowd events with large numbers of user arrivals in a short period of time. To determine how Danos would react to such events, an ISP setup is assumed that assigns a slice of $12Mbps$ capacity on network *access links* for Danos sessions. Videos sessions last for 10 minutes during which three flash-crowd events occur by adding around 150 clients within a 30-second window at the beginning, middle and towards the end of the stream. The experiment is repeated five times by uniformly varying user locations and device specifications and the average results of the five trials are shown in Figure 5.9a and Figure 5.10.

Users request the highest bitrate that they can support, depending on various device, ISP or CDN constraints. As the number of users increase, with three videos each at three bitrates, this can create up to nine concurrent streams in the access network which would exceed the available link capacities ($12Mbps$). mCast is unaware of the network state and resulted in congesting the network, causing up to 37% video frames lost for users, and affecting the average goodput of the system.

As Danos ran an optimization framework that considered network constraints, it reacted to flash-crowd events efficiently by balancing the load across redundant paths in the MESH topology and eliminating frame losses. This helped Danos to also stream better video quality to its users, in comparison to mCast, as illustrated in Figure 5.9a. In cases of no alternate paths, Danos reduced the bitrates of the

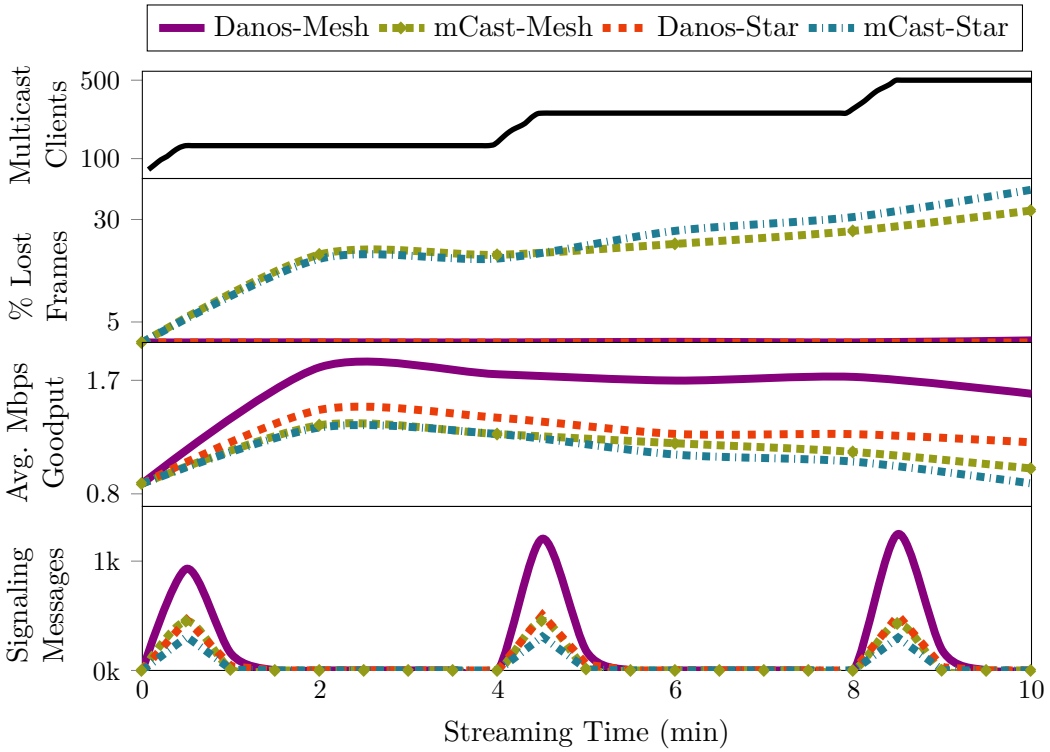



Figure 5.10: Comparison of Danos and mCast when handling flash-crowds.

affected clients to handle congestion. This is evident in the STAR topology where no redundant paths are available and Danos reduced the bitrates (Figure 5.9a), and hence the average goodput of the clients (Figure 5.10), to avoid frame losses.

The cost of active network agents comes in the form of signaling overhead messages. When Danos re-configures network nodes, OpenFlow messages are sent and more messages can be seen than mCast, which only re-configures when a client joins or leaves a stream. However, as shown in results (Figure 5.10), Danos generated around 1000 messages over a 1-minute duration for flash crowd events. This is considered low for SDN switches which can usually handle thousands of flow modification messages per second [111].

### 5.5.3 Cross-traffic scenario

In this experiment, an ISP setup is considered with no link slicing and 30*Mbps* *access links* are shared among 500 Danos users and up to 300 unicast users in the ISP network. All the Danos users request one of the three videos mentioned above, within the first minute of the streaming duration. The arrival of cross-clients is based on a normal distribution. These clients connect randomly to forwarding nodes of the ISP network, uniformly choose 1, 2 or 4Mbps as download rates and

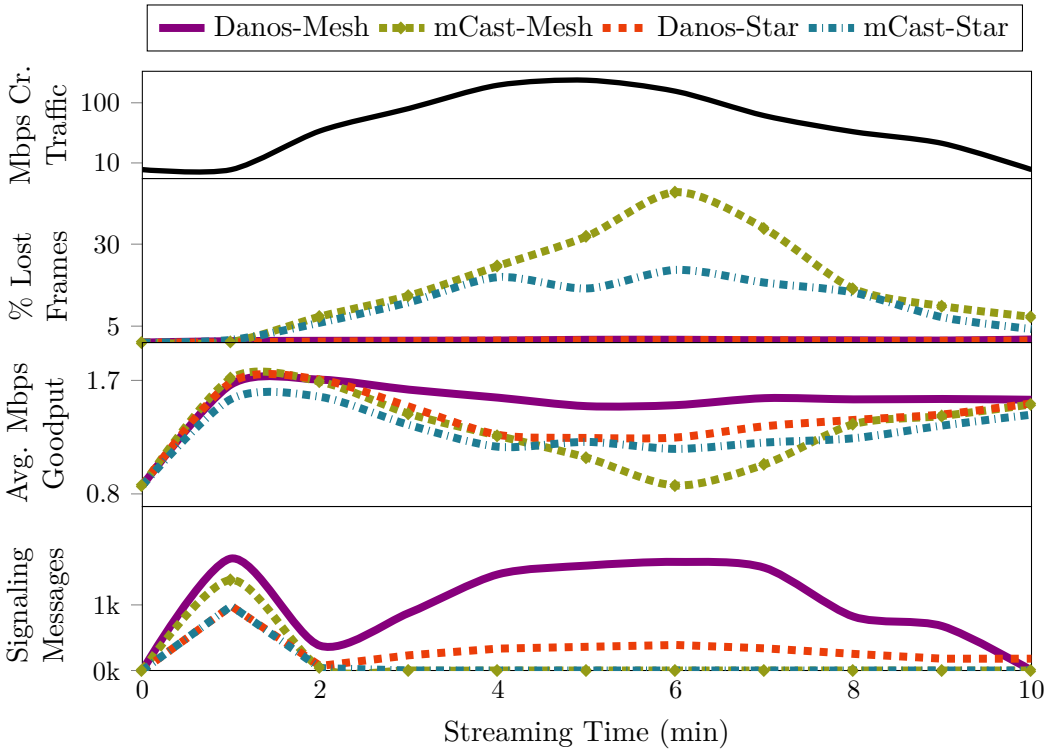



Figure 5.11: Comparison of Danos and mCast when reacting to cross-traffic.

stay active for 1-minute each. The cross-traffic generated in the network is shown in Figure 5.11 and follows a Gaussian distribution which is a commonly adopted model for Internet traffic [112]. The results shown are averaged over 5 trials.

With no or low cross-traffic towards the beginning and end of the streaming duration, both Danos and mCast assigned users with their highest supportable bitrates and induced no frame losses in MESH or STAR topology. As the cross-traffic started increasing and the network links got congested, mCast failed to react and tried to serve multicast users with the same bitrates, resulting in up to 45% frame losses at the peak cross-traffic.

Danos, similar to flash crowd events, responded by rerouting traffic where possible, especially in MESH topology, and reducing user bitrates otherwise. This also resulted in higher goodput rates for Danos where it was capable of serving 20% users with highest and 17% users with medium bitrate for the MESH topology. As the cross-traffic started decreasing, the percentage of lost frames decreased in mCast and the goodput increased. Danos, being aware of the network state, also reacted by further increasing bitrates for users while maintaining no frame losses.

Even with the dynamic configuration of the network, the signaling overhead was reasonable for Danos as shown in the Figure 5.11. Furthermore, due to its stable

response mechanism, Danos did not inflict excessive bitrate switching on clients and the cumulative bitrate switches for all 500 Danos clients was at most 100 over the complete streaming duration of 10 minutes.

## Summary

In this chapter, Danos was presented, an optimal live streaming service that is aware of device and network state and uses the global knowledge of SDN to enable resource-efficient network-layer multicast over the Internet. An architecture design was presented and various design issues and choices for ISPs and CDNs were discussed. An optimization model was formulated that builds optimal paths in real-time and ensures high video quality for users by minimizing frame losses or bitrate switches and maximizing the assigned bitrates. Performance analysis showed that the model is extremely scalable and can solve the problem in order of milliseconds for potentially millions of users. A prototype of Danos was implemented and comparison was conducted against mCast in real-world scenarios such as flash crowds and cross-traffic. Danos handled both events efficiently and improved average goodput by up to 70% while eliminating video frame losses for clients by adapting bitrates according to network congestion.

# Chapter 6

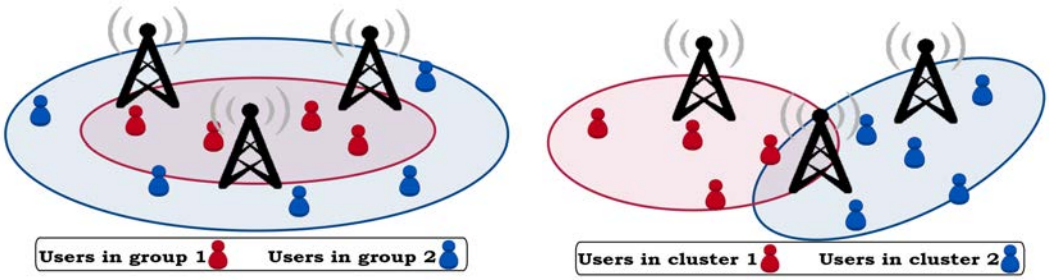
# RTOP: Optimizing SFN clusters and user groups in eMBMS

## 6.1 Introduction

Mobile data traffic is increasing rapidly at a 47% annual growth rate and video accounts for more than 60% of this traffic [1]. Live streaming services like Periscope, Facebook Live and Twitch [2] are becoming more popular, which further elevates the demand for High Definition (HD) video streaming over cellular networks. The delivery of highly popular content using the traditional unicast method leads to inefficient resource utilization and poor user experience.

While network layer multicast [113], such as mCast (Chapter 4) and Danos (Chapter 5), can reduce resource consumption in the backbone, core and wired access network, the wireless last hop, where the resources are scarce, still suffers from redundant unicast transmissions. Recent experiments and trials [114] illustrate that these drawbacks can be alleviated using Evolved Multimedia Broadcast Multicast Service (eMBMS).

eMBMS [11] is a 3GPP standard that enables multicast over the wireless spectrum by grouping users watching the same content and transmitting the content to a group just once. This results in an effective spectrum utilization, particularly when the number of active users is high. Furthermore, to improve the channel condition of User Equipment (UE) devices, eMBMS allows Evolved-Node Base Stations (eNB) in a spatially local area to transmit the same content at a common frequency and time, hence creating a Single Frequency Networks (SFN).




(a) *Synchronize eNBs to form one cluster. Partition users into groups based on their channel condition.*

(b) *Create multiple SFN clusters (two in this case) based on user distribution and available network resources.*

Figure 6.1: Various possible eMBMS configurations.

Interested UEs combine the signal received from each eNB in the SFN, improving their Signal to Interference Noise Ratio (SINR). A Multicast Coordination Entity (MCE) manages the eMBMS users and resource allocation for all eNBs in an SFN (Figure 6.1a).

To ensure that all the UEs can decode the transmitted signal, the Modulation and Coding Scheme (MCS) of a group is restricted to the UE with the worst channel condition. Similarly, synchronizing eNBs in an SFN brings forth two limitations: the eNB with the least available resources limits the amount of Resource Blocks (RB) available for an eMBMS session and users of one eNB with low MCS values can adversely affect users of other eNBs when creating user groups. To overcome these limitations, state-of-the-art solutions propose partitioning eNBs in the eMBMS service area into multiple SFN clusters [115] depending on the user distribution and available RBs at each eNB (Figure 6.1b). Alternatively, users are split into groups based on their channel conditions [15] with each group receiving an appropriate video bitrate (Figure 6.1a).

To maximize the benefits and potential of eMBMS, network operators need a solution that can run in real-time and solve the user grouping and SFN clustering problems. As shown in Section 2.4.3, jointly optimizing these problems can provide higher system utility than separate solutions. The existing models [12, 13, 14] ignore the inter-dependence of these two problems hence yielding sub-optimal results. Also most of these models are either too complex to solve in real-time for a large number of users [15]; do not consider multiple videos served by eMBMS at the same time [12, 13]; aim to maximize the network throughput instead of the application-level video bitrates [12, 14] or; ignore the impact of eMBMS resource allocation on unicast users [16].

In this chapter, a novel scalable resource management framework for eMBMS

is proposed that jointly optimizes resource allocation, SFN clustering and user grouping to maximize an operator-defined utility. The evaluation of the proposed solution showed that the joint optimization achieved up to 14% improvement in the system utility and 90% increase in the average bitrates received by users in comparison to state-of-the-art techniques [12, 14]. Additionally, the proposed framework guarantees a minimum video bitrate for all eMBMS users with most users receiving higher bitrates. The contributions made in this chapter are multi-fold:

- A joint optimization problem is formulated for SFN clustering and user grouping for multiple eMBMS video sessions. The solution of this problem determines the performance bound on a rate-based utility and presents a practical mechanism to handle the impact of eMBMS decisions on unicast users.
- A Real-Time Optimal Partitioning algorithm (RTOP) is proposed. RTOP is a scalable heuristics-based algorithm that computes optimal or near-optimal results for the resource management framework, in real-time, for typical eMBMS settings, independent of the number of users.
- Extensive evaluation is performed with various network configurations, user distributions and number of videos served by eMBMS with different bitrates. The results indicate the benefit of the joint optimization in comparison to state-of-the-art techniques [12, 14]. It is also shown that the utility achievable by RTOP is always within a 1% gap from the globally optimal solution.

## 6.2 Joint optimization model

### 6.2.1 System model

A cellular system is considered, consisting of a set of eNBs $B$ in the eMBMS service area, serving some unicast users and a set of videos $V$ to multicast users in set $M$ using eMBMS. For each video $v$, eNBs $B$ can be grouped to form one or more non-overlapping clusters of SFN. Each video $v$ is encoded at a set of bitrates $R_v$ and the MCE may transmit $v$ at one or more distinct bitrates in each cluster at a chosen Modulation and Coding Scheme (MCS). Note that the minor real-time variations in bitrates are handled by network buffers and for the

Table 6.1: Notations For RTOP optimization model

| Symbol | Description |
|---|---|
| | **INPUTS** |
| $B$ | Set of one or more eNBs in the eMBMS service area |
| $C$ | Set of possible clusters of eNBs (non-empty subsets of $B$) |
| $P$ | Set of all possible eNB configurations i.e. ways to configure eNBs $B$ into non-overlapping SFN clusters |
| $b_{pc}$ | Binary variable to inform if $b$ is in cluster $c$ for $p \in P$ |
| $E$ | Total number of available CQI levels (15 for LTE) |
| $S_e$ | Achievable spectral efficiency from a CQI level $e$ |
| $T$ | Total number of resource blocks available at any eNB |
| $\alpha$ | Maximum fraction of resources allowed for eMBMS |
| $f(r)$ | Operator-defined utility function that takes rate $r$ as input |
| $Y_b$ | Number of RBs requested by eNB $b$ for its unicast users |
| $V$ | Set of videos served by eMBMS in the service area |
| $R_v$ | Set of bitrates available for video $v$ |
| $M$, $M_v$ | Set of multicast users ($M$) subscribed to video $v$ ($M_v$) |
| $M_{vpce}$ | Number of users of video $v$ in cluster $c$ with MCS $e$ when eNB configuration $p$ is used |
| | **VARIABLES** |
| $P_{vp}$ | Binary variable to determine if eNB configuration $p$ has been chosen for video $v$ |
| $X_{vpcr}$ | Number of RBs allocated by eNBs in cluster $c$ of eNB configuration $p$ to video $v$ for bitrate $r$ |
| $M_{vpcer}$ | Binary variable to determine if users of video $v$ with MCS $e$ in cluster $c$ of eNB configuration $p$ are assigned bitrate $r$ |

optimization, an average value is considered over time. Based on the Channel Quality Indicator (CQI) level, a user may select a bitrate, and consequently an MCS, that is best suited for its channel condition.

As each MCS produces a certain spectral efficiency, the MCS chosen by the MCE for a bitrate of a video determines the number of frequency-time Resource Blocks (RB) needed to achieve that bitrate. Each eNB has $T$ RBs which are used to serve both unicast and multicast users. The maximum fraction of RBs allowed for eMBMS is $\alpha$ [116] and each eNB $b$, needs $Y_b$ RBs to serve its unicast users. Hence, the available RBs for eMBMS users at any eNB $b$ equals $\min(\alpha T, T - Y_b)$. Table 6.1 summarizes the notations used in this chapter. In such a system, different configurations of eMBMS are envisioned, leading to different achievable bitrates for users.

Section 2.4.3 presents a detailed example of various possible configurations for a given network scenario (Figure 2.2). These configurations include splitting

eNBs to form one or more SFN clusters for each video and within each cluster partitioning users based on their channel condition to form one or more user groups. The example shows that the total utility of the system depends on the eNB configuration, the number of user groups created, the number of users placed in a group and the number of RBs and bitrates assigned to each user group. To maximize an operator-defined utility, an optimization model is defined that considers all of these factors.

### 6.2.2 Problem formulation

The typical use cases of eMBMS service, such as sporting events, involve a large number of users. Hence, there is a need for scalable optimization framework to identify the performance bounds of resource allocation schemes. Existing work in the literature, e.g. [12, 16], employ optimization variables that increase with the number of users. The solution times grow exponentially as the number of users increases.

In the proposed optimization framework, the dependence on the number of users is eliminated by relying on the fact that there are a limited number of distinct CQI values, e.g., 15 CQI levels in LTE networks and regardless of UE's SINR value, one of the CQI level is assigned to it. This fact is leveraged for reducing the output variables by defining CQI groups per video per cluster. Instead of handling users individually, the optimization problem is formulated to find the optimal bitrate for each CQI group. For each eNB configuration $p$ and video $v$, the number of users that belong to cluster $c$ and report a CQI $e$, are denoted as $M_{vpce}$, and form part of the input to the optimization model. This simple modeling technique reduces the time-complexity and enables finding the optimal solution of the problem in a reasonable time when evaluating the effectiveness of the proposed heuristics.

In addition to eMBMS users, eNBs may also serve unicast users. In practice, an MCE has no control over how many RBs are allocated to a unicast user which is instead handled by the scheduler of the associated eNB. For each eNB $b$, $Y_b$ is taken as an input from the operator. An operator can choose the mechanism to calculate $Y_b$ based on the priority of eMBMS over unicast [14] or the number of unicast and multicast users in a cell, e.g. with a multicast weight function [12].

The optimization problem is formulated with the objective (Equation 6.1a) to maximize an operator-defined utility for all multicast users in all the CQI groups.

This is illustrated in *Problem 1.* $P_{vp}$ is a binary variable to determine whether eNB configuration $p$ has been chosen for video $v$, $M_{vpcer}$ is a binary variable to determine if users of CQI group $M_{vpce}$ are assigned bitrate $r$ and $X_{vpcr}$ is the number of RBs assigned to $r$ for $v$ in cluster $c$ of eNB configuration $p$.

**Problem 1: Optimal eNB configuration and user groups**

$$max \sum_{v \in V} \sum_{p \in P} P_{vp} \cdot \sum_{c \in p} \sum_{r \in R_v} \left( f(r) \cdot \sum_{e=1}^{E} M_{vpcer} \cdot M_{vpce} \right) \tag{6.1a}$$

*subject to*

$$\sum_{p \in P} P_{vp} = 1, \ \forall v \in V \tag{6.1b}$$

$$\sum_{r \in R_v} M_{vpcer} \leq 1, \forall e \in \{1, 2, .., E\}, v \in V, c \in p \in P \tag{6.1c}$$

$$\sum_{p \in P} \sum_{c \in p} \sum_{r \in R_v} \sum_{e=1}^{E} M_{vpcer} \cdot M_{vpce} = M_v, \ \forall v \in V \tag{6.1d}$$

$$X_{vpcr} \geq \max_{e} \left( \frac{M_{vpcer} \cdot r}{S_e} \right), \forall r \in R_v, c \in C \tag{6.1e}$$

$$\sum_{v \in V} \sum_{p \in P} \sum_{c \in p} \sum_{r \in R_v} b_{pc} \cdot X_{vpcr} \leq \min \left( \alpha T, T - Y_b \right), \forall b \in B \tag{6.1f}$$

Constraint 6.1b ensures that each video chooses only one eNB configuration. Constraint 6.1c and 6.1d guarantee that each CQI group (and hence user) is assigned one and only one bitrate. Constraint 6.1e ensures that the number of RBs used to transmit a video bitrate in a cluster are enough to decode it properly for users of any CQI group assigned that bitrate. Constraint 6.1f limits the total RBs used by eMBMS at any eNB to what's left after satisfying unicast resource requests by each eNB. It also limits the percentage of RBs allowed for eMBMS to $\alpha$, which is usually set to 60% [116] in LTE networks. This constraint can be tuned or relaxed in extremely congested networks to adjust the amount of resources that an operator wants to allow for unicast users and leave available for eMBMS users.

### 6.2.3 Time scale of optimization

The proposed problem formulation is a Linearly-Constrained Quadratic Program (LCQP) with all integer (mostly binary) variables and a global maximum bound (at highest bitrate for all users), hence it is NP-complete [117]. However, even with the notion of CQI groups, when practically computing the optimal solution,

the numerous possible eNB configurations and the dependence of each video's choice on other videos makes the problem complicated and time consuming to solve. Based on the experiments (Figure 6.5c), it can take up to 100s to compute the global maximum, which is not sufficiently fast to operate in real-time. Hence, an efficient heuristic-based algorithm is proposed that can find an optimal or near-optimal solution in real-time.

## 6.3 Real-time heuristics-based algorithm

The joint optimization problem involves four inter-dependent decisions to make:

1. Identifying an eNB configuration for each video and assigning users to the best cluster of that configuration
2. Creating user groups based on channel conditions
3. Assigning an appropriate video bitrate to each group
4. Allocating RBs to various videos and underlying groups

This section first presents how these decisions are taken when one video is served by eMBMS and then explains the additional steps needed to handle multiple video scenarios.

### 6.3.1 Handling a single video

In a single video scenario, all the eMBMS resources in the service area will be accessible to the video. Hence, the total utility mainly depends on the eNB configuration and underlying user grouping for that video.

#### 6.3.1.1 eNB configurations

Each eNB configuration is completely defined by identifying groups of eNBs acting as SFN clusters and assigning users to these clusters. The number of possible eNB configurations is equal to the Bell number[1] of the eNBs in the service area. Figure 6.2 shows an example scenario for a video with four eNBs (1,2,3,4), where eNB4 has higher unicast load (85%) than other eNBs (55%). A total of 100 users are interested in the video and the eNBs produce different spectral efficiency

[1] `http://mathworld.wolfram.com/BellNumber.html`

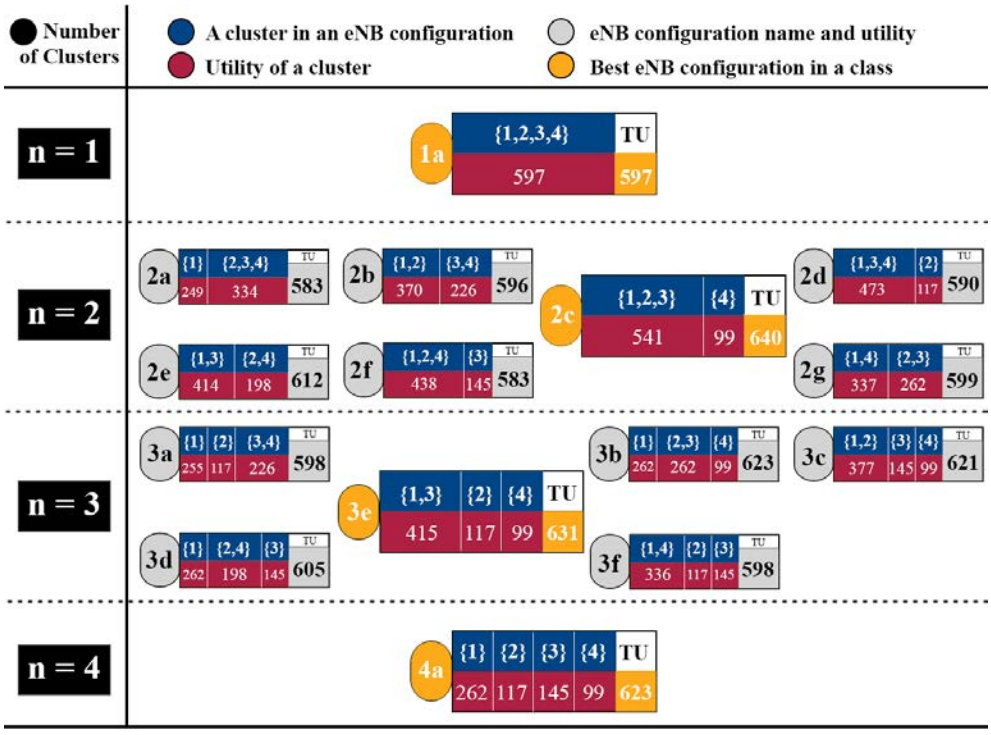



Figure 6.2: Heuristic example. *A candidate eNB configuration is chosen in each class. eNB4 is more congested than other eNBs, so configurations with 4 in a separate cluster perform better.*

values for the users when clustered differently. There are 15 possible eNB configurations (fourth Bell number) ranging from one cluster (all eNBs synchronized as a single SFN) to four clusters (each eNB in a separate cluster). The configurations are divided into $|B|$ classes defined by the number of clusters in each configuration. These classes are more relevant to the multiple video scenario and will be discussed in Section 6.3.2.

In each configuration, to maximize the system utility, users are assigned to the cluster that provides them with higher MCS values. In the single video case, the user grouping is explored for all the possible eNB configurations. Note that in general, eMBMS is used in a limited number of neighboring eNBs [116] serving a highly populated area.

#### 6.3.1.2 User grouping

For an eNB configuration, users in each SFN cluster can have disparate channel conditions. User grouping would enable improving the total utility by splitting users into groups based on their channel conditions and assigning an appropriate video bitrate to each group. On doing so, the achievable utility depends on the

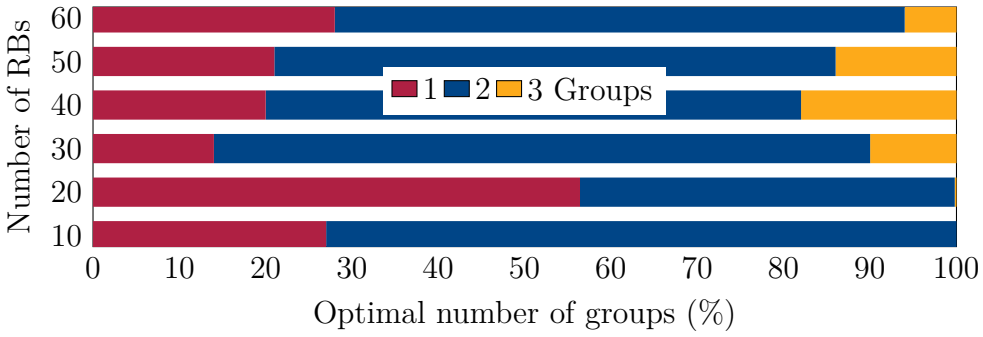



Figure 6.3: Number of user-groups to achieve optimal utility from different system configurations.

number of users in each group, the minimum user-MCS value per group, and the RBs available to each group. Exhaustively searching through all possible groups of $M_v$ users takes $\mathcal{O}(|R_v| \cdot |M_v|^{|M_v|})$ to solve. Hence, there is a scalability issue, especially for large number of users. This issue is resolved by creating CQI groups as explained in Section 6.2.2 and deciding which CQI groups should aggregate to form a user group.

Theoretically, the maximum number of user groups equals the number of distinct video bitrates available. However splitting users in too many groups reduces the share of RBs per group, and may reduce the achievable bitrate by a group, and hence the total utility. To analyze this behavior, simulations were conducted by distributing 10, 100 or 1000 users in the service area and varying the number of available RBs. Each setup was repeated 1000 times. Results (Figure 6.3) show that in 90% of the cases, the optimal utility was achieved by one or two user groups. Therefore, a user grouping algorithm is designed that aggregates CQI groups and either places all users in one group or creates two user groups.

Algorithm 1 presents the user grouping algorithm and involves three main steps. First, it identifies the number of RBs needed to assign a bitrate to the first (lowest) CQI group and calculates the utility achievable by placing all users in one group (Line 1-7). Then it measures the throughput that can be achieved by higher CQI groups with the remaining RBs (Line 8-9). If some CQI groups have enough throughput to support the next bitrate then the algorithm calculates the utility for splitting users in two groups and assigning the higher bitrate to those CQI groups (Line 10-19). The process is repeated for all bitrates and the user grouping option with the maximum utility is chosen as the optimal user grouping. This approach takes only $O(|R_v|^2)$ to solve.

The utility of each eNB configuration is found by running Algorithm 1 on each of

**Algorithm 1** User grouping algorithm

**Input:** $RBs$, $R_v$, CQI Groups in cluster (in ascending order) with CQI values $Q$ and number of users $N$

**Output:** Best Utility $U_{max}$ (initialized with 0), User Groups $G$ with number of users and RBs for each bitrate

1: **for** $i \leftarrow 0$ *to* $|R_v|$ **do**
2: $r1 \leftarrow R_v[i]$
3: $r1RBs \leftarrow r1$ / $Q[0]$ ▷ *Lowest CQI*
4: **if** $r1RBs > RBs$ **then** *break* ▷ *Can't increase rate*
5: $U \leftarrow f(r1) \times sum(N)$ ▷ *f * No. of users*
6: **if** $U > U_{max}$ **then** $U_{max} \leftarrow U$
7: $G[r1, users] \leftarrow sum(N)$; $G[r1, rbs] \leftarrow r1RBs$
8: $r2RBs \leftarrow RBs - r1RBs$ ▷ *Remaining RBs*
9: $thr \leftarrow [r2RBs \times cqi$ **for** $cqi$ *in* $Q[1:]]$
10: **for** $j \leftarrow i+1$ *to* $|R_v|$ **do** ▷ *for bitrates higher than* $r1$
11: $r2 \leftarrow R_v[j]$
12: $k \leftarrow index\ of\ first\ CQI\ group\ with\ thr \geq r2$
13: **if** *no* $k$ **then** *break* ▷ *Can't increase 2nd group's rate*
14: $G1 \leftarrow sum(N[0:k])$; $G2 \leftarrow sum(N[k+1:])$
15: $U \leftarrow f(r1) \times G1 + f(r2) \times G2$
16: **if** $U > U_{max}$ **then** $U_{max} \leftarrow U$
17: $G[r1, users] \leftarrow G1$; $G[r1, rbs] \leftarrow r1RBs$
18: $G[r2, users] \leftarrow G2$; $G[r2, rbs] \leftarrow r2RBs$

its clusters. The configuration with the highest utility is the optimal choice and the eNBs can be configured to form clusters accordingly. Results obtained from Algorithm 1 tell the number of groups to create in each cluster, number of users to place in each group and also the bitrate and number of RBs to assign to each group.

### 6.3.2 Handling multiple videos

In multiple video scenarios, the resources must be distributed optimally among all videos and underlying groups. Such distribution has an impact on the choice of eNB configuration and user grouping. Hence, the problem of choosing an eNB configuration for each video is combinatorial in nature and can result in exponentially increasing outcomes. To solve the problem in real-time, first the choices of eNB configurations are narrowed down for each video to a subset of candidate configurations. Then, the best combination of configurations is identified for the videos. Finally, the optimal resource allocation and user grouping for each video in their chosen eNB configurations is determined.

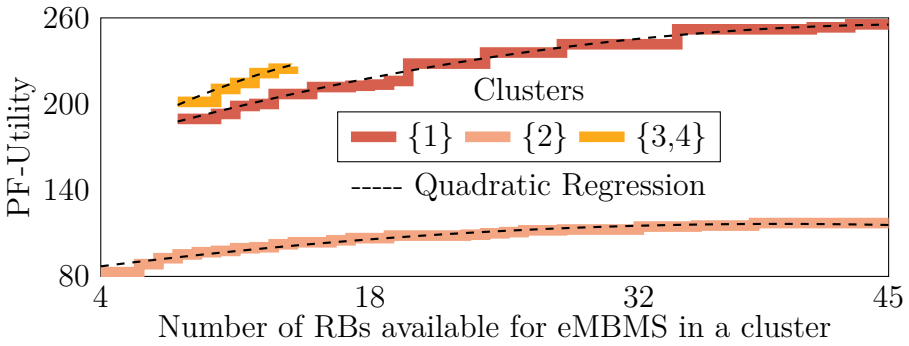



Figure 6.4: A sample Utility vs RB graph with quadratic regression. *For any cluster, the lower bound of RBs is the minimum RBs needed to satisfy all users of a video and the upper bound is the RBs available for eMBMS in that cluster.*

#### 6.3.2.1 Candidate eNB configurations

For each video, the process of Section 6.3.1 is used to obtain the maximum utility of each eNB configuration based on the total available eMBMS resources. Additionally, each configuration is classified based on the number of clusters in it and the best configuration for each class is chosen as a candidate configuration. Continuing with the example in Figure 6.2, the eNB configurations are divided into four classes and the configuration with the highest utility in each class is picked. This narrows down the possible candidate configurations to four (1a, 2c, 3e, 4a) for the considered video. This process is repeated for all the videos served by eMBMS in the service area to obtain a set of candidate eNB configurations for each video.

#### 6.3.2.2 Combination of eNB configurations

In this step, one eNB configuration is selected for each video from the previously identified candidates by maximizing an approximated utility function. First, the user grouping algorithm (Algorithm 6.3) is run on each cluster of an eNB configuration, to identify the achievable utility with up to two user groups per video. This step is repeated for all possible values of RBs, which can vary from one to the maximum RBs available for eMBMS in that cluster, i.e., $min(\alpha T, T - Y_b)$.

Figure 6.4 shows the achievable utility as a function of available RBs for one eNB configuration (3a from Figure 6.2), which consists of three clusters ({1}, {2} and {3,4}). The utility for each cluster increases as a step function and is calculated from a lower bound (minimum RBs needed to serve all users) to an upper bound (maximum RBs available to the video in the cluster). The process is repeated for

all candidate eNB configurations of each video in $V$.

RTOP then proceeds to find the best combination of eNB configurations for different videos using the measured utility data. To speed this search, a simplified optimization problem is defined with an approximated objective utility based on quadratic regression of the utility values, as shown in Figure 6.4. Quadratic regression is chosen as it provided a sufficiently accurate and fast solution in comparison to the less accurate linear regression and the slower higher-degree polynomials.

**Problem 2: Regression-based optimal utility calculation**:

$$max \sum_{v \in V} \sum_{c \in P_v} i_{vc} \cdot X_{vc}^2 + j_{vc} \cdot X_{vc} + k_{vc} \tag{6.2a}$$

*subject to*

$$\sum_{v \in V} \sum_{c \in P_v} b_{pc} \cdot X_{vc} \leq \min\left(\alpha T, T - Y_b\right), \forall\, b \in B \tag{6.2b}$$

$$X_{vc} \geq L_{vc}, \forall\, v \in V, c \in C, \tag{6.2c}$$

where $X_{vc}$ represents the RBs allocated to video $v$ in cluster $c$, $L_{vc}$ is the lower bound for $X_{vc}$ and $i_{vc}, j_{vc}, k_{vc}$ are the quadratic coefficients of the utility for video $v$ in cluster $c$.

Constraint 6.2b is similar to Constraint 6.1f and limits the available RBs in a cluster for eMBMS users. Constraint 6.2c ensures that RBs allocated to a $v$ in $c$ are enough to attain the lowest bitrate of $v$. Solving *Problem 2* gives the maximum achievable utility from a particular combination of eNB configurations of all the videos. This problem is solved for all combinations of candidate configurations and the combination with the highest utility is picked as the final combination for eNB configurations.

#### 6.3.2.3 Resource allocation and user grouping

With an eNB configuration chosen for each video, the optimal resource allocation and the user grouping can be found by backtracking the solution. The resource allocation problem is solved (Equation 6.2a) one more time for the chosen combination of eNB configurations, but instead of using the quadratic regression, the actual discrete utility values (e.g. Figure 6.4) are used. This gives the optimal RB share for users in all clusters subscribed to each video. These values are then sent as an input to user grouping algorithm (Algorithm 6.3) to define user groups,

**Algorithm 2** Complete RTOP algorithm

**Input:** $S_{vpc}$: CQI Groups in cluster $c$ of eNB configuration $p$ for video $v$; See Table 6.1 for all other inputs

**Output:** *Groups* with no. of users and RBs for each bitrate

1: **for** $v \in V$ **do**
2: **for** $p \in P$ **do**
3: $U[v,n,p] \leftarrow 0$ ▷ *n is # of clusters in p (Figure 6.2)*
4: **for** $c \in p$ **do** ▷ *For each cluster in p*
5: $rb \leftarrow min(\alpha T,\ T\text{-}Y_b$ **for** $b \in c)$ ▷ *Available RBs*
6: $utility, _ \leftarrow$ **Algorithm 1**$(rb, R_v, S_{vpc})$
7: $U[v,n,p]$ += $utility$
8: **if** $|V| == 1$ *and* $U[v,n,p] > U_{max}$ **then**
9: $U_{max} = U[v,n,p];\ Optimal_Solution = (p)$
10: **else if** $U[v,n,p] > maxU[v,n]$ **then**
11: $maxU[v,n] \leftarrow U[v,n,p]$
12: $Candidates[v,n] \leftarrow p$ ▷ *For class n*
13: **if** $|V| == 1$ **then** *go to* ***Line* 24** ▷ *Only one video*
14: **for** $p \in Candidates[v]$ **do** ▷ *eNB configurations for v*
15: **for** $c \in p$ **do** ▷ *Get Graphs for each cluster in p*
16: $rb \leftarrow min(\alpha T,\ T\text{-}Y_b$ **for** $b \in c)$ ▷ *Available RBs*
17: $Graph[v,p,c] \leftarrow$ **RBGRAPH**$(R_v, S_{vpc}, rb)$
18: $Cartesian \leftarrow (p_1, .., p_v) \mid p_v \in Candidates[v]$ **for** $v \in V$
19: $U_{max} \leftarrow 0$
20: **for** $solution \in Cartesian$ **do** ▷ *A combination of p's*
21: *Get Utility U with regression from Equation 6.2a*
22: **if** $U > U_{max}$ **then**
23: $Optimal_Solution \leftarrow solution; U_{max} \leftarrow U$
24: *Get Optimal_RBs for each cluster in Optimal_Solution without regression (Section 6.3.2)*
25: **for** $v \in V$ **do** ▷ *Backtrack to find the optimal user groups*
26: **for** $c, rb \in Optimal_RBs[v]$ **do**
27: $_, Groups[v,c] \leftarrow$ **Algorithm 6.3**$(R_v, S_{vpc}, rb)$

**RBGRAPH**$(R_v, S, maxRBs)$ ▷ $S \rightarrow$ *CQI groups*
28: **for** $rb \leftarrow 1$ *to* $maxRBs$ **do**
29: $Utilities[rbs], _ \leftarrow$ **Algorithm 6.3**$(rbs, R_v, S)$
30: **return** $Utilities$

allocated RBs per group and assigned bitrates. The MCE can now configure the eNBs to create video-specific SFN clusters and transmit different bitrates with the chosen MCS values on a certain set of RBs. The total complexity of RTOP is $\mathcal{O}(|V|\cdot|B|\cdot|T'|\cdot|R_v|^2 + |B|^{|V|})$.

#### 6.3.2.4 Complete RTOP algorithm

The complete RTOP algorithm is presented in Algorithm 2 and consists of three main steps. First, an approximate achievable utility (Line 7) is calculated for each video (Line 1) from all possible eNB configurations (Line 2), assuming that the video has access to all the RBs in a cluster and solving Algorithm 1 for each cluster (Line 4-6). Based on the highest approximate utility achievable from each class of eNB configuration, a set of candidates is selected (Line 8-12). Then, Utility vs RB graphs are calculated for each candidate configuration by running Algorithm 1 for all possible RBs value in each cluster of an eNB configuration (Line 14-17).

Then, the possible combinations of video-specific candidates are computed by taking Cartesian product of the sets of each video's candidate configurations (Line 18). Note that this step is not needed if there is only one video served in the eMBMS area (Line 13). The optimization model in Equation 6.2a is solved for each possible combination of video configurations using regression-based Utility vs RB graphs (Line 20-21). The combination with highest utility is chosen as the optimal eNB configuration for each video (Line 22-23). Finally, the solution is backtracked to find best user groups, RBs per group and MCS of each group using Algorithm 1 (Line 24-27).

## 6.4 Performance evaluation and comparison

For evaluation, RTOP uses the testbed presented in Section 3.4. In this section, first the simulation setup is presented, followed by the results of performance evaluation and comparison.

### 6.4.1 Simulation setup

An eMBMS service area is considered, consisting of up to five eNBs arranged in a hexagonal grid. Users are distributed normally or uniformly in the service area. The commonly used LTE parameters [65, 116] are applied to the simulation setup as listed in Table 6.2. A normal distribution represents cases such as sporting events or concerts where most of the users are located in the center of the service area. A uniform distribution represents cases such as shopping malls where users are evenly located across the service area.

Table 6.2: RTOP simulation parameters

| **Parameter** | **Value** |
|---|---|
| Cellular Layout | Hexagonal grid with up to 5 eNBs |
| Cell radius | 500 m |
| eNB Tx Power | 20 Watts |
| Carrier Frequency | 2.1 GHz |
| System Bandwidth | 20 MHz |
| Number of RBs in 20MHz | 100 |
| Path Loss Model | Log-Normal Shadowing n=4 (Urban) |
| White Noise Power Density | -174 dBm/Hz |
| User UE Noise Figure | 7 dB |
| Number of Users per Video | 30 to 150 |
| DASH Video bitrates [21] | 375, 750, 1750, 3000 and 4300 kbps |
| Channel Model | Multi-path Fading AWGN [99] |
| Spectral Efficiencies (bits/RB) from CQIs 1 to 15 | [20, 31, 50, 79, 116, 155, 195, 253, 318, 360, 439, 515, 597, 675, 733] |
| Simulation Laptop Specs. | Dual Core Intel i7-5500U, 16GB RAM |

UEs calculate Reference Signals Received Power (RSRP) from each eNB, measure the achievable SINR from various possible clusters and report the best CQI for each cluster based on AWGN BLER vs SINR curves [99] with 1% error margin. As mentioned in Section 3.4, the target BLER is set to 1% in eMBMS [116], contrary to 10% in unicast, as there are no physical layer re-transmissions.

For the system utility, Proportional Fairness (PF) metric is chosen. PF is a widely-used utility [12, 14, 16] for measuring system fairness and efficacy. Based on PF-utility, the performance of the following approaches is compared:

- **Optimal**: Optimal results of the optimization model (Section 6.2) that considers both SFN clustering and user grouping. To solve the model, the Gurobi Optimizer [109] is used.
- **BoLTE** [14]: As mentioned in Chapter 2 (Section 2.4.2), BoLTE creates SFN clusters to maximize PF-utility but does not consider user grouping. The proposed algorithm assumes single bitrate per video. For fair comparison, the same heuristics is used to calculate the utility of an eNB configuration, but by assigning the best achievable bitrate in each video cluster for all videos.
- Variable Groups (**VG**) [12]: As discussed in Section 2.4.2, VG creates user groups to maximize the PF-utility but does not consider SFN clustering. Also, the proposed algorithm solves the resource allocation problem for only

a single video.

- One Large SFN (**LSFN**): A scheme that considers only user grouping (no SFN clustering) based on the proposed optimization model and constraints. VG is replaced with LSFN when simulating scenarios with multiple videos to analyze the improvement of joint optimization in comparison to only creating user groups.
- **RTOP**: The proposed heuristics (Section 6.3).

The performance of these strategies is evaluated in two key scenarios: A generic scenario with multiple videos and a mega event scenario with one video. In both cases, videos are encoded at five different bitrates, as shown in Table 6.2. For mega event scenario, the impact of varying available resources for eMBMS is also studied.

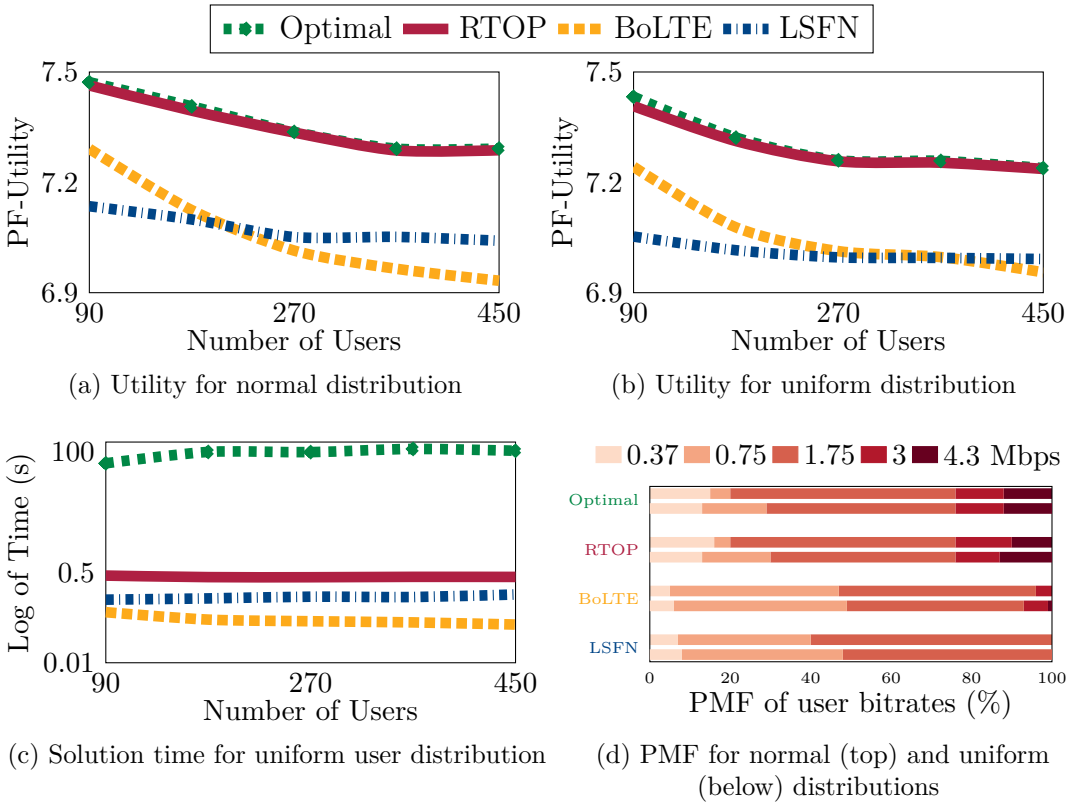
The key performance metrics include:

- **PF-Utility** of the system: PF is defined as the sum-log of bitrates assigned to all the users.
- **Probability Mass Function (PMF)** of the bitrates assigned to users: The PMF serves as an indicator of overall user experience by providing an idea of how many users can receive a certain bitrate in given conditions.
- **Degraded users**: Users with throughput less than the lowest available bitrate. Such users will not be able to receive video without losses or stalls and hence are regarded as degraded users.
- **Solution time**: The time taken by an algorithm to converge and find a solution to the resource allocation problem.

The impact of various network parameters and algorithms on these metrics is analyzed. For each configuration, the experiment is repeated 25 times by varying the user topology. The presented metrics are based on the average of these runs.

### 6.4.2 Generic scenario: Multiple videos

Three videos are streamed using eMBMS in a service area, where 1 eNB has higher unicast load (75 RBs) than other eNBs (50 RBs). Networks with 2, 3 4 and 5 eNBs were tested, and the trends in the results are consistent among the tested scenarios. This section presents results for 4 eNBs. Since VG cannot




(a) Utility for normal distribution

(b) Utility for uniform distribution

(c) Solution time for uniform user distribution

(d) PMF for normal (top) and uniform (below) distributions

Figure 6.5: RTOP comparison for generic scenario with multiple videos.

handle multiple videos, LSFN is considered as a reference to analyze the benefits of SFN clustering.

#### 6.4.2.1 System utility

Figures 6.5a and 6.5b plot the system utility for normal and uniform user distribution, respectively. The figures illustrate that the proposed heuristics achieved optimal or near-optimal utility with a gap less than 1%. Additionally, the figures show that in comparison to LSFN and BoLTE, RTOP improves the utility by up to 8% which is achieved by an increase of average bitrate by up to 50%. Note that as LSFN does not consider SFN clustering, the available RBs for eMBMS were restricted by the eNB with least resources (i.e. 25 RBs) and all the users had to be satisfied with these RBs. On the other hand, BoLTE lacks user grouping and could not assign rates to users commensurate to their channel conditions.

For small number of users, a slight gap between RTOP and optimal results can be noticed, especially for uniform user distribution (Figure 6.5b). This is because, in such cases, the difference between the achievable utility from different eNB configuration can be very low and might not be detected by RTOP, as it uses

heuristics and quadratic regression for faster approximation. However, the difference in utility was marginal and solutions chosen by RTOP were always within a 1% gap from the optimal solution.

#### 6.4.2.2 PMF of assigned bitrates

Figure 6.5d shows the PMF for normal and uniform user distributions. RTOP assigned bitrates in almost the same manner as optimal results, unlike BoLTE or LSFN that allocated lower bitrates to users. The figure shows that RTOP assigned bitrates to users proportionate to their channel conditions and hence almost 75% of users received a bitrate of 1.75Mbps or higher. On the contrary, this ratio dropped to around 50% for both BoLTE and LSFN.

#### 6.4.2.3 Solution time

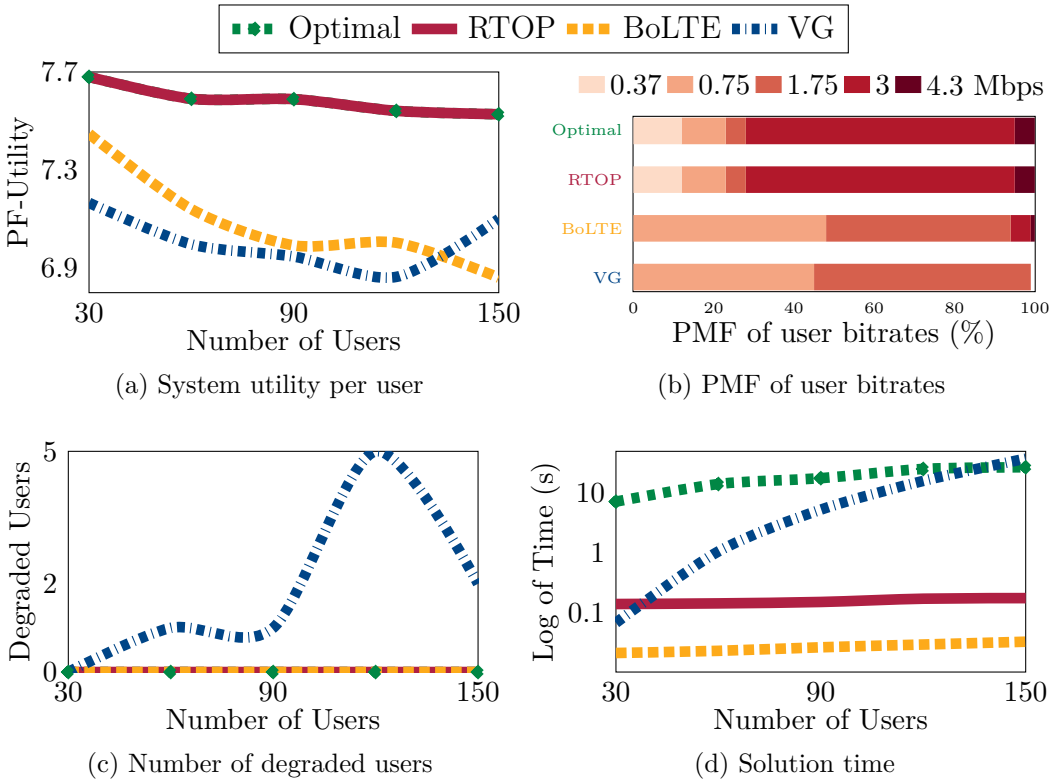
Figure 6.5c plots the time taken to compute the final solution for uniform user distribution. Solving the problem optimally took around 100 seconds which is practically infeasible to implement in a real-time network. RTOP was able to consistently solve the same problem in 500ms, which is well within the limits of expected time constraints [84]. BoLTE and LSFN solved the problem faster but at a much lower utility as indicated above.

### 6.4.3 Mega event scenarios

A mega event refers to highly popular live events, such as world cup finals [118]. In this section,the performance of RTOP is analyzed for a mega event that is transmitted to users distributed normally in a 5-eNB service area where 1 eNB has a higher unicast load (90 RBs) than the other 4 eNBs (75 RBs).

#### 6.4.3.1 System utility

Figure 6.6a shows the system utility of various algorithms. Similar to the generic scenario (Section 6.4.2), RTOP outperformed VG and BoLTE by increasing the utility up to 14% and average user bitrate up to 90%. Moreover, the solution of RTOP was identical to the optimal case. This is because the video had access to all the RBs available for eMBMS and RTOP did not need to perform regression for faster approximation.




(a) System utility per user

(b) PMF of user bitrates

(c) Number of degraded users

(d) Solution time

Figure 6.6: RTOP comparison for mega-event scenario

#### 6.4.3.2 Degraded users

While VG places each user in a group, it does not ensure that each group gets enough RBs to achieve at least the minimum bitrate. Hence, the throughput of a group may become very low. Users with very low throughput are likely to experience frame skipping and video stalls. Figure 6.6c plots the number of such degraded users. With VG, in some cases up to 4% users were degraded. The optimization model and RTOP define a constraint on the minimum allocated throughput to ensure a smooth streaming experience and eliminate drops or losses. BoLTE does not explicitly define such a constraint, however the implementation of BoLTE for this comparison assigns the best possible bitrate based on user-throughput in a cluster, and hence users can get lower bitrates if throughput is low, therefore no users were degraded.

#### 6.4.3.3 PMF of assigned bitrates

Figure 6.6b shows the PMF of user bitrates. As the users were distributed normally, most of the users were in the center of the service area where the channel conditions were good. However, the number of users that received high bitrates

of 3 or 4.3Mbps was only 6% with BoLTE and 0% with VG, which was unfair to users with good channel conditions. With RTOP this ratio increased to 72% leading to the highest utility.

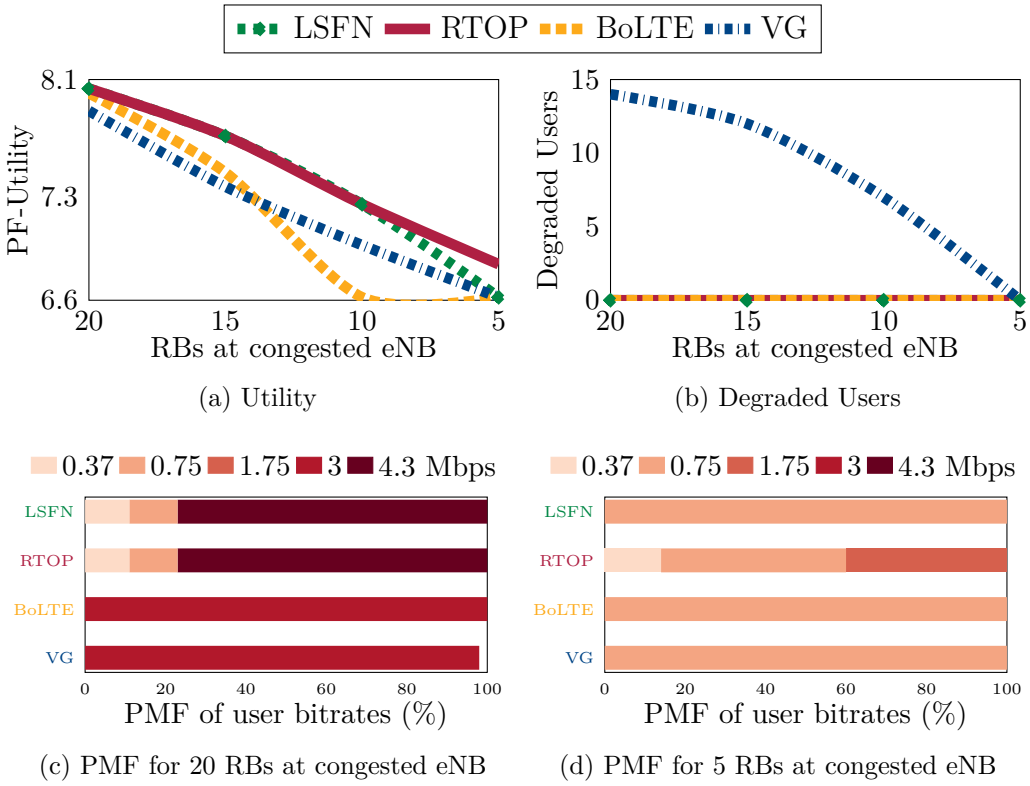
#### 6.4.3.4 Solution time

Figure 6.6d plots the time taken to compute the final solution. As the number of users increased, the computation time for VG increased exponentially, which makes it unsuitable for mega events with large number of users. The proposed optimization model (Section 6.2) is independent of number of users and instead depends on number of CQI groups, which is usually limited to 15 values in LTE. However, solving the model optimally still took more than 10 seconds which is not ideal for dynamic operating conditions. RTOP solved the same problem within 20ms for any number of users in the network. Hence, RTOP can be used in a dynamic network to accommodate changes in the network and solve the resource allocation problem in real-time.

### 6.4.4 Impact of available resources on various metrics

This section explores the impact of available resources at eNBs on the performance of various algorithms, in case of a mega event. As the RBs available to eMBMS decrease, subject to their channel conditions, the number of users receiving high bitrates decreases. An eMBMS service area is considered, comprising of 5 eNBs with 20 RBs for eMBMS. The impact of decreasing available RBs at one eNB is analyzed, when there are 1000 users normally distributed in the service area. Since RTOP always achieves the same result as the optimization model in a single video scenario, for clarity, the optimization model results are not displayed in the figures.

When there were 20 RBs available at each eNB, the best decision was to place all eNBs in one cluster and split users in two groups. LSFN and RTOP achieved maximum utility (Figure 6.7a) by making the right decisions. As VG is agnostic to video bitrates, it maximizes the system utility without assigning enough RBs to the user group with bad channel conditions. Hence, VG resulted in some degraded users (Figure 6.7b). BoLTE also achieved lower utility than RTOP as it is incapable of creating user groups and placed all users located in a cluster in one group.




(a) Utility (b) Degraded Users

(c) PMF for 20 RBs at congested eNB (d) PMF for 5 RBs at congested eNB

Figure 6.7: Impact of RBs available for eMBMS.

As the available RBs on the highly congested eNB decreased, the utility started dropping. Until this eNB had 10 RBs available for eMBMS, the best decision was to keep all eNBs in one cluster and hence LSFN and RTOP achieved the same utility by creating one cluster and grouping users based on their channel conditions. Although VG can also create user groups, most of the times it achieved a lower utility due to a lack of a constraint of serving all the users with a bitrate. BoLTE being unable to create user groups achieved lower utility as well.

As the available RBs decreased further, separating the highly congested eNB in a second SFN cluster was the better decision. RTOP made this decision and achieved optimal utility but LSFN being unable to create SFN clusters, kept all eNBs synchronized and achieved lower utility than RTOP. RTOP outperformed VG and BoLTE in these cases as well. Figure 6.7d shows the PMF of user bitrates at 5 RBs. While other algorithms served all the users with low bitrates, RTOP managed to serve 40% of users with medium bitrates even with only 5 RBs available to eMBMS at the highly-congested eNB.

## Summary

In this chapter, the joint optimization of user grouping, SFN clustering and resource allocation was considered in an eMBMS network. An efficient and scalable heuristic-based algorithm, RTOP was developed, that finds optimal or near-optimal results in real-time with no more than a 1% gap from the optimal solution. Additionally, RTOP was compared with state-of-the art techniques by conducting extensive simulations. In situations where other approaches could assign high bitrates to less than 10% users, RTOP was able to assign high bitrates to 75% of the users. Overall the proposed algorithm improved the system utility by up to 14% and the average user bitrates by up to 90% while avoiding service degradation for the users. The results also showed the advantage of multicast-based transmission modes. A well-designed resource allocation framework, such as RTOP, could serve hundreds of users with as low as 5 RBs.

# Chapter 7

# NIMBLE: Network optimization for eMBMS live video QoE

## 7.1 Introduction

Using unicast transmission mode for delivering live video streams in a dynamic mobile environment [10] leads to wasteful resource utilization and poor user experience. Evolved Multimedia Broadcast Multicast Service (eMBMS) is a 3GPP standard [11] that provides an alternative and more efficient method for delivering live content to a large number of cellular network users. To improve the resource utilization, eMBMS allows sharing resources among a group of users watching the same content and transmitting the content to a group just once. Furthermore, to improve the channel condition of users, eMBMS allows Evolved-Node Base Stations (eNB) in a spatially local area to transmit a video synchronously at a common frequency and time (Figure 1.2), hence creating a Single Frequency Networks (SFN) and improving the Signal to Interference Noise Ratio (SINR) for users.

Cellular network providers continue to evolve their eMBMS deployment strategies. The ultimate goal is to maximize the Quality of Experience (QoE) for users while satisfying the resource constraints of the underlying network. To best utilize the scarce wireless resources, operators have to make various decisions when configuring the physical network for eMBMS. The key configuration decisions are: which eNBs to synchronize and form SFNs; how to share resources among users with disparate channel conditions and; how to handle the impact of eMBMS decisions on eNB's unicast-load, i.e. non-eMBMS users.

To make the best use of eMBMS, various models and algorithms have been proposed in literature to solve the network configuration problem [12, 14]. RTOP (Chapter 6) formulates a joint optimization model that considers the combined impact of various configuration parameters and improves the system utility, in comparison to state-of-the-art schemes. However, similar to the existing algorithms, RTOP is based on fairly-static network scenarios and does not consider the time-variance of the network state or mobile users. With the standardization of more attractive and dynamic services over eMBMS, such as MBMS Operation On-Demand (MOOD) [119], and extension of eMBMS into 5G networks [120], a solution is needed that is capable of reacting to network changes in real-time and aims at improving the video QoE over the duration of the whole stream regardless of user-mobility patterns.

This chapter proposes NIMBLE, a network optimization framework for eMBMS-based live user experience. NIMBLE is a resource management and allocation framework, that configures the physical network with the objective to maximize the end-user experience. The proposed solution is extremely scalable and dynamic making it feasible for deployment in real-world cellular networks. Specifically, the following contributions are made through NIMBLE:

- A QoE optimization model is formulated that solves the network configuration problem while considering both eMBMS as well as unicast users in the eMBMS service area. The model considers the three fundamental factors of QoE: video bitrates received by users, video frames dropped or skipped by users and, the switches in video bitrates encountered by users.
- A heuristics-based algorithm is proposed, that can solve the optimization model in real-time regardless of the number of users and their mobility pattern. NIMBLE also introduces parameters to control and stabilize the network-state. Frequent network reconfiguration can result in frequent bitrate switches for users and higher overhead cost of network management. Delaying reconfiguration can reduce the responsiveness to user's channel condition and deteriorate user experience. NIMBLE is designed to consider these trade-offs when re-configuring the network.
- Real-world scenarios are implemented with varying mobility patterns for users, using the simulation-based testbed proposed in Section 3.4. Traces of real videos are generated with different bitrates to analyze the behavior of NIMBLE and compare it with RTOP and state-of-the art approaches. Extensive evaluation is conducted to show that NIMBLE can improve the

Table 7.1: Notations For NIMBLE optimization model

| Symbol | Description |
|---|---|
| | **INPUTS** |
| $B$ | Set of one or more eNBs in the eMBMS service area |
| $C$ | Set of possible clusters of eNBs (non-empty subsets of $B$) |
| $L$ | Set of all possible SFN layouts i.e. ways to configure eNBs $B$ into non-overlapping clusters |
| $b_{lc}$ | Binary variable to inform if $b$ is in cluster $c$ for SFN layout $l$ |
| $N$ | Total number of resource blocks available at any eNB |
| $\alpha$ | Maximum fraction of resources allowed for eMBMS |
| $\beta_u$ | Priority weight for unicast user $u$ |
| $U$, $U_b$ | Set of unicast users ($M$) associate to eNB $b$ ($U_b$) |
| $Y_{ub}$ | Number of RBs assigned by eNB $b$ to unicast user $u$ |
| $V$ | Set of videos served by eMBMS in the service area |
| $R_v$ | Set of bitrates available for video $v$ |
| $M$, $M_v$ | Set of multicast users ($M$) subscribed to video $v$ ($M_v$) |
| $E_{xy}$ | Spectral efficiency achievable by user $x$ in cluster or eNB $y$ |
| $\gamma_v$ | Weight of bitrate-switching penalty for video $v$ |
| $r'_m$ | Current streaming bitrate of user $m$ |
| $w_m$,$G_m$ | Switching count and magnitude experienced by user $m$ so far |
| | **VARIABLES** |
| $L_{vl}$ | Binary variable to determine whether SFN layout $l$ has been chosen for video $v$ |
| $m_{vlcr}$ | Binary variable to determine whether a user $m$ watching video $v$ is placed in cluster $c \in l$ and assigned bitrate $r$ |
| $X_{vlcr}$ | Number of RBs allocated by eNBs in cluster $c \in l$ to video $v$ for bitrate $r$ |

end-user experience with 150% increase in bitrates and 75% reduction in bitrate switches.

## 7.2 Optimal user experience model

The goal is to formulate an optimization model that can maximize user QoE while fairly allocating resources to users of multiple videos in an eMBMS service area.

### 7.2.1 System model

A cellular network is considered, with a set $B$ of identically configured eNBs in the eMBMS service area. A set of videos $V$ encoded at bitrates $R_v$ is served

through eMBMS to users in set $M$. For each video $v$, eNBs can be synchronized to form one or more non-overlapping clusters of SFN (denoted by set $C$). Each eNB $b$ has $N$ number of RBs available that it distributes among its unicast users (in set $U_b$) and eMBMS users in its SFN clusters.

The UEs are assumed to report their Reference Signals Received Power (RSRP) based on cell-specific reference signals. eNBs calculate the achievable spectral efficiency for their associated unicast users i.e. $E_{ub}\ \forall u \in U_b$. For eMBMS users, the report is delivered to MCE that estimates user's Signal to Interference Noise Ratio (SINR) from each possible SFN cluster and calculates the achievable spectral efficiencies $E_{mc}\ \forall c \in C$.

Based on channel condition of users in an SFN cluster of a video, an MCE may choose to transmit one or more distinct bitrates, each at a different MCS. A standard eMBMS client [119] is considered that may select a bitrate, and consequently an MCS, which is best suitable for its channel condition. The spectral efficiency of the chosen MCS determines the number of RBs needed to achieve that bitrate. The maximum number of RBs an eNB can allocate to eMBMS however, is limited by a fraction $\alpha$ of total RBs ($\alpha N$) and is usually set to 60% [116].

In such a system, different video-specific SFN layouts (denoted by set $L$) are possible, leading to different bitrates achievable for eMBMS users. A complete network configuration is defined as an SFN layout $l$ chosen for each video and bitrates transmitted by SFN clusters in $l$. The goal is to choose a network configuration that maximizes the QoE for all the users.

### 7.2.2 Problem formulation

When scheduling resources for users, cellular operators usually implement a fairness utility, such as Proportional Fairness (PF) [61]. The proposed optimization model is agnostic to the metric used and aims at maximizing any operator-defined system utility. However, to simplify the formulation, without loss of generality, PF is considered as the fairness metric, which is defined as sum-log of user bitrates. The optimization problem is a quadratic program with Integer constraints and consists of multiple parts.

#### 7.2.2.1 Bitrate utility of eMBMS users

Being one of the key QoE factor, user bitrates are included in the objective function. A binary variable $m_{vlcr}$ is defined to decide whether a user $m$ interested in video $v$ should be placed in cluster $c$ of an SFN layout $l$, and assigned a bitrate $r$. The sum bitrate-utility of all the eMBMS users $M_v$ of video $v$ can be calculated as:

$$Q(M_v) = \sum_{r \in R_v} \log(r) \sum_{m \in M_v} m_{vlcr} \tag{7.1}$$

#### 7.2.2.2 Switching utility of eMBMS users

Switches in video bitrates is another factor that impacts the video-watching experience for users. For a video $v$, the total switching-penalty of all the users $M_v$ can be calculated as:

$$S(M_v) = \sum_{m \in M_v} \sum_{r \in R_v} f(m, r) \cdot m_{vlcr}, \tag{7.2}$$

where $f(m, r)$ is a function that measures the impact of switching a user $m$ from its current bitrate to bitrate $r$.

The impact of quality switching for a user can be captured using a log-based utility as expressed below:

$$f(m, r) = \log(r) - \log(r'_m), \tag{7.3}$$

where $r'_m$ is the current streaming bitrate of user $m$. The logarithmic function reflects the reduced impact of visual experience when switching between higher bitrates.

Equation 7.3 considers switching to a higher bitrate ($r > r'_m$) as a reward i.e. improved QoE and switching to a lower bitrate ($r < r'_m$) as a penalty. Some QoE metrics penalize switching to a higher bitrate as well [68]. For such metrics, a modulus can be applied in Equation 7.3.

The importance and weight of the switching factor varies across QoE metrics [80]. Furthermore, while some QoE metrics consider the total number of switches in a session, others look at the switching magnitude [68] to capture abrupt quality variations. An operator can choose a metric that best suits the demand of their

users and $f(m, r)$ can be defined accordingly.

#### 7.2.2.3 Frames lost by eMBMS users

Generally, users in wireless networks experience losses if their SINR is not high enough to properly decode the MCS assigned to them. Depending on how live video is streamed, these losses can result in stalls, freezes or dropped and skipped frames. Majority of the eNB metrics consider such an experience to be the most annoying factor for users [121] and assign it the highest (negative) weight in QoE. To eliminate frame losses, a constraint is added to the optimization model which ensures that the bitrate assigned to a user is not transmitted at an MCS higher than that of the user. Furthermore, the constraint ensures a stall-free transmission, by allocating sufficient resources to each transmitted bitrate.

#### 7.2.2.4 Experience of unicast users

In general, especially when cellular operators do not control or provide the content for unicast users, they cannot quantify actual user-experience and instead maximize user-throughput. Unicast users are associated to the eNB which yields the strongest SINR and the highest spectral efficiency. Each eNB schedules its own unicast users and assigns them RBs. The throughput achieved by a user of eNB $b$, is the product of the user's spectral efficiency, $E_{ub}$, and the number of RBs assigned to it, $Y_{ub}$. Assuming PF-fairness and a service-differentiation or a priority coefficient $\beta_u$ for user $u$, the total throughput factor for all the unicast users $U$ in the eMBMS service area can be calculated as:

$$T(U) = \sum_{b \in B} \sum_{u \in U_b} \beta_u . \log(E_{ub} \cdot Y_{ub}) \tag{7.4}$$

#### 7.2.2.5 Optimization model

The objective of the optimization model is to maximize the utility for the unicast users and the eMBMS users of all the videos in each SFN cluster of the chosen SFN layout. This is illustrated in Problem 1 where, $L_{vl}$ is a binary variable to determine whether SFN layout $l$ has been chosen for video $v$; $m_{vlcr}$ is a binary variable to determine whether user $m$ will be associated to cluster $c$ and receive

bitrate $r$; $X_{vlcr}$ is the number of RBs allocated to the bitrate $r$ of video $v$ in cluster $c \in l$ and; $\gamma_v$ is the weight assigned to the switching penalty for video $v$.

**Problem 1: Maximizing unicast and eMBMS utility**

$$max\ T(U) + \sum_{v \in V} \sum_{l \in L} L_{vl} \cdot \sum_{c \in l} \Big( Q(M_v) - \gamma_v \cdot S(M_v) \Big) \tag{7.5a}$$

*subject to*

$$\sum_{l \in L} L_{vl} = 1,\ \forall v \in V \tag{7.5b}$$

$$\sum_{c \in l} \sum_{r \in R_v} m_{vlcr} \leq 1, \forall v \in V, l \in L \tag{7.5c}$$

$$\sum_{l \in L} \sum_{c \in l} \sum_{r \in R_v} m_{vlcr} = M_v,\ \forall v \in V \tag{7.5d}$$

$$X_{vlcr} \cdot E_{mc} \geq r \cdot m_{vlcr}, \forall m \in M, v \in V, r \in R_v, c \in l \in L \tag{7.5e}$$

$$\sum_{u \in U_b} Y_{ub} + \sum_{v \in V} \sum_{l \in L} \sum_{c \in l} \sum_{r \in R_v} b_{lc} \cdot X_{vlcr} \leq N, \forall b \in B \tag{7.5f}$$

$$\sum_{v \in V} \sum_{l \in L} \sum_{c \in l} \sum_{r \in R_v} b_{lc} \cdot X_{vlcr} \leq \alpha N, \forall b \in B \tag{7.5g}$$

Constraint 7.5b ensures that one SFN layout is chosen for each video. Constraint 7.5c guarantees that a user is not assigned more than one bitrates or placed in more than one clusters. Constraint 7.5d enforces that each user is assigned a bitrate and Constraint 7.5e ensures that the user can decode its assigned bitrate by allocating enough RBs based on the user's spectral efficiency. Constraint 7.5f says that the sum of RBs allocated to all the unicast and eMBMS sessions at an eNB must be less than total RBs available and Constraint 7.5g limits the percentage of RBs for eMBMS to $\alpha$ which is usually set to 60% [116].

### 7.2.3 Practical considerations

In addition to the aforementioned constraints, there are a few practical limitations that may hinder the deployment of *Problem 1* in real systems.

#### 7.2.3.1 Unicast resource allocation

In practice, an Multicast Coordination Entity (MCE) does not control the number of RBs ($Y_{ub}$), that each eNB may allocate to its unicast users. Hence, the term $T(U)$ in Equation 7.5a and $\sum_{u \in U_b} Y_{ub}$ in Constraint 7.5f cannot be solved

by any optimization algorithm running on an MCE. To still account for the impact of MCE decisions on unicast users, RTOP (Chapter 6) defines a variable $Y_b$ that represents the total RBs assigned to unicast users by eNB $b$. An operator can calculate $Y_b$ based on the priority of eMBMS over unicast [14] or through a multicast weight function that modulates the resource allocation between unicast and eMBMS users [12] e.g. by considering the number of users of each type. By taking $Y_b$ as an input for Constraint 7.5f and eliminating $T(U)$ from the objective, an optimization model is obtained that can theoretically be implemented on an MCE.

#### 7.2.3.2 Limited user feedback

The factor for bitrate-switching penalty, $S(M_v)$, in the objective (Equation 7.5a) assumes accurate knowledge of user's current state e.g. bitrate ($r'_m$ in Equation 7.3). However, in practice, eMBMS does not have a feedback loop between user's video-client and MCE, limiting the MCE's knowledge of current user-state. Establishing such a loop will incur high overhead cost and is usually avoided for broadcast/multicast based systems such as eMBMS. Therefore, a mechanism is needed to calculate the switching penalty without user's application-level feedback. Such an approach may not yield optimal results but is essential to enable real-world deployment of the optimization model. The proposed algorithm in Section 7.3.1.6, utilizes UE's reported channel condition to estimate the user-state and calculate the switching penalty.

#### 7.2.3.3 Time scale of optimization

In a cellular network, the network or user-state may vary or users may join or leave video streams, rendering a pre-computed network solution sub-optimal. To ensure a granular responsiveness to the dynamics, an algorithm must be able to solve the network reconfiguration problem in real-time, regardless of the number of users or state variations. The optimization model in *Problem 1* does not scale well with the number of users and hence a lightweight scalable algorithm is needed that can find optimal or near-optimal network configurations in real-time.

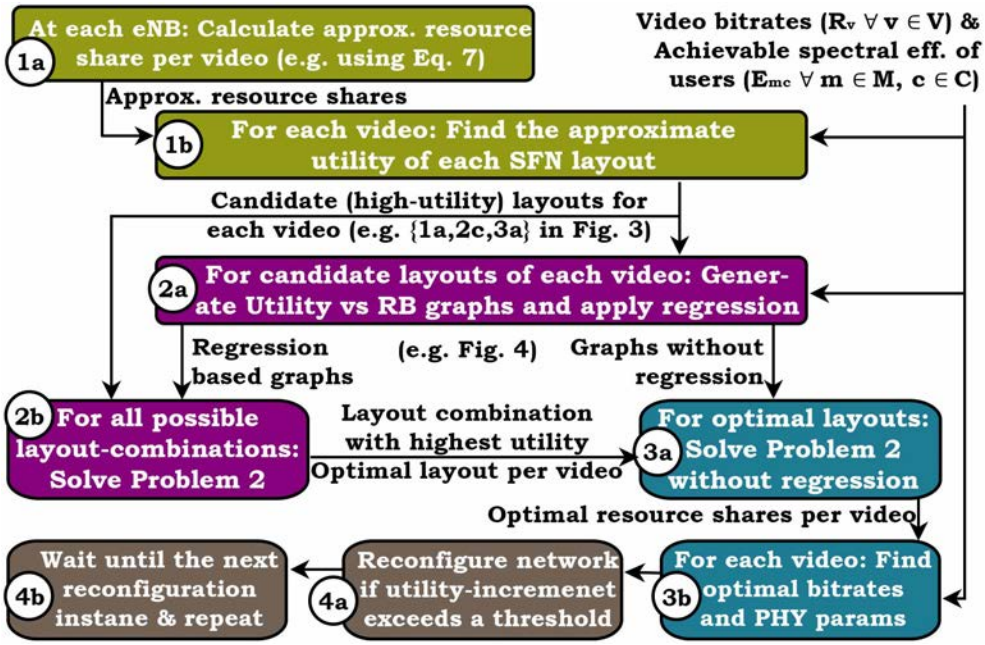



Figure 7.1: A flow-chart of heuristics used by NIMBLE

## 7.3 Lightweight heuristics-based algorithm

A network configuration is defined as an SFN layout chosen for each video, the bitrates served in each cluster of that layout and, the physical layer parameters, i.e. the MCS and number of RBs each bitrate is transmitted on. An algorithm for network optimization of eMBMS-based live video experience (NIMBLE) is developed. The approach used to build a dynamic network reconfiguration algorithm is to consider multi-stage heuristic-based techniques, inspired by RTOP design (Chapter 6), while introducing parameters to address different design challenges that arise due to the variation in network or user state over time.

The problem of selecting an optimal SFN layout for each video depends on the number of RBs allocated to each video and the consequent user grouping. This makes the problem combinatorial in nature and can result in exponentially-increasing outcomes. To reduce the computation time, the overall process is divided into four stages (Figure 7.1). The first stage calculates an approximate share of resources for each video (1a) and narrows down the choices of SFN layouts for videos to a set of candidate layouts (1b) with high utilities. The second stage: generates utility vs resource graphs for the candidate layouts (2a); uses these graphs to solve a simplified optimization problem for different possible layout-combinations (2b) and; picks the combination with the highest utility. The output of this stage consists of an SFN layout chosen for each video.

The third stage first calculates the optimal share of RBs for all the videos in each SFN cluster of its chosen layout (3a) by solving an optimization problem. Then, for the given RBs, it calculates the optimal bitrates to transmit for each video in each cluster and the MCS and RB share for the bitrates (3b). Finally, stage 4 reconfigures the network if the utility of the chosen configuration exceeds the utility of the current configuration by a pre-defined factor (4a). NIMBLE then waits until the next reconfiguration instance and repeats the process if there are any active eMBMS sessions (4b). The rest of the section, describes these stages in further details.

### 7.3.1 Candidate SFN layouts and their utility

This section presents the techniques introduced in NIMBLE to find a set of candidate SFN layouts for a video $v$ and approximate the utility that can be achieved by users of $v$, given a particular SFN layout. If a layout has more than one cluster, a user is assumed to associate with the cluster that provides the highest SINR value.

#### 7.3.1.1 Approximate resource share

When estimating the utility of a video, due to no prior knowledge of the optimal RB-allocation, a proportional resource distribution, denoted by $P_{vc}$, is considered among videos in a cluster $c$, to get an approximate share of RBs for a video $v$, as shown below (See Table 7.1 for notations):

$$P_{vc} = min(\alpha N, N - max(Y_b \forall b \in c)) \cdot \frac{|M_v|}{\sum_{v \in V} |M_v|} \tag{7.6}$$

For example, in a cluster where the eNB with the highest unicast-load ($Y_b$) has 30 RBs available for eMBMS and has to serve three eMBMS videos with 50, 100 and 150 users respectively, it will assume $30 * \frac{50}{50+100+150} = 5$ RBs allocated to the first video. Note that, to improve the accuracy of utility estimation, the distribution should be chosen according to the fairness-metric of the system (which in this case is PF).

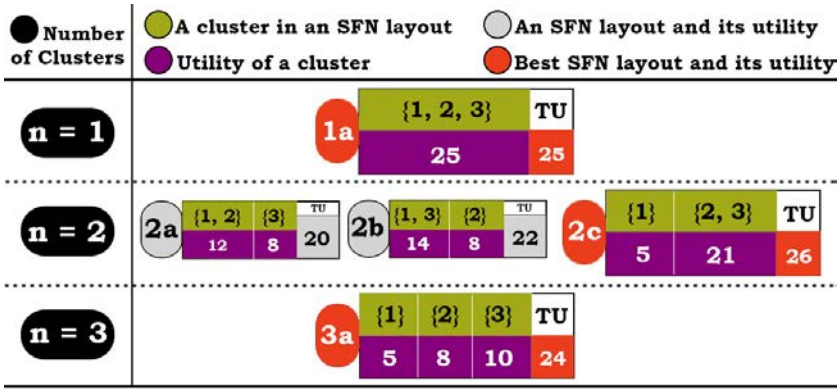



Figure 7.2: Example of SFN layouts classified by number of clusters in them. *In each class the layout with the highest utility is chosen as a candidate.*

#### 7.3.1.2 Multiple candidates per video

Although the aforementioned resource distribution is fair, it may not be optimal and hence the measured utility may also be sub-optimal. To increase the probability of retaining the best SFN layout for a video when applying this heuristic, multiple layouts are chosen as candidates for each video. This is achieved by classifying the layouts based on the number of clusters in them and, for each class picking a candidate layout as the one with the highest utility in a class (Figure 7.2). The utility of a layout $l_v$ is calculated by finding the best user-groups for each $c \in l_v$, assuming $P_{vc}$ as the available RBs, as explained below.

#### 7.3.1.3 Number of bitrates in a cluster

Depending on $P_{vc}$, users associated with the SFN cluster $c$ can be split into groups with each group receiving a different bitrate of the video. Serving too many bitrates reduces the share of RBs per user-group, and may reduce the achievable utility. Figure 6.3 presents results of extensive experiments and shows that 90% of the time, the optimal solution is to create no more than two groups. So for the NIMBLE heuristics algorithm, the maximum number of bitrates per video served in an SFN cluster are limited to two, with each group receiving the video at a different bitrate.

#### 7.3.1.4 Spectral efficiency of a user group

The MCS assigned to a group of users is restricted to the user with the worst channel condition. This avoids decoding errors or frame losses for any user, but

an inefficient user placement may unnecessarily limit the spectral efficiency of a group and hence the achievable bitrates. Therefore, NIMBLE calculates the utility that different user grouping combinations can achieve.

#### 7.3.1.5 Choosing the best user-grouping

: Assigning a bitrate to a user-group and placing a user in that group determines its two utility components, bitrate-factor and switching-penalty. For users $M_v$ of video $v$, served at $R_v$ bitrates, exhaustively testing all possible combinations of user-groups and bitrates takes $\mathcal{O}(|R_v| \cdot |M_v|^{|M_v|})$ to solve. This approach cannot be solved in real-time, especially for large number of users.

Instead, to calculate the bitrate-factor of utility, the user-grouping algorithm proposed in Section 6.3.1.2 is utilized, which pre-groups users based on their CQI values and finds the highest bitrate that can be assigned to each CQI group with the available RBs. Although this algorithm does not measure the switching-penalty that each user may incur, it runs in $O(|R_v|^2)$ and finds the maximum utility of a video in an SFN cluster for a given number of RBs. The user grouping algorithm is executed for all $c \in l_v$ to find the bitrates ($r_m$) that each $m \in M_v$ will receive.

#### 7.3.1.6 Utility with switching penalties

With a bitrate chosen for each user, Equation 7.3 can be used to calculate the switching penalty for each user. However, as explained in Section 7.2.3, due to the lack of a feedback loop, the algorithm does not know the current streaming bitrates ($r'_m$) of users, which is an input to Equation 7.3. To estimate $r'_m$ for each user, NIMBLE utilizes the physical-layer channel condition reported by users. Since MCE is aware of how the network is currently configured and what bitrates are being served, it can predict the highest user-bitrate based on user's channel condition and transmission MCS of each bitrate. NIMBLE assumes that each user is streaming the highest bitrate that they can receive and uses it to calculate the switching penalty.

The total penalty incurred on the utility of a layout is the sum of each user's penalty. Adding the switching penalty to the bitrate-factor gives an approximate utility of an SFN layout $l$. This process is repeated for each video for all $l \in L$ to calculate the achievable utilities. The layouts with highest utility in each class

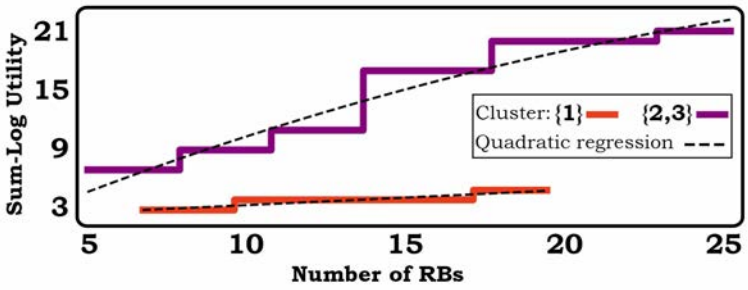



Figure 7.3: A sample utility vs RB graph with quadratic regression. *For any cluster, the lower bound of RBs is the minimum RBs needed to satisfy all users of a video and the upper bound is the RBs available for eMBMS in that cluster.*

(Figure 7.2) are chosen to form a set of video-specific SFN layouts.

### 7.3.2 Optimal SFN layout for each video

When estimating utilities of candidate layouts in Section 7.3.1, an approximate resource distribution ($P_{vc}$) was assumed among videos, which may yield sub-optimal or infeasible results. Therefore when choosing optimal layouts for videos, the candidate layouts are evaluated over a range of RB values. Furthermore, choosing a particular layout for a video can vary its users' cluster (and eNB) associations and consequently vary the resource share for other videos. Hence, different possible combinations of candidate layouts are explored, to pick a combination that maximizes the overall system utility while respecting network and user constraints (Section 7.2).

#### 7.3.2.1 Utility vs RB graphs

For each candidate layout $l_v$ of a video $v$, the user grouping algorithm is executed for all possible RB allocations and bitrate-based utility vs RB graphs are obtained. Due to the piece-wise relationship between video bitrates and throughput, the utility vs RB graphs are stair functions (as shown in Figure 7.3). If $X_{vc}$ represents the RBs allocated to $v$ in each cluster $c \in l_v$, then for a given $X_{vc}$, the utility of $l_v$ can be calculated as follows:

$$h(X_{vc} \forall c \in l_v) = \sum_{c \in l_v} Graph(X_{vc}) \tag{7.7}$$

As the function has to be evaluated for multiple possible RB values, SFN layouts

and videos, a quadratic regression is applied to speed-up the evaluation and obtain a continuous layout-utility function:

$$h(X_{vc}\forall c\in l_v) = \sum_{c\in l_v} i_{vc}\cdot X_{vc}^2 + j_{vc}\cdot X_{vc} + k_{vc}, \tag{7.8}$$

where $i_{vc}$, $j_{vc}$ and $k_{vc}$ are the quadratic regression coefficients.

#### 7.3.2.2 Combination of SFN layouts

In this step, the optimal SFN layout $l_v$ for each video is selected by maximizing the sum of regression-based layout-utility function (Equation 7.8) for different combinations of previously identified candidates while respecting the resource and QoE constraints.

**Problem 2: Maximizing utility for a combination of SFN layouts**

$$max \sum_{v\in V} h(X_{vc}\forall c\in l_v) \tag{7.9a}$$

*subject to*

$$\sum_{v\in V}\sum_{c\in l_v} b_{lc}\cdot X_{vc} \leq \min\left(\alpha N, N - Y_b\right), \forall\, b\in B \tag{7.9b}$$

$$X_{vc} \geq D_{vc}, \forall\, v\in V, c\in l_v, \tag{7.9c}$$

where $D_{vc}$ is the lower bound for $X_{vc}$.

Constraint 7.9b is similar to Constraint 7.5f and 7.5g and limits the RBs available to eMBMS users in a cluster. Constraint 7.9c ensures that each video gets enough RBs to at least attain the lowest bitrate for all users and avoid stalls.

### 7.3.3 Optimal network configuration

Solving *Problem 2* for various possible combinations of candidate layouts, gives the combination with the highest achievable utility and optimal $l_v$ $\forall v \in V$. To acquire the complete network configuration, the optimal solution for bitrates to transmit in each cluster and the physical layer parameters are required. With an SFN layout chosen for each video (Section 7.3.2), the solution is backtracked to find the optimal resource allocation and the user grouping. *Problem 2* is solved one more time for the chosen combination of SFN layouts, but instead of using the quadratic regression, the actual step function (Equation 7.7) is used. This

provides the optimal $X_{vc}$ for each video and cluster. These details are passed as an input to user grouping algorithm (Algorithm 1) to determine the bitrates to transmit for each video and the associated MCS values and number of RBs.

### 7.3.4 Controlling network reconfiguration

Based on the results of experiments conducted in Section 7.4 over a dual-core i7 processor laptop with 16GB RAM; regardless of the number of eMBMS users per video, NIMBLE was able to solve the network configuration problem in less than 500ms. This computation time is fast enough to implement NIMBLE in a dynamic and mobile network. However, there are a few additional network-based parameters to consider for a real-world deployment of NIMBLE.

NIMBLE runs on an MCE and if the solution involves changing the SFN layout or bitrates to serve for any video, then MCE updates the Multicast Control Channel (MCCH) and Multicast Transport Channel (MTCH). The information carried by MCCH includes MCS and sub-frame allocation. Changes in scheduling information are sent to all the involved eNBs and eNBs advertise MCCH to UEs [11]. Frequent reconfigurations increase the overhead cost associated with these actions and message exchanges. Frequent reconfigurations may also increase the physical-layer Transport Block (TB)-losses. When the current network configuration and the configuration chosen by NIMBLE are different, two design parameters are introduced by NIMBLE to make the network configuration more practical and reliable.

#### 7.3.4.1 Utility increment threshold

Based on the amount of variability in the cellular system, it is possible that the new network configuration is only slightly better than the current one. In other words, the improvement in the system-utility may not be significant. A threshold ratio $\delta$ is defined and the network is only reconfigured when the increase in utility exceeds the threshold. Note that, depending on user mobility patterns or unicast-load on eNBs, the current network configuration may become infeasible i.e. cannot deliver stall-free video to all the users anymore. In such cases, the QoE objectives are not compromised and the network is configured, regardless of the amount of gain in utility.

#### 7.3.4.2 Reconfiguration trigger

In fairly-static network scenarios, an event-based approach can be used to decide when to re-run NIMBLE. The number of events can be counted, such as users leaving or joining eMBMS session, beginning or end of an eMBMS session, users' channel condition changed beyond certain limits etc. A NIMBLE re-run can be triggered when an operator-defined threshold is met.

For more dynamic network scenarios, where the number of variations is high, a better approach is to run NIMBLE periodically at regular intervals, denoted by $i$. Setting $i$ too low can result in frequent network reconfigurations and setting it too high can reduce the responsiveness to the changes in network or user state. An operator can set $i$ to reflect the characteristics of their network.

### 7.3.5 Complete NIMBLE algorithm

Algorithm 3 presents the complete algorithm of NIMBLE based on the aforementioned techniques and involves four main steps. First the layout-utility function is used (Lines 16-23) to find approximate utility achievable by different SFN layouts for each video (Lines 2-4). The layouts are classified based on the number of clusters in them i.e. size of the layout and from each class, the layout with the highest utility is chosen as a candidate (Lines 5-6). Then all possible combinations of layouts for different videos are explored (Line 7) and the combination with the highest sum-utility is chosen (Lines 8-12) by solving *Problem 2* with regression (Equation 7.8).

Then, Problem 2 is solved (Line 14) without regression (Equation 7.7) to find the optimal resource shares in each cluster of the chosen video layouts and user grouping algorithm is executed to find the optimal user groups, bitrates and physical-layer parameters in each cluster. Finally, if the utility of the best combination is higher than the current configuration's utility by a factor of $\delta$ or if the current configuration is not feasible anymore (Line 13), the combination is chosen as the new solution and the network is reconfigured accordingly. NIMBLE then waits for a duration $i$, until it is time to run the algorithm again (Line 15), and repeats the process until there are any active eMBMS sessions (Line 1).

**Algorithm 3** Complete NIMBLE algorithm

**Input:** $M_{vlc}$: Users of $v$ in cluster $c \in l$; $C_N$: Current network config. and $U_N$: its utility. See Table 7.1 for other inputs.

**Output:** SFN cluster and user groups for each video

**while** *Active_eMBMS_Sessions* **do**
  **for** $v \in V$ **do**
    **for** $l \in L$ **do**
      $utility, Graph[v,l] \leftarrow$ **LAYOUT_U**$(v,l)$
      **if** $utility > maxU[v,|l|]$ **then**
        $maxU[v,l] \leftarrow utility; Candidates[v,|l|] \leftarrow l$
  $Cartesian \leftarrow \{(l_1..l_v) \mid l_v \in Candidates[v]$ **for** $v \in V\}$
  $U_{max} \leftarrow 0$
  **for** $combo \in Cartesian$ **do** ▷ *Layout combinations*
    *Get Utility U by solving Problem 2*
    **if** $U > U_{max}$ **then**
      $U_{max} \leftarrow U; Optimal_Combo \leftarrow combo$
  **if** $C_N \notin Cartesian \vee U_N < \delta \cdot U_{max}$ **then**
    *Solve Prob. 2 for Optimal_Combo without regression to get optimal RB shares and user groups for each video*
  $sleep(i)$ ▷ *until next interval*

**LAYOUT_U**$(v,l)$ ▷ *Utility of an SFN layout*
$PF_utility \leftarrow 0; Graph \leftarrow []$
**for** $c \in l$ **do**
  $maxRBs \leftarrow min(\alpha N,$ *N-*$Y_b$ **for** $b \in c)$ ▷ *Available RBs*
  **for** $RBs \leftarrow 1$ *to* $maxRBs$ **do**
    $Graph[RBs] \leftarrow$ **User_Grouping**$(M_{vlc}, RBs)$
  $PF_RBs \leftarrow \frac{|M_v|}{sum(M_w \forall w \in V)} * maxRBs$
  $PF_utility += Graph[PF_RBs]$
**return** $PF_utility, Graph$

## 7.4 Performance evaluation and comparison

NIMBLE is implemented on the simulation-based testbed proposed in Section 3.4 with real-video traces for demonstration and evaluation. Commonly used network parameters [116] are applied to the simulation setup as listed in Table 7.2. In this section, first the details of the simulation setup are presented. Then the performance of NIMBLE is evaluated in real-world scenarios and the impact of control parameters is analyzed. Finally, NIMBLE is compared with state-of-the art approaches and RTOP.

Table 7.2: NIMBLE simulation parameters

| Parameter | Value |
|---|---|
| Cellular Layout | Hexagonal grid with 4 eNBs |
| eMBMS Service Area | 1250 m x 875 m |
| eNB Tx Power | 10 Watts |
| Frequency, bandwidth, RBs | 2.1 GHz, 20 MHz and 100 |
| Path Loss Model | Log-Normal Shadowing |
| Path Loss Exponent | 4.5 (Indoor Obstructed) |
| Noise Density & UE Noise | -174 dBm/Hz & 7 dB |
| Channel Model | Multi-path Fading AWGN [99] |
| Handover Hysteresis Margin | $1dB$ of average-SINR (over 250ms) |
| Spectral Efficiencies (bits/RB) from CQIs 1 to 15 | [20, 31, 50, 79, 116, 155, 195, 253, 318, 360, 439, 515, 597, 675, 733] |
| User Mobility Model | Random-Way Point [97] |
| User Movement Speed | Walking: 1∼1.5 m/s-Stationary:0m/s |
| Pause/Waiting Times | Shopping: 0∼300s-Browsing: 0∼30s |

### 7.4.1 Simulation setup

An eMBMS service area is considered covering a large shopping mall (Figure 3.7) with four eNBs arranged in a hexagonal grid. Three live DASH-based [21] eMBMS videos are served in the mall and five minutes of the transmission is analyzed. For each video, H.264 encoding is used with 24 frames/sec (fps), 8 frames per Group of Picture (GOP), to generate trace files at low (400kbps), medium (1.5Mbps) and high (4Mbps) video quality. A 10% Forward Error Correction (FEC) is also assumed to be encoded in each frame, and a frame is considered lost if more than 10% of its transmitted bytes are lost. A total of 1200 eMBMS users are generated (400 per video) in the shopping mall.

The mall consists of an entertainment zone with restaurants, theater, resting area, food court etc. Around 30% of the eMBMS users are located in the entertainment zone and the closest eNB is considered to be congested. The rest of the eMBMS users are distributed uniformly over the shopping mall. There are three types of users in the mall:

- **Resting**: These are static users and during the duration of the simulation are assumed to not change their location. Most of the resting users (around 80%) are located in the entertainment zone.
- **Shopping**: Random Way Point (RWP) model [97] is used to determine the mobility pattern for users that are shopping. These users may spend a long

time at a shop, so a high pause time (Table 7.2) is set for this user category.

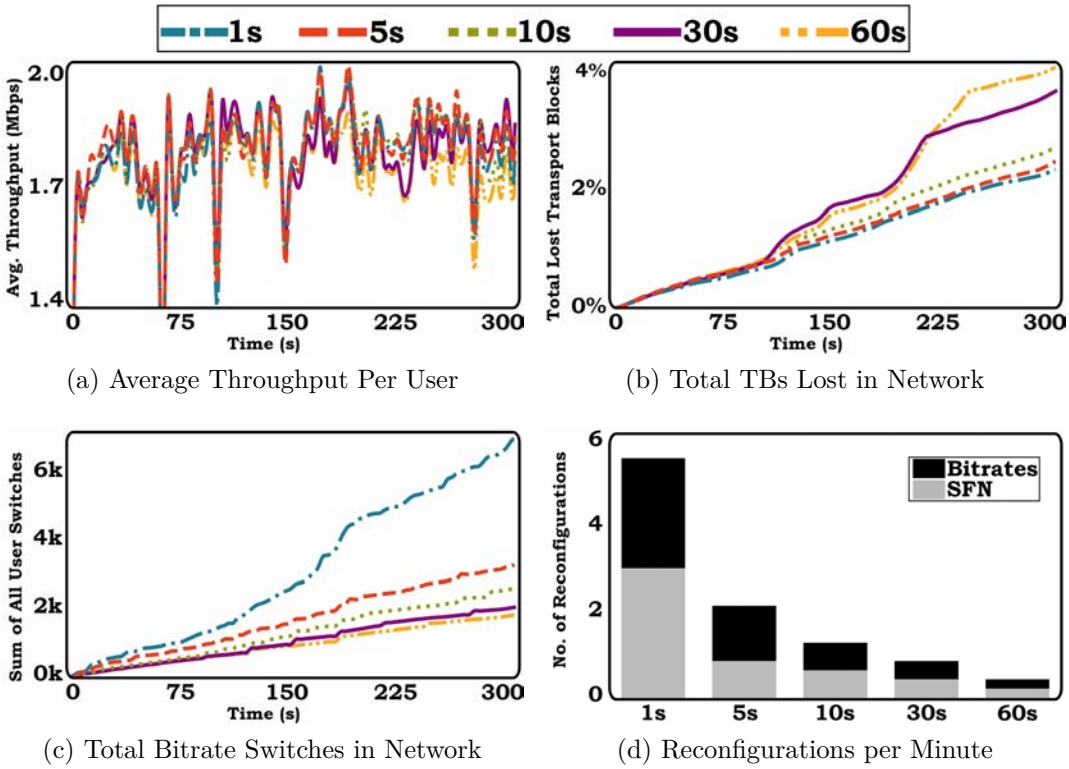
- **Browsing**: These are the type of users that do not spend too much time at one shop. RWP model is used to determine the mobility pattern and a low range is set for pause time (Table 7.2) to mimic the behavior of browsing users. These users exhibit the highest mobility pattern in the simulation scenario.

A total of 900 mobility traces are generated, with equal probability of each user-type, and are assigned to the 1200 eMBMS users. A mobility trace is defined by the initial location of a user and the mobility pattern for the duration of the simulation. Using 900 traces for 1200 users simulates users that may follow exactly the same mobility pattern, e.g. two or more people visiting together.

**Mobility and application management**: On the physical layer, the UE averages its SINR over a duration of 250ms for each SFN cluster to choose the best cluster and applies handover-hysteresis to avoid ping-pong effect. The UE also reports its achievable MCS from each cluster based on AWGN BLER vs SINR curves [99]. The target BLER is set to 1% which is the standard in eMBMS [11]. On the application layer, a greedy video client is assumed that subscribes to the highest available bitrate decodable with the user SINR. When switching between bitrates, the client waits until an end of a GOP to switch, hence providing a smooth switching experience for the end-user.

The key performance evaluation metrics are

- Throughput: Average size of decodable frames received by users per second.
- Transport blocks (TB) and frames lost in the network.
- Total number of bitrate switches encountered by users over the duration of the simulation.
- Stall Duration: A stall is defined as more than one consecutive frames not played by a user, either due to network loss or a missing dependency.
- QoE: A regression-based QoE model [68] is used and is defined as: $-56.6P_r + 0.007\overline{B} + 0.0007\overline{B_s} + 54.0$, where $P_r$ is the re-buffering percentage, $\bar{B}$ is the average bitrate and $\bar{B}_s$ is the average bitrate switch magnitude.
- Fairness:F-index [83] is used for measuring fairness and is defined as: $1 - \frac{2\sigma}{H-L}$, where $\sigma$ is the standard deviation of user QoE scores, H is the upper QoE bound (100) and L is the lower QoE bound (0).




(a) Average Throughput Per User

(b) Total TBs Lost in Network

(c) Total Bitrate Switches in Network

(d) Reconfigurations per Minute

Figure 7.4: Analyzing recalculation interval $i$ at $\delta = 2\%$: *Delaying reconfiguration may reduce switches but results in higher losses.*

- Network Reconfiguration: Changes made to SFN layout or streaming bitrates of a video per minute.

### 7.4.2 Analysis of control parameters

The widely-used Proportional Fairness (PF) metric [61] is adapted as the system utility for NIMBLE and the impact of the two control parameters (Section 7.3.4) is analyzed. The parameter values are varied and the impact is examined on average user throughput, total transport blocks lost in the network, total bitrate switches experienced by all the users and the network reconfiguration frequency. For each scenario, the experiment is repeated 5 times by varying user mobility patterns and the average results are presented.

#### 7.4.2.1 Tuning periodic reconfiguration interval ($i$)

For this set of experiments the increment threshold ($\delta$) is fixed to 2% and the solution recalculation interval is varied between 1s and 60s. Delaying network

reconfiguration (e.g. $i = 60s$) resulted in lesser reconfiguration overhead (Figure 7.4d) but also reduced NIMBLE's responsiveness to changes in network or user condition and incurred high ($\approx 4\%$) network losses (Figure 7.4b).

On the other hand, configuring network too often (every 1s) increased the switches in bitrates (Figure 7.4c) and overhead costs but reduced losses by responding quickly to the system dynamics. Due to user mobility and $\delta$ being same for these experiments, the throughput trend was similar for different recalculation intervals. However, due to lesser losses in network and quicker response to changes in user channel condition, approaches that ran more often achieved higher throughput (Figure 7.4a). In comparison to $i = 60s$, $i = 5s$ increased the average system throughput by 4%.

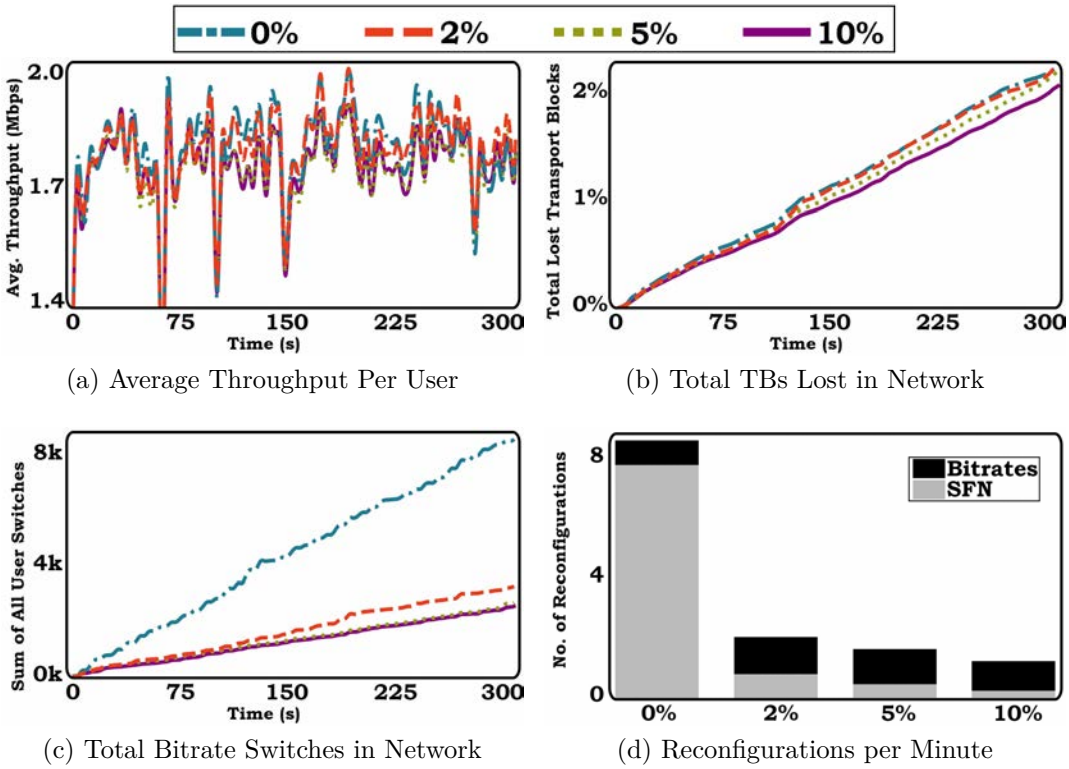
These experiments demonstrate the importance of selecting an appropriate reconfiguration interval and show that prolonging reconfiguration can reduce the responsiveness to user's channel condition and deteriorate user experience by increasing the losses or reducing the throughput, whereas reconfiguring too often can negatively impact user experience by increasing the number of bitrate switches. For the following experiments $i = 5s$ is chosen as the reconfiguration interval which, in the conducted simulation scenario, keeps bitrate switches and reconfiguration overhead to a reasonable value while maintaining responsiveness to network-state and user channel condition.

#### 7.4.2.2 Tuning utility increment threshold ($\delta$)

For this set of experiments the recalculation interval $i$ for NIMBLE is fixed to 5 seconds and $\delta$ is varied between 0% and 10%. Higher $\delta$ values decrease the probability of reconfiguring network based on bitrate gains in an effort to reduce bitrate switches. This trend can be seen in Figure 7.5 where $\delta = 0\%$ achieved the highest throughput (Figure 7.5a) but also the highest number of switches (Figure 7.5c).

Regardless of the value of $\delta$, NIMBLE reconfigures the network if the current configuration has become infeasible i.e. can cause losses for any user. Therefore, even though higher $\delta$ values reconfigured network less often (Figure 7.5d), they did not incur higher losses (Figure 7.5b) and the difference in percentage of lost TBs was insignificant across different values of $\delta$.

As NIMBLE ran every 5 seconds and found a solution with BLER$\leq 1\%$ for each user, the total percentage of lost TBs was around 2%. The additional 1% is




(a) Average Throughput Per User

(b) Total TBs Lost in Network

(c) Total Bitrate Switches in Network

(d) Reconfigurations per Minute

Figure 7.5: Analyzing increment threshold $\delta$ at $i = 5s$: *Lower values can achieve higher throughput but incurs higher switches and reconfiguration overhead.*

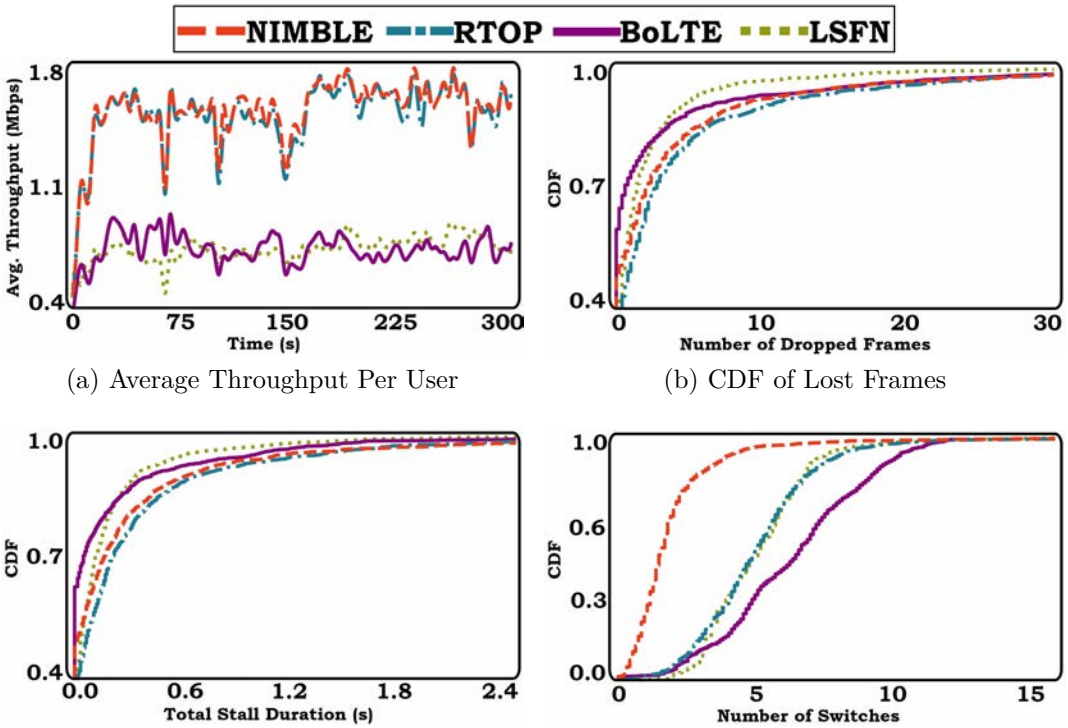
accounted by the possible degradation in channel condition of some users during the 5-second interval due to the path loss or mobility pattern. Based on these experiments and analysis, $\delta = 2\%$ is selected for NIMBLE in the following experiments, which avoids network reconfiguration on slight utility-increments, while maintaining responsiveness to user experience through a 3% switches-reduction in comparison to $\delta = 0\%$ and a 2.5% throughput-increase in comparison to $\delta = 5\%$ and $\delta = 10\%$.

### 7.4.3 Comparison with state-of-the-art algorithms

Based on PF-utility, the performance of following approaches is compared.

**BoLTE** [14]: Creates SFN clusters to maximize PF-utility but does not consider grouping users based on their channel condition. The proposed algorithm assumes single bitrate per video. For a fair comparison, the utility of an SFN layout is calculated by assigning the best achievable bitrate for each video by the heuristics proposed in BoLTE.

**One Large SFN (LSFN)**: In [12], Chen et al. propose partitioning users into




(a) Average Throughput Per User

(b) CDF of Lost Frames

(c) CDF of Total Stall Duration

(d) CDF of Switches to Lower Bitrates

Figure 7.6: Results of comparing algorithms: *NIMBLE serves users with higher throughput and lesser switches.*

groups based on their channel condition, but assume that all eNBs in the service area are always synchronized, hence creating one large SFN cluster. The proposed scheme only solves the problem for one eMBMS video. For evaluation with real-world scenarios, VG [12] is extended based on a subset of the proposed optimization model to work with multiple videos and the approach is named LSFN.

**RTOP** (Chapter 6): This algorithm solves a joint optimization problem for creating SFN clusters as well as user groups to maximize bitrate-based PF utility. However, RTOP does not consider the variation in network or user state over time and the impact of bitrate-switches on user QoE.

**NIMBLE**: The proposed algorithm.

For fair comparison, each algorithm is run every 5 seconds for PF-utility maximization over the same network and user-state. For each scenario, the experiment is repeated 5 times by varying user mobility traces and the average results are presented.

#### 7.4.3.1 System throughput

Figure 7.6a shows the average throughput of the system. As LSFN does not consider SFN clustering, the RBs available to eMBMS were restricted by the congested eNB and all the users had to be served with limited RBs. On the other hand, BoLTE lacks user grouping and could not assign rates to users commensurate to their channel conditions. NIMBLE and RTOP considered both these factors and hence increased average throughput by up to 150%.

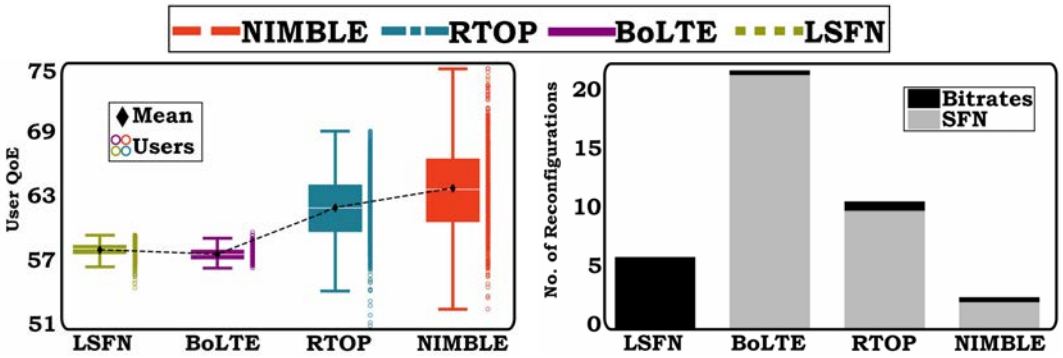
NIMBLE achieved slightly higher throughput than RTOP. This is because RTOP incurs slightly higher drops on the users (Figure 7.6b) due to frequent network reconfigurations (Figure 7.7b). By using the control parameter $\delta$, NIMBLE stabilizes the network and user state, and increases the successfully received frames and hence the average throughput.

#### 7.4.3.2 Lost frames and stalls

The implementation of each algorithm in the testbed ran every 5 seconds and had an explicit or implicit constraint to avoid frame losses or stalls. Hence all the algorithms reacted well to the possible variation in user channel condition and mobility. Almost all the users lost less than 30 frames (0.4%) and stalled for less than 3 seconds (1%) during the five-minutes of simulation (Figure 7.6b and 7.6c).

#### 7.4.3.3 Switches in bitrates

Based on subjective evaluation, less than one switch per minute is acceptable by users and does not annoy them [70]. This implies 5 switches per the 5-minute simulation duration which was achieved by only 40% users with BoLTE and around 55% with LSFN or RTOP. However, with NIMBLE around 98% of users encountered less than 5 switches (Figure 7.6d). NIMBLE achieved lower switch count while still serving users with the highest throughput (Figure 7.6a). This is because unlike other algorithms, NIMBLE takes into account the impact of switching bitrates when reconfiguring the network and avoids too many switches which can annoy users and have a negative effect on video QoE.




(a) User QoE: Box And Scatter Plot (b) Reconfigurations per Minutes

Figure 7.7: Results of QoE and network reconfiguration: *NIMBLE reduces network reconfigurations and increases QoE for end-users.*

#### 7.4.3.4 QoE and fairness

With least number of switches and highest throughput, NIMBLE provided the best QoE for users, followed by RTOP which achieved similar throughput but switched bitrates more often than NIMBLE. BoLTE and LSFN ignored the impact of switches and also failed to maximize the throughput as they do not consider combined advantage of SFN clustering and user grouping, hence had lower QoE values. In comparison to LSFN, NIMBLE increased the average user QoE by 13% (Figure 7.7a).

In terms of f-index fairness (Table 7.3), LSFN and BoLTE performed slightly better, but this is because most of the users had *equally bad* experience. This makes the system seem fair but actually performs worse because of the inability to react to the underlying network or user state. NIMBLE assigned bitrates to users commensurate to their channel condition and available RBs across eNBs.

To further analyze this, the number of users that achieved a higher (or lower) QoE with other algorithms in comparison to NIMBLE are measured. As shown in Table 7.3 (Row 2), NIMBLE improved QoE for more than 1000 users at the expense of only 54 users for LSFN, 30 users for BoLTE and 135 users for RTOP. Also, as shown in Figure 7.7a, the decrease in QoE is marginal for most of the users.

#### 7.4.3.5 Network reconfigurations

NIMBLE incurred less overhead cost (Figure 7.7b) by reconfiguring the network less often, especially in comparison to BoLTE and RTOP. While the actual ac-

Table 7.3: Fairness analysis of NIMBLE

| **Algorithm** | **LSFN** | **BoLTE** | **RTOP** | **NIMBLE** |
|---|---|---|---|---|
| f-index | 0.96 | 0.97 | 0.86 | 0.83 |
| vs NIMBLE | 1116 ↓ 54 ↑ | 1154 ↓ 30 ↑ | 1064 ↓ 135 ↑ | - |

ceptable overhead cost depends on operators and their network preferences, these results show that with efficient network management and configuration, NIMBLE was able to achieve higher user QoE while keeping the reconfigurations to minimum. This reduction is due to the control parameters (Section 7.3.4) considered by NIMBLE when reconfiguring the network.

## Summary

In this chapter, an optimization model was formulated and a heuristics-based algorithm was proposed for eMBMS network reconfiguration that aims to maximize the end-user QoE rather than just the assigned bitrates or network-layer throughput. Key design challenges were addressed that arise due to dynamic nature of cellular networks and user mobility or varying channel condition over time. Control parameters were propose and their impact on various performance metrics was analyzed. It was shown how these parameters can be adjusted for a better trade-off between network responsiveness, efficiency and stability. The proposed algorithm, NIMBLE, was compared with state-of-the-art approaches and results showed the achieved improvement in user QoE and various underlying metrics. NIMBLE performed better because, in addition to control parameters, it jointly optimized SFN clustering and user grouping problems when reconfiguring the network and allocating resources to different eMBMS video sessions. Overall NIMBLE achieved a 150% increase in user throughput, 75% reduction in user switches, 15% increase in average user QoE, all while reducing the network reconfiguration count by 90%.

# Chapter 8

# Conclusion and future work

## 8.1 Summary

This thesis focused on improving the network resource utilization, system utilization and end-user experience for live video streaming in core, wired access and radio access networks. The traditional Internet architecture is best suited for end-to-end unicast transmissions as they provide high control to content servers over the users, and maintain user and data privacy. However, IP unicast fails to take advantage of temporal commonality in live streaming user-sessions and results in waste of resources due to redundant and duplicate re-transmissions. This may be acceptable for low-volume traffic services but with the proliferation of demand for high-definition live streaming, flash crowd events occur frequently and increase the peak bandwidth requirements. From a functional point of view, multicast at network or physical layer would be a desirable choice to avoid sending the same video stream to potentially millions of concurrent users.

However, due to the rigid nature of traditional Internet and lack of key features in existing multicast solutions, its use is currently restricted to intra-domain services, while Over The Top (OTT) streaming services still suffer from inefficient resource management. On the other hand, the 3GPP standard, Evolved Multimedia Broadcast Multicast Service (eMBMS), enables multicast for physical-layer transmission in cellular networks, but the state-of-the-art resource allocation algorithms are either too complex to solve in real-time, yield sub-optimal results, are based on fairly-static network scenarios, maximize network-level throughput rather than overall user QoE or do not consider the time-variance of the network state or mobile users. The work in thesis addressed the limitations of existing so-

lutions and as such proposed various architectural designs, network components, optimization models and algorithms. Performance evaluation showed significant reduction in resource consumption and improvement in metrics associated with user experience.

## 8.2 Key contributions and findings

The following contributions were made in the area of OTT live video streaming and in particular for establishing realistic and practically deploy-able services:

1. The existing literature on live streaming was surveyed, especially the approaches that consider multicast-based transmission. Limitations and challenges were identified that restrict the usage of multicast in different network domains. User expectations and QoE requirements to measure the performance of a system were established (Chapter 2).

2. A novel Internet architecture, mCast, was proposed that adapts SDN to enable network-layer multicast for OTT streaming services. mCast merges the flexibility and control of SDN with resource efficiency of multicast to reduce inter-domain and intra-domain traffic for Internet Service Providers (ISP) and Content Delivery Networks (CDN). mCast is transparent to the clients and provides CDNs with the same level of control, over user sessions, as in IP unicast. A communication framework between ISPs and CDNs was proposed to enable mCast while retaining user and data privacy. A prototype of mCast was built over an emulated testbed and experiments were conducted to show the feasibility, scalability and gains of mCast. mCast was compared with IP unicast in different network topologies with up to 1000 video clients located in an ISP and streaming real videos from content servers in CDN. For the same network link capacities, where unicast users suffered up to 40% frame losses, mCast was able to serve all the users effectively with no network losses. mCast achieved this by serving through multicast and reducing the ISP link utilization by more than 50%. mCast further improved user experience by reducing video start-up delays by 80% (Chapter 4).

3. A device-aware network-assisted optimal streaming service, Danos, was proposed for live video streaming. Danos is a novel service that uses multicast at the network layer, similar to mCast, and provides dynamic bitrate adaptive streaming at the application layer, similar to DASH. Danos introduces several key components to the mCast architecture that are essential for building a cross-

layer optimal service and handling users with heterogeneous device capabilities. Danos gathers details of various ISP and CDN operation constraints as well as user-device specification and uses this information to respond to any event that can affect user experience or result in packet losses. A multi-objective optimization problem was formulated to accommodate these constraints and maximize the perceived video quality of all the users while minimizing the consumption of the ISP network links. A scalable real-time guided model was proposed to solve the problem optimally. The performance analysis of the model showed that it can be solved in the order of milliseconds for millions of users (Chapter 5).

4. As there is no feedback loop between users and content servers in multicast, an approach was proposed for Danos that presents how the bitrate of users can still be switched smoothly without causing frame jitter. Furthermore, because start-up delays have a high negative impact on user experience, a mechanism was devised for Danos to synchronize various components in the parallelized architecture, that minimizes the start-up delays for clients when optimizing and reconfiguring the network. A prototype of Danos was implemented over an emulated testbed and the performance was compared against mCast for real-world scenarios, including flash crowd events and cross-traffic. Danos handled both events efficiently and improved average goodput by up to 70%. As new users joined the video session, Danos continued to react accordingly through periodic optimization. mCast is unaware of the network link state and resulted in up to 45% losses in congested network scenario, whereas Danos eliminated these losses by either re-routing users or reducing bitrates in events of congestion (Chapter 5).

5. For eMBMS in cellular networks, a real-time optimization algorithm, RTOP, was proposed that solves a joint optimization model at a particular network instance. Unlike existing work in the literature, the proposed optimization model considers the combined impact and inter-dependence of various network design features such as: which eNBs to synchronize and form Single Frequency Networks (SFN); how to share resources among users with disparate channel conditions and; how to handle the impact of eMBMS decisions on each eNB's non-eMBMS users. The joint optimization allowed for better resource allocation of the scarce wireless spectrum. A scalable heuristics-based algorithm was proposed that mostly yields optimal results or else results with less than a 1% gap from optimal solution. The algorithm is independent of the number of users and was able to solve the problem in less than 500ms for typical eMBMS system configuration for thousands of users. Simulation comparison of RTOP with state-of-the-art eMBMS approaches showed an improvement of 14% in Proportional Fairness (PF), achieved by assigning high

bitrates to 7 times more users. Average user bitrate was increased by 90% and service degradation was eliminated (Chapter 6).

6. A network optimization framework for eMBMS-based live user QoE (NIMBLE) was proposed. Similar to approaches in the literature, RTOP assumes a fairly-static network when solving the optimization problem and ignores the impact of variability in network or user state over time. In NIMBLE, first a QoE optimization model was formulated that solves the network configuration problem while considering the three fundamental factors of QoE: video bitrates assigned to users, video frames dropped or skipped by users and, switches in video bitrates encountered by users. Then a scalable heuristics-based algorithm was proposed, that can solve the optimization model in real-time regardless of the number of users and their mobility pattern. Finally, stability parameters were introduced to consider the trade-off between frequent network reconfiguration and responsiveness to network state or user channel condition. NIMBLE was compared against RTOP and other recent research in real-world scenarios with varying user-mobility patterns. In comparison to RTOP, 88% users had a better viewing experience with NIMBLE, mostly due to less switches in bitrates. Furthermore, NIMBLE reduced the network reconfiguration rate by 90% which made the network more stable and reduced the overhead cost (Chapter 7).

7. An emulator for evaluating SDN multicast architectures and algorithms for live video streaming (eSMAL) was designed. eSMAL can emulate multi-domain network topologies using Mininet and provides implementation of SDN controller, live video servers and, lightweight live video clients. eSMAL provides a panoramic GUI for modifying various evaluation parameters and monitoring the effect on output in form of graphs and statistics. An additional, open-source network animator GUI MiniNAM, was also developed and integrated in eSMAL. MiniNAM displays the network topology and real-time traffic flows with packet level information for the live traffic. eSMAL can also support large-scale experiments by disabling the GUIs and decoding features of video clients with logs saved for post-processing. Up to 1000 multicast video clients, running as separate user instances, were successfully tested on a single virtual machine. A prototype of mCast and Danos was implemented on eSMAL for evaluation, comparison and demonstration purposes (Chapter 3).

8. For evaluating eMBMS based resource allocation algorithms, such as RTOP and NIMBLE, a discrete-event based LTE physical layer simulator was built. The simulator is capable of analyzing physical layer eMBMS features such as

user grouping and SFN clustering which are not provided by the commonly available open-source network simulators. An animation tool was also developed for viewing network configuration and user state over the simulation duration. The animator takes the network configuration traces and user received-frame traces as an input and displays various statistics such as the network topology, user mobility pattern, SFN cluster configuration, bitrates being transmitted, the RBs and MCS used to transmit each bitrate, bitrates received by users, channel condition (MCS) of users, lost frames and switches experienced by users (Chapter 3).

## 8.3 Future work

The contributions made in this thesis have introduced innovative ways to deliver live video streams from content servers to end-users. Various directions and dimensions were covered for core network as well as wired and radio access network. The proposed work can be extended in the following ways:

- When constructing multicast trees, mCast uses a simple extension of Dijkstra's algorithm to find routes to users. As the architecture of mCast is modular, this algorithm can be replaced without modifying the rest of the modules in the architecture. A more dynamic algorithm can be implemented on the routing module that further improves resource utilization e.g. by balancing load across forwarding nodes. Furthermore, as users leave or join the groups, the multicast tree becomes sub-optimal. Similar to Danos, a periodic reconfiguration of the tree can update the network in a better way. The dynamics of such an approach can be studied for mCast.
- Although the video clients and servers in mCast are designed to support Scalable Video Coding (SVC), the impact of SVC on the network and possible resource management strategies have not been explored. Some research in the literature has proposed using SVC or Multiple Description Coding (MDC) for multicast in SDN. Such approaches can be incorporated in Danos to avoid the need of multiple user groups per video and instead enable adaptive streaming by controlling the number of SVC layers to transmit to each user. Elaborate optimization models for such an approach are not defined in the literature and can be formulated and implemented in Danos.
- For congested networks, mCast and Danos rely on multicast at the network-layer to reduce network load and provide sustainable streaming services. If

the available resources are still insufficient, network coding may be used to improve network's throughput, efficiency and scalability. Network nodes with insufficient capacity can encode packets or merge packets using an algorithm and transmit the accumulated result. A destination network node can be chosen which will receive and decode the accumulated result, using the same algorithm, to recover the original packets. Such an approach requires fewer transmissions and hence improves efficiency. Finding optimal coding solutions for such scenarios remains an open problem.

- NIMBLE optimizes eMBMS network configuration by running an algorithm periodically. Although, the algorithm runs in real-time regardless of the number of users, this still requires active network agents. For some specific use cases of eMBMS, especially where the network state and user behavior is predictable, machine and deep learning approaches can be adapted for network management with reinforcement techniques. The regression-based supervised approach used by RTOP and NIMBLE to analyze resource and utility relation, can serve as a useful pre-requisite for data acquisition and training.

- With the increase in wireless home users, e.g. over Wi-Fi, it is important to explore methods for efficient multicast delivery to wireless users, other than in cellular networks. Generally, ISP control terminates at user premises and streams are delivered over user-owned wireless access points to potentially multiple devices. Enabling SDN over such access points can facilitate better scheduling of wireless resources and smooth switching between unicast and multicast. NIMBLE is designed for cellular networks, however it can be adapted for other wireless standards as the nature of resource scheduling and user experience is similar.

- The proposed testbed for Danos (and mCast) emulates network topologies and virtual instances of users. The testbed provides insight into the working mechanism and performance of the proposed system in a controlled environment. Further work can integrate Danos in a real-world deployment of globally distributed live streaming users across SDN-enabled ISPs. Testing on such a setup can provide more accurate and concise results, due to more realistic cross-traffic, network topologies and constraints. Similarly, feedback from real-world users can provide a better insight in to the end-user experience, possibly with subjective QoE evaluation.

# References


[1] Cisco, *Visual Networking Index: Forecast and Trends, 2017-2022 (White Paper)*, Feb 2019.

[2] *Twitch*. `https://twitchtracker.com/` (Retrieved 28-Aug-2019).

[3] Ericsson Consumer Lab, "TV and Media - The evolving role of TV and media in consumers' everyday lives," tech. rep., Ericsson, November 2016.

[4] B. Wang, X. Zhang, G. Wang, H. Zheng, and B. Zhao, "Anatomy of a Personalized Live Streaming System," in *Proceedings of the 2016 Internet Measurement Conference (IMC)*, pp. 485–498, ACM, 2016.

[5] R. Sweha, V. Ishakian, and A. Bestavros, "Angelcast: Cloudbased Peer-assisted Live Streaming using Optimized Multi-tree Construction," in *Proceedings of the 3rd Multimedia Systems Conference (MMSys)*, pp. 191–202, ACM, 2012.

[6] C. Diot, B. Levine, B. Lyles, H. Kassem, and D. Balensiefen, "Deployment issues for the IP multicast service and architecture," *IEEE Network*, vol. 14, pp. 78–88, January 2000.

[7] M. Cha, P. Rodriguez, J. Crowcroft, S. Moon, and X. Amatriain, "Watching Television over an IP network," in *In Proceedings of the 8th ACM SIGCOMM conference on Internet measurement (IMC)*, pp. 71–84, ACM, 2008.

[8] E. Haleplidis, K. Pentikousis, S. Denazis, J. Salim, D. Meyer, and O. Koufopavlou, *Software-Defined Networking (SDN): Layers and Architecture Terminology*, January 2015. `https://www.rfc-editor.org/rfc/rfc7426.txt`.

[9] Cisco Public, "Global Cloud Index: Forecast and Methodology, 2016-2021 (White Paper)," tech. rep., Cisco, 2018.

[10] Ericsson, *Ericsson Mobility Report*, June 2019. `https://www.ericsson.com/en/mobility-report/reports/june-2019` (Retrieved 09-Oct-2019).

[11] 3GPP, "Multimedia Broadcast/Multicast Service (MBMS); Architecture and functional description," Tech. Rep. 23.246, 3GPP, 2015.

[12] J. Chen, M. Chiang, J. Erman, G. Li, K. Ramakrishnan, and R. Sinha, "Fair and optimal resource allocation for LTE multicast (eMBMS): Group partitioning and dynamics," in *IEEE Conference on Computer Communications (INFOCOM)*, October 2015.

[13] J. Yoon, H. Zhang, S. Banerjee, and S. Rangarajan, "Muvi: A multicast video delivery scheme for 4G cellular networks," *ACM International Conference on Mobile Computing and Networking (MobiCom)*, 2012.

[14] R. Sivaraj, M. Arslan, K. Sundaresan, S. Rangarajan, and P. Mohapatra, "BoLTE: Efficient network-wide LTE broadcasting," *IEEE International Conference on Network Protocols (ICNP)*, April 2017.

[15] R. Afolabi, A. Dadlani, and K. Kim, "Multicast Scheduling and Resource Allocation Algorithms for OFDMA-Based Systems: A Survey," *IEEE Communications Surveys & Tutorials*, vol. 15, First 2013.

[16] Y. Yang, M. Kim, and S. Lam, "Optimal partitioning of multicast receivers," *IEEE International Conference on Network Protocols (ICNP)*, November 2000.

[17] E. Nygren, R. Sitaraman, and J. Sun, "The Akamai network: a platform for high-performance internet applications," *ACM SIGOPS Operating Systems Review*, vol. 44, no. 3, pp. 2–19, 2010.

[18] J. Martin, Y. Fu, N. Wourms, and T. Shaw, "Characterizing Netflix Bandwidth Consumption," in *2013 IEEE 10th Consumer Communications and Networking Conference (CCNC)*, pp. 230–235, Jan 2013.

[19] K. Pires and G. Simon, "YouTube live and Twitch: a tour of user-generated live streaming systems," in *Proceedings of the 6th ACM Multimedia Systems Conference (MMSys)*, pp. 225–230, ACM, 2015.

[20] H. Parmar and M. Thornburgh, "Adobe's Real Time Messaging Protocol," *Copyright Adobe Systems Incorporated*, pp. 1–52, 2012.

[21] J. Quinlan, A. Zahran, and C. Sreenan, "Datasets for AVC (H.264) and HEVC (H.265) Evaluation of Dynamic Adaptive Streaming over HTTP (DASH)," *ACM Multimedia Systems Conference (MMSys)*, May 2016.

[22] J. Iyengar and M. Thomson, "QUIC: A UDP-Based Multiplexed and Secure Transport," tech. rep., IETF, July 2019. `https://datatracker.ietf.org/doc/draft-ietf-quic-transport/`.

[23] *Global Media Format Report*, 2019. `http://www.encoding.com/files/2019-Global-Media-Formats-Report.pdf` (Retrieved 28-Aug-2019).

[24] M. Cha, H. Kwak, P. Rodriguez, Y.-Y. Ahn, and S. Moon, "Analyzing the video popularity characteristics of large-scale user generated content systems," *IEEE Transactions on Networking*, vol. 17, no. 5, pp. 1357–1370, 2009.

[25] J. Godas, B. Field, A. Bernstein, S. Desai, T. Eckert, and H. Parandekar, "IP Multicast In Cable Networks (White Paper)," tech. rep., Cisco Systems, 2005.

[26] T. Bova and T. Krivoruchka, "Reliable UDP Protocol," tech. rep., IETF, February 1999. `https://www.ietf.org/proceedings/44/I-D/draft-ietf-sigtran-reliable-udp-00.txt`.

[27] R. Sweha, V. Ishakian, and A. Bestavros, "NOVA: QoE-driven optimization of DASH-based video delivery in networks," in *IEEE Conference on Computer Communications (INFOCOM)*, IEEE, 2014.

[28] D. Bhat, A. Rizk, M. Zink, and R. Steinmetz, "SABR: Network-Assisted Content Distribution for QoE-Driven ABR Video Streaming," in *Proceedings of the 8th ACM on Multimedia Systems Conference (MMSys)*, pp. 62–75, ACM, 2017.

[29] Y. Sani, A. Mauthe, and C. Edwards, "Adaptive Bitrate Selection: A Survey," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2985–3014, 2017.

[30] D. Stephen, E. Deborah, F. Dino, J. Van, L. Ching-gung, and W. Liming, "Protocol Independent Multicast (PIM): Protocol specification," *Internet Draft RFC*, 1995.

[31] W. Fenner, *Internet Group Management Protocol, Version 2*, November 1997. `https://tools.ietf.org/html/rfc2236`.

[32] M. Moyer, J. Rao, and P. Rohatgi, "A survey of security issues in multicast communications," *IEEE Network*, vol. 13, no. 6, pp. 12–23, 1999.

[33] P. Gill, M. Schapira, and S. Goldberg, "A survey of interdomain routing policies," *ACM SIGCOMM Computer Communication Review*, vol. 44, pp. 28–34, January 2014.

[34] K. Almeroth, "Managing IP multicast traffic: A first look at the issues, tools, and challenges," in *IP Multicast Initiative white paper*, 1999.

[35] E. Lee and S. Park, "Internet group management protocol for IPTV services in passive optical network," *IEICE Transactions on Communications*, vol. 93, no. 2, pp. 293–296, 2010.

[36] D. Li, Y. Li, J. Wu, S. Su, and J. Yu, "ESM: Efficient and scalable data center multicast routing," *ACM Transactions on Networking (TON)*, vol. 20, no. 3, pp. 944–955, 2012.

[37] I. Hwang, A. Nikoukar, K. Chen, A. T. Liem, and C. Lu, "QoS enhancement of live IPTV using an extended real-time streaming protocol in Ethernet passive optical networks," *IEEE Journal of Optical Communications and Networking*, vol. 6, pp. 695–704, August 2014.

[38] R. Srinivasan, V. Vaidehi, R. Rajaraman, S. Kanagaraj, C. Kalimuthu, and R. Dharmaraj, "Secure Group Key Management Scheme for Multicast Networks," *International Journal of Network Security*, vol. 11, pp. 33–38, July 2010.

[39] P. Dingyi, S. Arto, and D. Cunsheng, *Chinese remainder theorem: applications in computing, coding, cryptography*. World Scientific, 1996.

[40] M. Hosseini, D. T. Ahmed, S. Shirmohammadi, and N. D. Georganas, "A Survey of Application-Layer Multicast Protocols," *IEEE Communications Surveys & Tutorials*, vol. 9, pp. 58–74, September 2007.

[41] B. Furht, R. Westwater, and J. Ice, "IP simulcast: A new technique for multimedia broadcasting over the Internet," *Journal of Computing and Information Technology*, vol. 6, no. 3, pp. 245–254, 1998.

[42] Y. Liu, Y. Guo, and C. Liang, "A survey on peer-to-peer video streaming systems," *Peer-to-Peer Networking and Applications*, vol. 1, no. 1, pp. 18–28, 2008.

[43] G. Gheorghe, R. Cigno, and A. Montresor, "Security and privacy issues in P2P streaming systems: A survey," *Peer-to-Peer Networking and Applications*, vol. 4, no. 2, pp. 75–91, 2011.

[44] Open Networking Foundation, *OpenFlow Switch Specification 1.3.0*, June 2012.

[45] R. Enns, M. Bjorklund, and J. Schoenwaelder, "Network configuration protocol (NETCONF)," *Network*, 2011.

[46] B. Pfaff and B. Davie, "The Open vSwitch Database Management Protocol.," tech. rep., RFC 7047, 2013. `https://www.rfc-editor.org/rfc/rfc7426.txt`.

[47] J. Medved, R. Varga, A. Tkacik, and K. Gray, "Opendaylight: Towards a Model-Driven SDN Controller architecture," in *Proceeding of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2014*, pp. 1–6, IEEE, 2014.

[48] P. Berde, M. Gerola, J. Hart, Y. Higuchi, M. Kobayashi, T. Koide, B. Lantz, B. O'Connor, P. Radoslavov, W. Snow, *et al.*, "ONOS: Towards an Open, Distributed SDN OS," in *Proceedings of the third workshop on Hot topics in software defined networking*, pp. 1–6, ACM, 2014.

[49] *Ryu SDN framework*, 2017. `https://osrg.github.io/ryu/` (Retrieved 29-Sep-2019).

[50] D. Kotani, K. Suzuki, and H. Shimonishi, "A design and implementation of openflow controller handling ip multicast with fast tree switching," *IEEE/IPSJ 12th International Symposium on Applications and the Internet (SAINT)*, July 2012.

[51] C. Marcondes, T. Santos, A. Godoy, C. Viel, and C. Teixeira, "Castflow: Clean-slate multicast approach using in-advance path processing in programmable networks," *Proc. of the 2012 IEEE Symposium on Computers and Communications (ISCC)*, pp. 94–104, July 2012.

[52] A. Craig, B. Nandy, I. Lambadaris, and P. Ashwood-Smith, "Load balancing for multicast traffic in SDN using real-time link cost modification," in *2015 IEEE International Conference on Communications (ICC)*, pp. 5789–5795, June 2015.

[53] J. Jiang, H. Huang, J. Liao, and S. Chen, "Extending dijkstra's shortest path algorithm for software defined networking," *16th Asia-Pacific Network Operations and Management Symposium (APNOMS)*, September 2014.

[54] S. Zhang, Q. Zhang, H. Bannazadeh, and A. Leon-Garcia, "Routing algorithms for network function virtualization enabled multicast topology on sdn," *IEEE Transactions on Network and Service Management*, vol. 12, pp. 580–594, August 2015.

[55] M. Reza, J. Reza, K. Manijeh, and A. Reza, "A novel multicast traffic engineering technique in SDN using TLBO algorithm," *Telecommunication Systems*, vol. 68, pp. 583–592, July 2018.

[56] F. Corasa, J. Domingo-Pascuala, F. Mainob, D. Farinaccib, and A. Aparicioa, "Castflow: Clean-slate multicast approach using in-advance path processing in programmable networks," *Elsevier Journal Computer Networks*, vol. 59, pp. 153–170, Feburary 2014.

[57] E. Yang, Y. Ran, S. Chen, and J. Yang, "A Multicast Architecture of SVC Streaming Over OpenFlow Networks," *IEEE Global Communications Conference (GLOBECOM)*, December 2014.

[58] K. Noghani and M. Sunay, "Streaming multicast video over software-defined networks," *Proc. of the 2014 IEEE 11th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*, pp. 551–556, October 2014.

[59] B. Yan, S. Shi, Y. Liu, W. Yuan, H. He, R. Jana, Y. Xu, and H. Chao, "LiveJack: Integrating CDNs and Edge Clouds for Live Content Broadcasting," in *Proceedings of the 2017 ACM on Multimedia Conference (MM)*, pp. 73–81, 2017.

[60] J. Ruckert, J. Blendin, and D. Hausheer, "Software-Defined Multicast for Over-the-Top and Overlay-based Live Streaming in ISP Networks," *Journal of Network and Systems Management*, vol. 23, pp. 280–308, April 2015.

[61] F. Kelly, "Charging and rate control for elastic traffic," *European Transactions on Telecommunications*, vol. 8, pp. 33–37, Jan 1997.

[62] T. Mladenov, S. Nooshabadi, and K. Kim, "Efficient Incremental Raptor Decoding Over BEC for 3GPP MBMS and DVB IP-Datacast Services," *IEEE Transactions on Broadcasting*, vol. 57, June 2011.

[63] F. Hou, L. Cai, P. Ho, X. Shen, and J. Zhang, "A cooperative multicast scheduling scheme for multimedia services in IEEE 802.16 networks," *IEEE Trans. on Wireless Communications*, vol. 8, March 2009.

[64] H. Won, H. Cai, D. Eun, K. Guo, A. Netravali, I. Rhee, and K. Sabnani, "Multicast scheduling in cellular data networks," *IEEE Transactions on Wireless Communications*, vol. 8, September 2009.

[65] J. Monserrat, J. Calabuig, A. Aguilella, and D. Barquero, "Joint Delivery of Unicast and E-MBMS Services in LTE Networks," *IEEE Transactions on Broadcasting*, vol. 58, April 2012.

[66] A. Snoeren, D. Andersen, and H. Balakrishnan, "Fine-Grained Failover Using Connection Migration," in *3rd Conference on USENIX Symposium on Internet Technologies and Systems*, vol. 1, pp. 19–19, 2001.

[67] ITU, "Vocabulary for performance, quality of service and quality of experience," Tech. Rep. P.10/G.100, ITU, 2017.

[68] Z. Duanmu, A. Rehman, and Z. Wang, "A Quality-of-Experience Database for Adaptive Video Streaming," *IEEE Transactions on Broadcasting*, vol. 64, pp. 474–487, June 2018.

[69] K. Shunmuga and S. Ramesh, "Video Stream Quality Impacts Viewer Behavior: Inferring Causality Using Quasi-experimental Designs," in *Proceedings of the 2012 Internet Measurement Conference*, IMC '12, pp. 211–224, 2012.

[70] N. Cranley, P. Perry, and L. Murphy, "User perception of adapting video quality," *International Journal of Human-Computer Studies*, vol. 64, no. 8, pp. 637–647, 2006.

[71] A. Zahran, J. Quinlan, K. Ramakrishnan, and C. Sreenan, "SAP: Stall-aware pacing for improved DASH video experience in cellular networks," in *Proceedings of the 8th ACM on Multimedia Systems Conference (MMSys)*, pp. 13–26, 2017.

[72] A. Takahashi, D. Hands, and V. Barriac, "Standardization Activities in the ITU for a QoE Assessment of IPTV," *IEEE Communications Magazine*, vol. 46, no. 2, pp. 78–84, 2008.

[73] R. Serral-Gracia, E. Cerqueira, M. Curado, M. Yannuzzi, E. Monteiro, and X. Masip-Bruin, "An Overview of Quality of Experience Measurement

Challenges for Video Applications in IP Networks," in *International Conference on Wired/Wireless Internet Communications*, pp. 252–263, Springer, 2010.

[74] X. Tao, L. Dong, Y. Li, J. Zhou, N. Ge, and J. Lu, "Real-Time Personalized Content Catering via Viewer Sentiment Feedback: A QoE Perspective," *IEEE Network*, vol. 29, no. 6, pp. 14–19, 2015.

[75] Q. Huynh-Thu and M. Ghanbari, "Scope of Validity of PSNR in Image/Video Quality Assessment," *Electronics letters*, vol. 44, no. 13, pp. 800–801, 2008.

[76] A. Hore and D. Ziou, "Image Quality Metrics: PSNR vs. SSIM," in *2010 20th International Conference on Pattern Recognition*, pp. 2366–2369, IEEE, 2010.

[77] M. Pinson and S. Wolf, "A New Standardized Method for Objectively Measuring Video Quality," *IEEE Transactions on broadcasting*, vol. 50, no. 3, pp. 312–322, 2004.

[78] P. Callet, C. Viard-Gaudin, S. Pechard, and E. Caillault, "No Reference and Reduced Reference Video Quality Metrics for End to End QoS Monitoring," *IEICE Transactions on Communications*, vol. 89, no. 2, pp. 289–296, 2006.

[79] D. Mocanu, G. Exarchakos, H. Ammar, and A. Liotta, "Reduced Reference Image Quality Assessment via Boltzmann Machines," in *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, pp. 1278–1281, IEEE, 2015.

[80] Y. Liu, S. Dey, F. Ulupinar, M. Luby, and Y. Mao, "Deriving and Validating User Experience Model for DASH Video Streaming," *IEEE Transactions on Broadcasting*, vol. 61, pp. 651–665, Dec 2015.

[81] W. Cherif, A. Ksentini, D. Negru, and M. Sidibe, "A_PSQA: Efficient Real-Time Video Streaming QoE Tool in A Future Media Internet Context," in *2011 IEEE International Conference on Multimedia and Expo*, pp. 1–6, IEEE, 2011.

[82] R. Jain, A. Durresi, and G. Babic, "Throughput fairness index: An explanation," in *ATM Forum contribution*, vol. 99, 1999.

[83] T. Hoßfeld, L. Skorin-Kapov, P. E. Heegaard, and M. Varela, "Definition of QoE Fairness in Shared Systems," *IEEE Communications Letters*, vol. 21, pp. 184–187, Jan 2017.

[84] Y. Bejerano, C. Raman, C. Yu, V. Gupta, C. Gutterman, T. Young, H. Infante, Y. Abdelmalek, and G. Zussman, "DyMo: Dynamic monitoring of large scale LTE-Multicast systems," *IEEE INFOCOM*, May 2017.

[85] Z. Lu and H. Yang, *Unlocking the Power of OPNET Modeler*. Cambridge University Press, 2012.

[86] L. Breslau, D. Estrin, H. Yu, K. Fall, S. Floyd, J. Heidemann, A. Helmy, P. Huang, S. McCanne, K. Varadhan, *et al.*, "Advances in network simulation," *IEEE Computer*, no. 5, pp. 59–67, 2000.

[87] G. Riley and T. Henderson, *The ns-3 Network Simulator*. Springer Berlin Heidelberg, 2010.

[88] D. Estrin, M. Handley, J. Heidemann, S. McCanne, Y. Xu, and H. Yu, "Network visualization with NAM, the vint network animator," *IEEE Computer*, vol. 33, no. 11, pp. 63–68, 2000.

[89] *NetAnim*. `https://www.nsnam.org/wiki/NetAnim` (Retrieved 09-Oct-2019).

[90] B. Lantz, B. Heller, and N. McKeown, "A network in a laptop: rapid prototyping for software-defined networks," *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*, October 2010.

[91] R. Fontes, A. Oliveira, P. Sampaio, T., Pinheiro, and R. Figueira, "Authoring of OpenFlow networks with visual network description (SDN version)(WIP)," in *Proceedings of the 2014 Summer Simulation Multiconference*, p. 22, 2014.

[92] *Mininet 2.3.0*. `http://mininet.org/` (Retrieved 29-Oct-2018).

[93] J. Quinlan, A. Reviakin, A. Khalid, K. Ramakrishnan, and C. Sreenan, "D-LiTE-ful: An evaluation platform for DASH QoE for SDN-enabled ISP offloading in LTE," in *Proceedings of the Tenth ACM International Workshop on Wireless Network Testbeds, Experimental Evaluation, and Characterization (WINTECH)*, pp. 91–92, 2016.

[94] *The Internet Topology Zoo*. `http://www.topology-zoo.org/` (Retrieved 03-Feb-2019).

[95] A. Detti, G. Bianchi, C. Pisa, F. Proto, P. Loreti, W. Kellerer, S. Thakolsri, and J. Widmer, "SVEF: An Open-Source Experimental Evaluation Framework for H.264 Scalable Video Streaming," *IEEE Symposium on Computers and Communications*, July 2009.

[96] J. Reichel, H. Schwarz, and M. Wien, "Joint scalable video model jsvm-9," *Joint Video Team*, January 2007.

[97] D. Maltz, J. Broch, D. Johnson, Y. Hu, and J. Jetcheva, "A performance comparison of multi-hop wireless ad hoc network routing protocols," in *ACM International Conference on Mobile Computing and Networking (MobiCom)*, vol. 114, 1998.

[98] S. Rappaport *et al.*, *Wireless Communications: Principles and Practice*, vol. 2. Prentice Hall PTR New Jersey, 1996.

[99] J. Ikuno, M. Wrulich, and M. Rupp, "System level simulation of LTE networks," *IEEE 71st Vehicular Technology Conference*, May 2010.

[100] W. Yang and M. Souryal, *LTE Physical Layer Performance Analysis*. US Department of Commerce, National Institute of Standards and Technology, 2014.

[101] Tech Pro Research, *SDN and the data center: Deployment plans, business drivers, and preferred vendors*, 2016. `https://goo.gl/CEyzBs` (Retrieved 29-Aug-2019).

[102] T. Koch and A. Martin, "Solving steiner tree problems in graphs to optimality," *Networks: An International Journal*, vol. 32, no. 3, pp. 207–232, 1998.

[103] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, *An Architecture for Differentiated Services*, December 1998. `https://www.rfc-editor.org/rfc/pdfrfc/rfc2475.txt.pdf`.

[104] A. Beloglazov and R. Buyya, "Adaptive threshold-based approach for energy-efficient consolidation of virtual machines in cloud data centers," *Proceedings of the 8th International Workshop on Middleware for Grids, Clouds and e-Science*, November 2010.

[105] X. Dimitropoulos, P. Hurley, A. Kind, and M. Stoecklin, "On the 95-percentile billing method," in *International Conference on Passive and Active Network Measurement (PAM)*, pp. 207–216, 2009.

[106] W. B. Norton, *The Internet Peering Playbook: Connecting to the Core of the Internet*. DrPeering Press, 2012.

[107] J. C. Chuang and M. A. Sirbu, "Pricing multicast communication: A cost-based approach," *Journal of Telecommunication Systems*, vol. 17, pp. 281–297, July 2001.

[108] M. Garey and D. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. New York: W. H. Freeman & Co., 1979.

[109] *Gurobi Solver 8.1*, 2018. `http://www.gurobi.com/` (Retrieved 28-Aug-2019).

[110] M. Bari, R. Boutaba, R. Esteves, L. Granville, M. Podlesny, M. Rabbani, Q. Zhang, and M. Zhani, "Data Center Network Virtualization: A Survey," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 2, pp. 909–928, 20013.

[111] *NoviFlow*. `https://noviflow.com/products/noviswitch/` (Retrieved 07-Oct-2019).

[112] R. Schmidt, R. Sadre, and A. Pras, "Gaussian traffic revisited," in *Proceedings of IFIP Networking*, pp. 1–9, 2013.

[113] L. Lao, J. Cui, M. Gerla, and D. Maggiorini, "A comparative study of multicast protocols: top, bottom, or in the middle," *Joint Conference of the IEEE Computer and Communications Societies*, March 2005.

[114] GSA, "LTE Broadcast (eMBMS) Market Update," Tech. Rep. March, Global Mobile Suppliers, 2018.

[115] C. Borgiattino, C. Casetti, C. Chiasserini, and F. Malandrino, "Efficient area formation for LTE broadcasting," *IEEE International Conference on Sensing, Communication, and Networking (SECON)*, June 2015.

[116] EBU, "Delivery of broadcast content over LTE networks," Tech. Rep. TR 027, European Broadcasting Union, 2014.

[117] S. A. Vavasis, "Quadratic programming is in NP," *Information Processing Letters*, vol. 36, October 1990.

[118] F. Chisari, "When football went global: Televising the 1966 world cup," *Historical Social Research / Historische Sozialforschung*, vol. 31, pp. 42–54, 2006.

[119] 3GPP, "Multimedia Broadcast/Multicast Service (MBMS); Protocols and codecs," Tech. Rep. 26.346, 3GPP, 2015.

[120] D. Barquero, W. Li, M. Fuentes, J. Xiong, G. Araniti, C. Akamine, and J. Wang, "5G for Broadband Multimedia Systems and Broadcasting," *IEEE Transactions on Broadcasting*, vol. 65, no. 2, pp. 351–355, 2019.

[121] R. Mok, E. Chan, and R. Chang, "Measuring the quality of experience of HTTP video streaming.," in *Integrated Network Management*, pp. 485–492, 2011.