HAL
open science

Multiple Live Videos Delivery in Underprovisioned Networks

Jiayi Liu

To cite this version:

Jiayi Liu. Multiple Live Videos Delivery in Underprovisioned Networks. Networking and Internet Architecture [cs.NI]. Télécom Bretagne, Université de Rennes 1, 2013. English. NNT : . tel-00978741

HAL Id: tel-00978741

https://theses.hal.science/tel-00978741v1

Submitted on 14 Apr 2014

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

N°d'ordre : **2013telb0284**

Sous le sceau de l' Université européenne de Bretagne

Télécom Bretagne

En habilitation conjointe avec l'Université de Rennes 1

Ecole Doctorale – MATISSE

Multiple Live Videos Delivery in Underprovisioned Networks

Thèse de Doctorat

Mention : Informatique

Présentée par **Jiayi Liu**

Département : Informatique

Laboratoire : IRISA

Directeur de thèse : Antoine Beugnard

Soutenue le 4 Novembre 2013

Jury :

M. Laurent Schumacher, Professeur, University Namur (Rapporteur)
M. David Coudert, Professeur, INRIA (Rapporteur)
M. Antoine Beugnard, Professeur, Télécom Bretagne (Directeur de thèse)
M. Guillaume Pierre, Professeur, Université de Rennes I / IRISA (Président)
M. Cyrcil Concolato, Maître de conférences, Télécom ParisTech (Examinateur)
M. Juan Pedro Munoz Gea, Maître de conférences, Universidad Cartagena (Examinateur)
M. Gwendal Simon, Maître de conférences, Télécom Bretagne (Encadrant)

Acknowledgement

During this thesis, I have been able to work in both Brest and Rennes, where I met many brilliant scientists, colleagues and hearty friends. The work presented in this thesis received contributions from many of them in many ways. I appreciate this chance to express my most sincere thanks to them, who helped me along the way.

First of all, I am heartily thankful to my supervisor Gwendal Simon. This thesis would not have been possible without his help. Many thanks to his patient guidance, encouragement, useful critiques and support from the initial to the final level which enabled me to develop an understanding of the subject. Also, I am very grateful for his warm concern for my life in France, especially his enlightened policy on asking for leaves, which enabled me to spend several Chinese New Year in China, as well as stay bedside my mother throughout her illness.

I would like to express my sincere appreciation to Professor Antoine Beugnard, the director of this thesis, for his advice and supervision. I would like to also thank Annie Gravey for her constant support, and Jean-Marie Bonnin for taking me in the RSM Department in Rennes.

I would like to also thanks the committee members for their effort on making this dissertation finished. Thanks them for accepting the invitation and spending their precious time on improving this thesis.

My sincere appreciation to all my colleagues in our research group: Yaning Liu, Jimmy Leblet, Fen Zhou, Eliya Buyukkaya, YiPing Chen, Zhe Li, Xu Zhang, Wei You, and Karine Pires. I will never forget the pleasant days we spend together: the insightful discussions and the agreeable coffee break chats. Moreover, thanks Qiao Wang for her internship work.

I would like to show my gratitude to my coworkers. Their willingness to give their time so generously has been very much appreciated. First of all, I would like to thank the researchers from the CDN project: Raouf Hamzaoui and Shakeel Ahmad from De Montfort University. My very great appreciation to Professor Raouf Hamzaoui, for his valuable and constructive suggestions during the collaboration. Then, many thanks to professor Catherine Rosenberg and Hanan Shpungin from University of Waterloo, Géraldine Texier from Télécom Bretagne, the collaboration with them is pleasant and fruitful.

My grateful thanks are also extended to all members of both Computer Science Department and Network Security and Multimedia Department of Télécom Bretagne. Thanks Armelle Lannuzel and Anais Renaud for the organization of conference trips. I am also very much indebted to all my friends in Brest and Rennes. Thanks their warm companionship during these years.

Finally, I owe my deepest gratitude to my family. I am indebted to my boyfriend Chen Wang. When I feel frustrated, his warm encouragement always buoyed me up. Thanks to his company in both practical and spiritual life, the six years life in France becomes more easier and full of good remembrance. I am indebted to my mother Ailing Jia, a strong-minded, generous and affectionate single-mother. It is not easy for her to bring me up, and even support my study abroad. Although I was brought up in a single-parent family, I lacked for nothing for she tried all her best in providing me with good living conditions and giving me her entire love. She is the greatest

mother in my heart.

Abstract

With the proliferation of new categories of IP-enabled devices (such as smartphones, tablets, etc.), nowadays, Internet users can ubiquitously access the online video services. This promotes new types of services (for example, the user-generated live video broadcasting), as well as new streaming techniques (such as rate-adaptive streaming). As a result, scientists have observed a formidable growth of Internet traffic dominated by the videos. A consequent challenge is the bandwidth availability problem—a delivery network can be insufficiently provisioned under the heavy transmission burden imposed by the huge volume of video traffic. Such underprovisioning problem is more severe for *live* videos due to its real-time requirement.

In this thesis, we focus on bandwidth efficient video delivery solutions for live streaming in underprovisioned video delivery networks. More specifically, we target to capture the aforementioned trends to find solutions for: (1) user-generated live streaming, and (2) rate-adaptive live streaming. We finally realized the following contributions:

First of all, we built an multioverlay peer-to-peer (P2P) video sharing system which allows ordinary Internet users to broadcast their own live videos. Typically, such a system consists of multiple independent P2P live video streaming systems, and faces the problem of finding a suitable allocation of peer upload bandwidth. So far, no efficient solution has been proposed for the important case when the overall system is underprovisioned, that is, when peers do not have enough upload bandwidth to ensure a diffusion of videos at full quality. We designed various objective functions for this upload bandwidth allocation problem and showed how optimal solutions can be efficiently computed. Simulation results demonstrated that our solutions improve on existing algorithms in terms of video quality.

Then, we studied the problem of delivering live rate-adaptive streams in the content delivery network (CDN). For live streaming in underprovisioned CDN delivery network, the goal is to maximize the throughput of the network. Previous theoretical models that deal with streaming capacity problems do not capture the emerging reality raised by rate-adaptive streaming. Thus, we identified a new *discretized* streaming model, which is more suitable for multiple live video channels in modern CDNs. For this model we formulated a general optimization problem through Integer Linear Programming (ILP) and showed that it is NP-complete. Further, we presented a fast, easy to implement, and near-optimal algorithm with approved approximation ratios for a specific scenario. This work is the first step towards streaming multiple live rate-adaptive videos in CDN and provides a fundamental theoretical basis for deeper investigation.

Last, we further extended the discretized streaming model into an user-centric one which maximizes the overall satisfaction of an user population. The performance of this user-centric discretized streaming model is approved through a set of toy-CDN simulations. Further, we presented a practical system, which efficiently utilizes CDN infrastructure to deliver live video streams to viewers in dynamic and large-scale CDNs. The benefits of our approaches on reducing the CDN infrastructure capacity is validated through a set of realistic trace-driven large-scale simulations.

All in one, this thesis explores bandwidth efficient live video delivery solutions

in underprovisioned delivery network for multiple streaming technologies. The aim is to maximally utilize the bandwidth of relay nodes (peers in P2P and forwarding equipments in CDN) to achieve an optimization goal.

Keywords: Live Streaming, P2P, Multioverlay, CDN, rate-adaptive streaming, DASH, Underprovisioned System.

Contents

Acknowledgement **i**

Abstract **iii**

Contents **vii**

List of Figures **x**

List of Tables **xi**

Résumé **xiii**

1 Introduction **1**
1.1 Status of Online Live Video Streaming 1
1.2 Motivations and Challenges 4
1.2.1 Bandwidth constraint for P2P user-generated live video sharing system 4
1.2.2 Bandwidth constraint for live rate adaptive streaming over CDN 5
1.3 Contributions 6
1.3.1 A Multioverlay Peer-to-Peer Live Video Sharing System 6
1.3.2 Discretized streaming model for delivering live rate adaptive videos over CDN 7
1.3.3 A user-centric live rate adaptive streaming system 7
1.4 Organization of Dissertation 8

2 Online Live Video Streaming **11**
2.1 Live video streaming 11
2.2 Peer-to-Peer Live Video Streaming 13
2.2.1 Overall Perspective 13
2.2.2 Tree-based P2P Systems 13
2.2.3 Mesh-based P2P Systems 15
2.2.4 Multioverlay P2P Systems 17
2.3 Live Video Streaming over CDN 20
2.3.1 CDN: Overall Perspective 20
2.3.2 CDN Architecture for Live Streaming 21
2.3.3 Dynamic Adaptive Streaming over HTTP (DASH) 23

2.3.4 Live DASH over CDN 25

3 Multioverlay P2P Video Sharing: Resource Allocation in Under-Provisioned System 29

3.1 Introduction 29
3.1.1 Multioverlay P2P Live Video Sharing System 29
3.1.2 Our Contribution 30
3.2 System Model 31
3.3 Minimizing Under-provisioning: Bipartite Network Flow 33
3.4 Cost Function Driven Bandwidth Allocation Strategies 35
3.4.1 Minimum-cost Maximum-flow Problem 35
3.4.2 Strategy I: Prioritize Overlay Diversity 35
3.4.3 Strategy II: Prioritize Overlay Popularity 36
3.4.4 Strategy III: Prioritize Fee-Paying Sources 37
3.4.5 Strategy IV: Prioritize User Preference 37
3.4.6 Practical Optimization 38
3.5 Fair Bandwidth Allocation Strategy 38
3.5.1 Problem Formulation 38
3.5.2 Dual Decomposition Solution 39
3.5.3 Distributed Algorithm 40
3.6 Implementation and Practical Details 41
3.6.1 Overall Architecture and Peer Dynamics 41
3.6.2 Peer-Server Communication Overhead 42
3.6.3 Algorithm Computation Time 42
3.7 Evaluation 42
3.7.1 Simulator Platform 43
3.7.2 Simulation Setting 45
3.7.3 Static Scenario Simulation 47
3.7.4 Dynamic Scenario Simulation 51
3.8 Conclusion 54

4 Discretized Streaming Model for Live Rate Adaptive Streaming 57

4.1 Introduction 57
4.2 System Model and Problem Definition 58
4.2.1 Live Rate Adaptive Streaming in CDN 58
4.2.2 Problem Definition 60
4.3 Problem Formulation and Problem Complexity 60
4.3.1 Integer Linear Programming formulation 60
4.3.2 NP-completeness 62
4.4 A Practical Scenario and Algorithm 64
4.4.1 Practical Bundle Delivery in CDN 64
4.4.2 The Bundle-Delivery Algorithm 64
4.4.3 Performance Analysis 68
4.5 Evaluation 70
4.6 Conclusions and future work 71

5 A User-centric Live Rate Adaptive Streaming CDN System 73
5.1 Introduction . . . 73
5.1.1 Our Contributions . . . 74
5.2 User-centric Discretized Streaming Model . . . 75
5.2.1 User satisfaction . . . 75
5.2.2 Live video streaming in a CDN . . . 77
5.3 Formulation of the Capacity Problem . . . 78
5.3.1 Objective Functions . . . 78
5.3.2 Integer Linear Program Formulation . . . 79
5.4 Proof-of-Concept for User-centric Discretized Streaming . . . 81
5.5 A Practical System: SCADOOSH . . . 83
5.5.1 Type Specified User Assignment . . . 85
5.5.2 Utility-based Content Placement . . . 87
5.5.3 Utility driven delivery trees construction . . . 89
5.6 Evaluation . . . 91
5.6.1 Simulation settings . . . 92
5.6.2 Results . . . 93
5.7 Conclusion . . . 96

6 Conclusion 99
6.1 Synopsis . . . 99
6.2 Limitations and Perspectives . . . 101

Bibliography 114

Publications 115

Glossary 117

List of Figures

2.1 Tree-based P2P overlay 14
2.2 Multitree-based P2P overlay 15
2.3 Formulation of a mesh-based P2P overlay 16
2.4 CDN delivery network 21
2.5 Live stream delivery over CDN: the main actors 22
2.6 Media Presentation Data file 24
2.7 A live MPD file example 25
2.8 Distributing live rate adaptive streams in CDN 26

3.1 Three sources in a multioverlay live video system. 30
3.2 Example of a sub-optimal bandwidth allocation. 30
3.3 Bipartite Flow Network 34
3.4 The two level simulator platform 44
3.5 Simulation Setting 45
3.6 Total amount of missing upload capacity. 47
3.7 Average chunk losses of preferred and non-preferred videos per peer for Preference-based. 47
3.8 Total chunk losses. 48
3.9 Average chunk losses in overlays. Overlays are ordered by popularity. The video bit rate is 330 kbps. 49
3.10 PSNR results. 50
3.11 Number of peers in the system. 52
3.12 Upload bandwidth deficit in the system. 52
3.13 PSNR results. 53
3.14 CNG user communication tool 55

4.1 The CDN graph associated with 3-SAT 63
4.2 T_j generation: $V_j = \{u, v\}$, $f_u^j = 3$, $f_v^j = 2$. 67
4.3 Evaluation for small instances 71
4.4 Evaluation for large instances 71

5.1 MOS models 76
5.2 Relative satisfaction for users who can play up to the fourth representation 77
5.3 Topology of CDN toy-network 82
5.4 CDF of user satisfaction 82

5.5 Received representations on each edge server 83
5.6 The SCADOOSH system 84
5.7 Two types of user-generated video 88
5.8 ARIMA model identification 89
5.9 Evaluation of the prediction 90
5.10 Graph transformation 91
5.11 Variation of global population over the 150 periods 92
5.12 CDF of user satisfaction for three numbers of reflectors 94
5.13 Average satisfaction of users regarding to each edge server family . . . 95
5.14 Average satisfaction of users: `GEN` vs. `SPC` 95
5.15 CDF of user satisfaction: `GEN` vs. `SPC` 96

List of Tables

2.1 Comparison of tree-based and mesh-based P2P streaming protocols . . 17
2.2 Representations in Akamai adaptive video streaming 23

3.1 Notations used in Chapter 3 33
3.2 Computation time for minimum-cost maximum-flow algorithms. 42

4.1 Notations used in Chapter 4 59
4.2 Notations for the Bundle-Delivery algorithm 65

5.1 Representation bit-rates 76
5.2 Technologies and ratio of associated users 93

Résumé

Évolution des technologies de transmission de flux vidéo en direct

Le mot *streaming* correspond à la technique de transfert de contenu vidéo sous forme d'un flux régulier et continu. La transmission de flux vidéo en ligne peut être classifiée en deux types: *vidéo à la demande* (abrégée en VoD de l'anglais Video on Demand) et *flux vidéo en direct* (terme anglais, live streaming). Pour les services de vidéo à la demande, le service est rendu selon la demande des client, donc les contenus vidéos sont consultables à tout moment. Cependant, le service de flux vidéo en direct nécessite l'envoi de contenu vidéo en temps réel (ou en léger différé) à un large public depuis l'Internet. À cause de l'exigence de "temps réel", la transmission de flux vidéo en direct est beaucoup plus difficile que VoD.

Depuis la première émission en direct en ligne réalisée par "Severe Tire Damage" le 24 juin 1993, la transmission de vidéo en ligne est devenue l'un des plus grands et plus populaires domaines dans le monde de l'Internet. Il y a plusieurs types d'applications/platformes existant qui fournissent différents types de contenus à des millions de utilisateurs à chaque seconde. A titre d'exemple, plus de 1,000,000 utilisateurs simultanés ont regardé Jeux Olympiques de Beijing depuis la platforme PPLive [ppl]. Sur *justin.tv*, une plateforme de vidéo en direct sur le Web, une nouvelle vidéo commence chaque seconde, et 300 millions de vidéos sont vues mensuellement [jus]. Ce sont seulement deux exemples parmi beaucoup d'autres.

Pourtant, la distribution des vidéos en direct pour des milliers d'utilisateurs via l'Internet n'est pas facile. Au début, les protocoles *IP multicast* ont été proposés pour réduire la charge du réseau [DC90]. L'idée fondamentale de ces protocoles est d'envoyer un seul flux de données à partir du fournisseur de contenu vers de multiple destinataires qui sont membres d'un groupe multicast. Par exemple, les protocoles SRM [FJL+95] et RMTP [PSLB06] sont des études réalisées dans ce domaine. Cependant, à cause de sa complexité, la multicast IP est peu déployé: il a besoin de modifications sur l'infrastructure, d'implementer des caractéristiques de couche supérieure (tel que le contrôle de congestion, etc.), et de maintenir l'état de chaque groupe à la couche IP. L'alternative proposée par les scientifiques consiste à gérer la fonctionnalité de multicast au niveau de la couche application en construisant des réseaux superposés (réseau overlay) au niveau applicatif. A ce sujet, deux approches principales ont été proposées: les systèmes pair-à-pair (P2P, de l'anglais "peer-to-peer") et les Content Delivery Networks (CDNs).

Pour les deux approches, la première question à régler est le problème de passage à l'échelle, en d'autres mots, la capacité de s'adapter à un changement du nombre d'utilisateurs de plusieurs ordres de grandeur. Les systèmes P2P (voir par exemple [LGL08, YV07, hCRSZ00, ppl]) sont intrinsèquement capable de passage à l'échelle, parce que les utilisateurs partagent leurs ressources avec d'autres utilisateurs, donc plus il y a des participants dans le système, plus il y a de ressources. Chaque pair joue à la fois le rôle de serveur et de client. Cette caractéristique résout le problème du passage à l'échelle et permet à un grand nombre d'utilisateurs de profiter du service. Les utilisateurs organisent un réseau P2P overlay pour la diffusion de la vidéo. Par contre, un Content Delivery Network (CDN) a été élaboré pour s'adapter à une utilisation grande échelle des applications [mKPB06, KSW+04, AMM+11, NSS10]. Un Réseau CDN est basé sur le déploiement de plusieurs serveurs, qui collaborent pour répliquer le contenu (vidéo) et fournir un accès rapide et fiable au service. Les utilisateurs sont redirigés de manière transparente vers le serveur approprié. En fait, P2P et CDN sont deux solutions complémentaires. Les CDNs peuvent fournir des services garantis, mais ils doivent provisionner leur réseau avec suffisamment de capacité, ce qui engendre un coût non négligeable. Alors que les solutions P2P peuvent offrir les services avec une garantie de service qui est seulement probable, mais à moindre coût. Par conséquent, les deux techniques, P2P et CDN, ont des défis et des exigences particulières. Un fournisseur de service de streaming vidéo en direct doit envisager attentivement la bonne approche lors de la phase de conception d'un système.

Au cours des dernières années, la consommation de flux vidéo en ligne a augmenté significativement. La nouvelle génération de technologies d'accès, par exemple 3G et 4G, permet d'accéder aux services de vidéo en ligne depuis n'importe quel endroit et à n'importe quel moment. Une bande passante plus large mène à une plus grande popularité des vidéos en Haute Définition (HD). Les dispositifs avec caméras équipés permettent à chacun de générer du contenu en direct pour partager, et par conséquent, de nouveaux types de services de vidéo en direct émergent. Toutes ces tendances imposent de nouveaux défis aux services de flux vidéo en direct. Dans la suite, nous allons d'abord exposer nos observations sur l'état actuel et l'évolution de streaming vidéo en direct:

La domination du trafic vidéo. Les vidéo occupent déjà la majeure partie du trafic Internet mondial, sur les réseaux fixes et aussi sur les réseaux mobiles [san, cisa, cisb]. En 2012, l'analyse réalisée dans [san] sur le trafic Internet en Amérique du Nord a montré que 65% du trafic réseau fixe et 57% du trafic réseau mobile en sens descendant sont la vidéo. Les mêmes chiffres pour l'Europe étaient de 38% et 39% pour le réseau fixe et le réseau mobile, respectivement. Jusqu'à présent, la croissance de la partie vidéo de l'ensemble du trafic Internet correspond la prediction du rapport Cisco 2010 [cisb] qui prévoit que la vidéo représentera 90% du trafic global en 2015. Actuellement, le service de vidéo en ligne, y compris des flux vidéos en direct, est devenu la principal industrie du loisir en ligne. Par exemple, le Jeux Olympiques de Londres représente entre 8-12% du trafic réseau aux États-Unis au sommet [san].

La démocratisation de producteurs de vidéos en direct. C'est la beauté de l'Internet de changer la production de contenu de la part des quelques magnats des

médias à des millions d'utilisateurs ordinaires. Aujourd'hui, les vidéos en direct en ligne ne sont pas générées uniquement par un petit groupe d'éditeurs bien identifiés qui émettent les événements importants: comme les dernieres nouvelles, politiques et sports, etc. Maintenant, un spectateur peut diffuser un match depuis un stade; un adepte des jeux vidéos peut diffuser le vidéo capturée de son écran quand il joue du jeux vidéo en ligne; ou encore, une ligue sportive de jeunesse locale peut émettre les matchs pour les parents à regarder, etc. Il existe plusiere des plates-formes qui permettent de diffuser les vidéos en direct généré par les utilisateurs. De plus, les applications de réseau social (abrégée en OSN de l'anglais Online Social Network) permettent à ces "sources" de promouvoir leur propre vidéos. Par exemple, ces plates-formes comprennent le plate-forme *justin.tv* [jus], le plate-forme *ustream* [ust], et le plate-forme *Xfire* [xfi]. Ces services attirent une grande population: par exemple, *Xfire*, qui permet aux joueurs du jeu en ligne massivement multijoueur (MMOG, de l'anglais massively multiplayer online game) de capturer des vidéos en direct à partir de leur écran de jeu et de les diffuser à d'autres joueurs, annonce plus de 20 millions de joueurs [SI11].

L'hétérogénéité des consommateurs de vidéos. Aujourd'hui, les gens peuvent regarder des vidéos depuis leur terminaux de nouvelle génération (tels que les smartphones et tablettes) n'importe où et à n'importe quel moment. Typiquement, il est devenu courant pour les utilisateurs mobiles de profiter des divertissements vidéos via des connexions sans fil (3G et WiFi). Un rapport récent de Cisco montre que la vidéo a dépassé 50% du trafic mobile à la fin de l'année 2012 [cisa]. Dans le même temps, le rapport souligne la plus grande diversification des terminaux: les smartphones et les appareils plus récents représentent davantage de trafic qu'avant. Une étude [nbc] a montré que pendant les Jeux Olympiques de Londres, 45% des demandes des vidéos sont réalisées à partir d'appareils mobiles (tablettes et téléphones). Ces appareils ont des tailles d'écran différentes, donc ils acceptent des vidéo avec des résolutions différentes. De même, ils sont connectés à l'Internet depuis des accès différents, ainsi ils peuvent lire des vidéos ayant des débits différents. Par conséquent, ils ont des exigences différentes en streaming. Pour illustrer ces propos, la dernière génération de télévisions connectées est capable d'afficher une vidéo avec des résolution jusqu'à 1080p, alors qu'un utilisateur de smartphone est seulement autorisé à regarder de la vidéo en 360p. L'*Adaptatif Bit Rate Streaming* a été conçu pour servir une population hétérogène d'utilisateurs tout en garantissant une bonne qualité d'expérience (QoE) pour tout le monde. Dans le streaming adaptatif, un flux vidéo a des multiples représentations, chacune correspondant à une certaine qualité de la vidéo avec une certaine résolution, et un certain débit. Cela permet au client de passer de façon dynamique d'une vidéo à l'autre en fonction de la variation de la qualité de sa connection réseau. Aujourd'hui, le streaming adaptatif est de plus en plus utilisé dans les architectures utilisant des CDN. Cette technique est désormais normalisée à MPEG sous le nom de DASH (Dynamic Adaptative Streaming over HTTP) [das, Sto11]. La standard DASH est une nouvelle étape vers une adoption plus large des technologies de streaming adaptatif

Comme nous l'avons présenté, un prestataire de services doit examiner attentivement la bonne approche (soit P2P ou CDN) lors de la conception d'un système de

streaming en direct. Le CDN est le meilleur choix pour le streaming des vidéos adaptatif pour au moins deux raisons. D'abord, le technologie de streaming adaptatif est particulièrement efficace pour exploiter la liaison descendante disponible d'un utilisateur lorsque cet utilisateur a une connexion de réseau stable. C'est plus difficile dans les systèmes P2P où des connexions multiples et transitoires sont utilisées entre les pairs. En second lieu, les serveurs CDN peuvent offrir différentes représentations, donc il est facile de passer d'une représentation à l'autre. Le passage de représentations dans les systèmes P2P est plus difficile, car cela nécessite de réorganiser les overlays pour trouver d'autres pairs ayant le bon contenu. Dans DASH, où les morceaux de vidéos peuvent être de deux secondes seulement, le changement devient très difficile en P2P.

Au contraire, pour les vidéos en direct générés par les utilisateurs, le P2P est un meilleur choix pour les raisons suivantes. Tout d'abord, la popularité des vidéos de ces types constitue un défi majeur: (1) d'une part, un grand nombre de vidéos sont générés simultanément; (2) de l'autre côté, la plupart des vidéos sont regardés par une petite population (*e.g.* amis du diffuseur dans les applications de réseau social). Cette distribution de popularité est plus difficile à gérer pour un CDN, qui préfère faire du streaming avec un petit nombre de contenus à un grand nombre de nœuds. Deuxièmement, la limitation de système P2P pour traiter le streaming adaptatif n'a pas d'importance ici car les contenus générés par les utilisateurs peuvent rarement bénéficier de l'avantage de technologie streaming adaptatif. En effet, les appareils qui capturent et initient le flux vidéo sont limitées en capacité, ainsi le flux vidéo brut est de qualité relativement médiocre et il est impossible de tirer de cette vidéo brute suffisamment de représentations diversifiées. Par ailleurs, codage par un acteur central pour un grand nombre de vidéos provoque un coût de calcul lourd et un large délai supplémentaire.

Motivations et Défis

La motivation globale de cette thèse est de concevoir des systèmes de streaming vidéo en direct qui répondent aux tendances mentionnées ci-dessus, en particulier de vidéo en direct générée par l'utilisateur, et de vidéo en direct de streaming adaptatif. Basés sur l'observation des tendances ci-dessus, nos objectifs sont comme suit:

- Notre premier objectif est de concevoir un système qui permet aux utilisateurs de diffuser leurs propres vidéos en direct en utilisant l'approche P2P.
- Le deuxième objectif est de construire un système de streaming en direct qui offre des vidéos adaptatifs depuis le réseau CDN.

Dans ces deux travaux, nous essayons de résoudre le problème de transmettre des flux vidéo en direct dans une infrastructure de livraison (l'overlay P2P et le réseau de CDN) sous la contrainte de la bande passante des nœuds relais (des pairs et des équipements intermédiaires CDN). Comme nous l'avons dit avant, le trafic vidéo occupe la première place du traffic internet, et il est prévu que ce trafic continue d'augmenter. En conséquence, un défi principal est la disponibilité de la bande passante dans l'infrastructure de distribution pour les deux systèmes. Donc, notre tâche

est de trouver les solutions efficaces en utilisant la bande passante des équipements et de concevoir des politiques de diffusion pour les systèmes “sous-provisionnés”. Dans ce qui suit, nous discutons en détail les défis rencontrés par chaque système.

Contrainte de bande passante pour le système P2P de partager les vidéos en direct générés par les utilisateurs

Il y a de multiples overlay P2P dans les systèmes de partage de vidéos en direct générés par les utilisateurs. Pour chaque vidéo, un overlay P2P est formé par le diffuseur et les utilisateurs (pairs) qui regardent cette vidéo en direct. Le diffuseur émet sa propre vidéo en direct, les pairs participent à la diffusion de la vidéo à d'autres pairs dans la même overlay P2P.

Typiquement, dans ce service, les utilisateurs peuvent regarder plusieurs vidéos en direct simultanément. Cette fonctionnalité est très appréciée par les utilisateurs. Quelques scénarios d'exemples comprennent: (*i*) la TV avec multi-canal, où un utilisateur peut regarder plusieurs chaînes simultanément [WXR11]; (*ii*) le streaming multi-caméra, où un client peut regarder le même événement à partir de points de vue différents [NBC12]; (*iii*) la coopération des joueurs MMOG, où les joueurs d'une même équipe peuvent transmettre la vidéo de leur jeu à l'autre [ABH$^+$11]. Cependant, cette caractéristique requiert une grande disponibilité de bande passante. Dans le streaming P2P, il est largement accepté que le goulot d'étranglement des ressources provient de la bande passante montant aux pairs. La connection d'un pair à l'Internet est limité, et ce lien doit être partagé entre tous les overlays pour lesquelles le pair participe. Par conséquent, un défi majeur est le problème d'allocation de bande passante, qui consiste à trouver une allocation de bande passante (montant) de chaque pair à ses vidéos regardés (des P2P overlays).

Parfois, le système peut être sous-provisionné, c'est-à-dire que la bande passante de l'ensemble des pairs n'est pas suffisante pour soutenir la diffusion de toutes les vidéos. En dépit de l'énorme volume de recherche sur streaming P2P, le déficit des ressources a été constamment ignoré. Quand les scientifiques sont confrontés à la réalité du déficit de ressources [SIB12, WLZ11a], ils ont proposé de déployer des serveurs supplémentaires comme des assistants de bande passante pour empêcher le système de devenir sous-provisionné. Cependant, cette solution n'est pas applicable à un scénario avec beaucoup d'overlays de petite taille, car il faudrait réserver et gérer un grand nombre de machines virtuelles, avec chacune générant une petite quantité de trafic. Ainsi, un autre défi consiste à définir des stratégies d'allocation de bande passante pour le système sous-provisionné, par exemple, comment partager le déficit de la bande passante entre les overlays.

Contrainte de bande passante pour diffuser de vidéo adaptatif en direct sur CDN

Le standard récent DASH résout le problème de diffuser des vidéos pour un vaste ensemble d'appareils hétérogènes, cependant, il est très exigeant pour l'infrastructure de CDN: le chaîne vidéo est en effet un ensemble des flux plus petits, chacun correspondant à une représentation différente de la vidéo. Par exemple, les vidéos HD de

Netflix sont encodés à 14 représentations avec un taux cumulé de plus de 20 Mbps par chaîne [AGH+12]. La conséquence est que les CDNs rencontrent un énorme problème de passage à l'échelle. Par exemple, le leader mondial de fournisseur CDN, Akamai, a récemment annoncé que son infrastructure "*doit être augmentée par un facteur de 100 fois dans les cinq années prochaines pour suivre à la demande de diffuser des vidéos en temps-réel*" [Ing12]. Le problème de passage à l'échelle est plus difficile pour les systèmes de streaming en direct que pour la VOD, pour lequel il est possible de retarder les transferts aux heures creuses [LSYR11]. Streaming en direct exige que le fournisseur de CDN dispose d'une infrastructure de distribution solide, par exemple, pour s'assurer que les équipements de l'infrastructure CDN sont capables de transmettre des flux à partir de serveurs d'origine aux serveurs edge, puis aux utilisateurs. Des travaux réalisés précédents dans ce domaine [ASV11, AMM+11, ZAB+12] ont mis en évidence que la bande passante montante des équipements intermédiaires est la ressource la plus critique à gérer.

Par conséquent, le défi principal pour le CDN d'aujourd'hui est de concevoir des solutions efficaces en termes d'utilisation de la bande passante pour le système CDN malgré une infrastructure sous-provisionnée. Dans des infrastructures bien gérées, comme les CDN, les arbres sont la solution la plus efficace et robuste pour transmettre des données [ASV11, AMMS03]. Plus précisément, le problème est de construire un ensemble d'arbres dans le réseau de CDN pour maximiser le nombre de flux délivrés, sous les contraintes de capacité des équipements intermédiaires et la topologie de CDN.

Le streaming adaptatif est inventé pour maximiser la QoE des utilisateurs avec des appareils divers et des liens d'accès différents. Un rapport récent sur l'expérience de l'utilisateur à regarder des vidéos en ligne révèle que les utilisateurs ont regardé des vidéos avec mauvaise qualité: "*60% de tous les vidéos ont de la qualité dégradée*" [Con]. L'état est encore pire pour les vidéos en direct: le pourcentage de vidéos avec qualité haute (HQ de l'anglais High Quality) diminue de 60% au début de 2012 à 35% à la fin pour les vidéos en direct en ligne. Cette observation illustre un autre défi qui consiste à réaliser une solution orientée sur la satisfaction des utilisateurs.

Contributions

Selon les discussions ci-dessus sur les objectifs et les défis, nous avons concentré nos efforts sur la gestion des ressources en bande passante limitées, pour le streaming vidéo en direct pour les systèmes P2P et les systèmes CDN. Nous listons les contributions de cette thèse comme suit.

Un système pair-à-pair multi-overlay pour partager les vidéos en direct

Le travail dans cette partie est un élément prépondérant du projet CNG (de l'anglais "Community Network Game"), un projet financé par la Commission Européenne FP7 STREP. L'ambition du projet CNG est de permettre aux utilisateurs MMOG de délivrer leur propre contenu (vidéo par exemple) aux autres joueurs en utilisant le technologie P2P.

L'étude réalisée dans ce travail est un système P2P multi-overlay de diffusion de vidéos en direct. Nous nous concentrons sur le problème de l'allocation de bande passante, c'est à dire les utilisateurs qui regardent plusieurs vidéos doivent décider comment partager leur bande passante de liaison montante parmi les overlays. Typiquement, nous faisons attention à l'approvisionnement des overlays. Un overlay est sur-provisionné, si la bande passante qui lui est réservé est supérieure à sa demande de diffusion, autrement, il est sous-provisionné. Donc, la bande passante est dite *gaspillée* si elle a été attribuée aux overlays sur-provisionnés, alors qu'elle aurait pu être attribuée à ceux sous-provisionnés. Notre première contribution est la formulation du problème d'allocation de bande passante optimale avec l'objectif de minimiser le gaspillage de bande passante. Nous le résoudrons par le problème de flot maximale dans un graphe biparti. Nous montrons donc qu'une solution optimale peut être trouvée en temps polynomial du nombre d'utilisateurs dans le système.

Notre deuxième contribution est la conception de plusieurs stratégies d'allocation de bande passante, dont l'objectif est de partager le déficit entre les overlays pour un système sous-provisionné. Nous avons proposé plusieurs politiques (par exemple, en donnant la priorité à la diversité des vidéos, aux vidéos les plus populaires, aux vidéos préférées et aux utilisateurs primes), et plus nous avons montré qu'il est possible de trouver la solution optimale depuis un problème de flot maximale avec coût minimum dans le même graphe biparti. Comme ca, les solutions peuvent être trouvées avec des algorithmes en temps quasi-polynomial. En plus, nous avons aussi proposé une stratégie équitable (*fairness*) qui alloue la bande passante basée sur les demandes des overlays. Un algorithme distribué est conçu pour cette stratégie. Enfin, nous avons également proposé des mécanismes concernant la praticabilité du système. Les performances de notre proposition ont été validées par une simulation dynamique à grande échelle basée sur des trace réelle.

Nos contributions cherchent à résoure des problèmes ouverts que les travaux précédents [WLL08, WXR09, WXR11] n'ont pas réglés. Tout d'abord, les travaux précédents sont aveugle au provisionnement de l'overlay. Ensuite, ils obtiennent les résultats biaisés telles que certains overlays peuvent souffrir de déficit de ressource, même si le système est bien provisionné [WXR09, WXR11]. Deuxièmement, nous avons défini des politiques pour un système sous-provisionné. Dans des travaux précédents, le sous-provisionnement du système est ignoré [WXR09, WXR11] ou totalement inacceptable [WLL08]. Enfin, nous avons comparé notre étude à celle la plus proche [WXR09, WXR11], l'évaluation montrant les avantages de notre approche.

Modèle de streaming discrétisé pour diffuser les vidéos adaptatifs en direct sur CDN

Le deuxième contribution de cette thèse est la formulation d'un nouveau modèle, *le modèle de streaming discrétisé*, pour distribuer les vidéos adaptatifs dans le réseau CDN. Cette contribution comble une lacune importante dans la littérature scientifique liée à CDN et streaming du vidéo en direct.

En effet, le streaming du vidéo en direct a reçu trop peu d'attention dans la littérature. Une étude bibliographique [Pas12] a référencé 438 documents liés à la diffusion de contenu depuis P2P, les réseaux assistés par des pairs et CDN; mais, seulement deux

citations pour le streaming du vidéo en direct dans les CDNs [AMMS03, AEVW04]. Les travaux plus récents [AMM+11, NSS10, ASV11] consacrés au streamin vidéo en direct dans les CDNs sont sur un sujet qui, selon nous, est moins important aujourd'hui. Leur objectif est de réduire le coût de la transmission sur les liens dit de peering entre les utilisateurs et les nœuds CDN. Cependant, les CDNs modernes fonctionnent avec des serveurs qui se trouvent au sein du réseau des fournisseur d'accès à Internet (FAI, le terme en anglais est Internet Service Provider (ISP)), et sur des accords de peering avec ces FAIs [net]. En fait, le coût pour faire transiter du trafic entre les réseaux différents a diminué significativement [Kro11], à un point où il n'est plus la considération principale. Au contraire, nous croyons que la préoccupation majeure pour les fournisseurs de CDN est la disponibilité de la bande passante par suite de l'augmentation rapide du trafic. En outre, aucun des travaux précédents cible en streaming adaptatif, tandis que le streaming adaptatif est une grave menace pour la capacité limitée dans les réseaux d'infrastructure de CDN.

Dans le même temps, une série de travaux ont étudié la capacité de streaming de réseaux [SLC+11, ZLW11, NL11, KS11]. L'objectif est de déterminer le débit maximum qui peut être livré à tous les nœuds d'un réseau. Ces travaux sont basés sur la vidéo avec débit élastique et l'hypothèse que les flux de données sont divisibles infiniment. Par contre, dans le contexte de streaming adaptatif, les débits des différentes représentations du vidéo sont bien prédéfinis. Et plus, chaque flux doit être soit livré entièrement, soit ne pas être livré du tout. Le débit du réseau doit maximiser le nombre de flux délivrés, pas le débit livrable. Ainsi, nous avons conçu un nouveau modèle, qui maximise le nombre de flux délivrés d'un réseau.

Pour ce travail, nous avons les contributions suivantes. D'abord, nous donnons une formulation du problème général par l'optimisation linéaire (OL) en nombres entiers (terme en anglais Integer Linear Programming (ILP)) et nous prouvons que le problème est NP-complet. Le problème général est formulé comme la maximisation de l'utilité des flux livrés par un ensemble d'arbres. La complexité NP-complétude implique qu'il est actuellement impossible de trouver une solution rapide et optimale pour le cas général. Ainsi, nous nous concentrons sur un scénario pratique, qui correspond à la mise en ouvre de CDN d'aujourd'hui en flux vidéo. Nous présentons un algorithme, qui est rapide, facile à mettre en ouvre, et quasi-optimale. Nous montrons théoriquement le facteur d'approximatio de l'algorithme, qui est négligeable pour la configuration considérée. Enfin, nous évaluons le facteur d'approximatio de l'algorithme par une simulation numérique.

Un système concentré sur les utilisateurs en streaming adaptatif en direct

La troisième partie de ma thèse explore davantage le modèle de streaming discrétisé. Dans ce travail, nous spécialisons le modèle discrétisé général à un modèle centré sur les utilisateurs. Comme l'objectif d'un CDN est de satisfaire les utilisateurs, nous proposons un modèle qui maximise la satisfaction (mesurée en la Qualité d'Expérience (QoE)) des utilisateurs sur les flux livrés. Ainsi, ce modèle est nommé *modèle de streaming discrétisé concentré sur les utilisateurs*. La première contribution est un fondement théorique pour ce travail. D'abord, nous définissons un modèle qui per-

met au fournisseur de CDN d'estimer objectivement la satisfaction de utilisateur sur chaque représentation dans le contexte de streaming adaptatif. Ce modèle permet la définition de l'utilité de flux à base de QoE de utilisateur pour le modèle streaming discrétisé. Nous présentons plusieurs objectives possibles et nous nous concentrons sur l'objectif de garantir la *max-min fairness* de la satisfaction des utilisateurs, et en même temps de maximiser la satisfaction totale des utilisateurs. Nous formulons un modèle OL, qui conjointement décide: (1) la représentation qui doit être envoyée aux serveurs edge; (2) un ensemble d'arbres enracinés au serveur d'origine, et avec des serveurs edge qui sont des feuilles; et (3) l'association des utilisateurs et serveurs edge. Ces trois points correspondent aux trois mécanismes principaux de CDN pour la diffusion du vidéo en direct: (1) le placement de contenu, (2) la diffusion de contenu, et (3) l'association des utilisateurs. L'évaluation du modèle OL sur un ensemble de mini-infrastructures CDN démontre les avantages de notre modèle, notamment en comparaison des approches précédents [SLC+11, ZLW11, NL11, KS11].

Ensuite, la deuxième contribution justifie le modèle discrétisé concentré sur les utilisateurs dans un système pratique. En revisitant les trois mécanismes principaux de CDN, le système est capable d'utiliser efficacement l'infrastructure CDN pour diffuser les vidéos adaptatifs en direct aux utilisateurs à grande échelle dans un environnement dynamique. Le système contient trois composantes. Chaque composante traite de l'un des trois mécanismes CDN principaux susmentionnés:

- Un composant d'association d'utilisateur assemble les utilisateurs avec des exigences similaires.
- Un composant de placement de contenu décide à où placer les représentations en calculant l'utilité des représentations pour chaque serveur edge.
- Un composant de diffusion de contenu crée un overlay des arbres qui exploite la capacité de l'infrastructure CDN à l'égard de l'utilité de contenu.

La performance du système est validée par des simulations basées sur traces réels à grande échelle. Les résultats montrent que le système pourrait atteindre en moyenne une satisfaction haute avec peu de coût d'infrastructure CDN dans un environnement dynamique.

Chapter 1

Introduction

In this chapter, we first provide a brief discussion on the current status and trend of the online live video streaming services. This discussion reveals the main challenges of live video streaming for various application scenarios and types of platforms, and leads to the motivation of this thesis. Then, the main contributions of the thesis are introduced, followed by a brief outline of the organization of this dissertation at the end of this chapter.

1.1 Status of Online Live Video Streaming

Online video streaming services can be roughly classified into two types: *Video on Demand* (VOD) services and *live* video services. As the name suggests, VOD services allow users to re-access the same video content multiple times at any moment while live video services broadcast real-time video streams provided by the *content providers* to an audience over the Internet. The "real-time" demand makes live video streaming far more challenging than VOD services.

Since the first online live show performed by the "Severe Tire Damage" band on June 24 1993, live video streaming has become one of the biggest and most popular realms in the Internet world. Multiple successful applications/platforms provide live video service to millions of people every second. For example, PPLive [ppl] served over 1,000,000 simultaneous users for Beijing Olympics games. On justin.tv, a web-based live video platform, one new live video starts every second, and users watch more than 300 million videos every month [jus]. These are two examples among many others.

Yet, delivering live streams to thousands of users on the Internet is not trivial. At the beginning, native *IP multicast* was proposed [DC90]; for example SRM [FJL+95] and RMTP [PSLB06]. However, IP multicast is sparsely deployed due to its complexity: it requires modifying the infrastructure, implementing complex high layer features (such as congestion control and flow control) and maintaining per group state at the IP layer. As an alternative, scientists have proposed to lift the multicast functionality to the application layer by building application-level infrastructure overlays. To this end, two main classes of approaches have been proposed: peer-to-peer (P2P) systems and Content Delivery Networks (CDNs).

For both approaches, *scalability* is the most important issue to address. Scalability is the ability of the system to deliver a video stream to an audience set as large as possible. P2P approaches (see for instance [LGL08, YV07, hCRSZ00, ppl]) are intrinsically scalable because each user contributes to the delivery with its own resources, so the more users, the more resources. Peers organizes into P2P *overlays* to assist video stream delivery. On their side, CDN providers [mKPB06, KSW+04, AMM+11, NSS10] are able to scale by leveraging a large set of machines (often called *surrogate* servers or *edge* servers), which are deployed at carefully selected locations in the Internet. Users are transparently re-directed to the appropriate server. P2P and CDN are two complementary solutions. CDNs can provide guaranteed services, however this comes from provisioning the CDN with sufficient capacity at a non-negligible cost. Whereas P2P solutions can provide services with "statistical" guarantees at a lower cost. Both P2P and CDN have special design challenges and requirements, consequently, a service provider must carefully consider the right approach when designing a live video streaming system.

During the last couple of years, the appetite for video stream consumption has significantly grown. New generation of access technique, such as 3G and 4G, make online video services ubiquitously available. Larger broadband bandwidth leads to the popularity of High Definition (HD) videos. Camera-enabled devices allow everybody to generate live content for sharing, and new types of live video services arise consequently. All these trends impose new challenges to live streaming services. In the following, we will first state our observations on the current status and trend of online live video streaming:

The domination of video traffic. Video traffic is already the major portion of global Internet traffic, on both fixed and mobile networks [san, cisa, cisb]. In 2012, measurement of Internet traffic in north America showed that during peaks, 65% of the fixed network and 57% of the mobile network downstream traffic are video [san]. The same figures for Europe were 38% and 39% for fixed and mobile network respectively. So far the growth of the video part of the overall Internet traffic matches the expectations of the 2010 Cisco report [cisb], which predicts that video will represent 90% of the overall traffic in 2015. Currently, online video service, including live videos, has become the main vector of online entertainment. For example, the London Olympic Games accounts for between 8-12% of network traffic in the US at its peak levels [san].

The democratization of live videos producers. It is the beauty of the Internet to shift the content production from a handful of media magnates to millions of end-users. Today's online live videos are not generated only by a small set of well-identified publishers who report global events, such as hot sports and breaking news. Now a fan can stream herself while watching a game at a stadium, a gamer can stream the screen-captured video of her game play, a local junior sport league can broadcast matches for relatives to watch, *etc.* Multiple platforms allow ordinary Internet users to broadcast their own user-generated live videos. Moreover, popular Online Social Network (OSN) applications enable these "sources" to promote their user-generated live videos. Such platforms include the *justin.tv* [jus] platform, the

ustream [ust] platform, and the *Xfire* [xfi] platform. These services are attracting a large population: for example, *Xfire*, which allows players of Massively Multiplayer Online Games (MMOGs) to capture live videos from their game screen and broadcast them to other players, are reported as hosting over 20 million gamers in [SI11].

The heterogeneity of video consumers. Last generation devices (such as smartphones and tablets) allow users to consume videos anytime and anywhere. Typically, it has become commonplace for mobile users to enjoy online video entertainment through wireless connections (3G and WiFi). A recent Cisco report shows that video exceeded 50 percent of mobile traffic by the end of 2012 [cisa]. At the same time, the report emphasizes an increasing device diversification: smartphones and newer device categories such as tablets are accounting for a more significant portion of the traffic. Real statistics collected during London Olympic Games shows that 45% of video requests are released from mobile devices (tablets and phones) [nbc]. These devices have different screen size, so they tolerate different video resolution. Similarly, they are connected to the Internet through different types of access link, so they can accomodate different video bit-rates. Consequently, they have different streaming requirements. For example, a wired home last-generation TV is capable to watch a video with resolution up to High-Definition, whereas a smartphone user is only allowed to watch low quality video. *Rate adaptive streaming* technique has been designed to serve a more heterogeneous population of users with a good Quality of Experience (QoE). In rate adaptive streaming, a video stream has multiple representations, each corresponding to a certain video quality with certain resolution and bit-rate. This allows the client to adaptively switch from one video quality to another according to the quality of the network link. The recent Dynamic Adaptive Streaming over HTTP (DASH) standard [das, Sto11] is a new step toward a broader adoption of rate-adaptive streaming technologies.

As we stated earlier, a service provider must carefully consider the right approach (either P2P or CDN) when designing a live streaming system. Clearly, CDN is the best choice for streaming rate adaptive live videos. At least two reasons explain it. First rate-adaptive technologies is especially efficient to exploit the available downlink capacity of an user when this user has a stable network connection. In P2P systems, the multiple and transient connections make rate-adaptive technologies less efficient. Second CDN servers can offer different representations, and it is easy to switch from one representations to another. Switching from representations in P2P systems is harder because it requires a re-organization in the overlay in order to find other peers having the right content. At the time scale of DASH, where chunks can be only two seconds, switching becomes challenging.

On the contrary, for streaming user-generated live videos, P2P is a better choice for the following reasons. First of all, the popularity distribution of videos in such services poses a major challenge: (1) a large number of videos are generated simultaneously, (2) most streams are watched by a small population (*e.g.* friends of the video broadcaster in OSN applications). Such popularity distribution is harder to manage for a CDN, which prefers streaming the same content to a large set of surrogates. Second, the limitations of P2P system to deal with rate-adaptive streaming does not matter here because user-generated contents can rarely use rate-adaptive technologies. Indeed,

the devices that capture and initiates the video streams have a limited capacity, so the raw video stream is of relatively poor quality and it is impossible to derive from this raw video a diverse enough set of representations. Moreover performing the transcoding task by some centralized actor for a large number of live videos imposes heavy computation cost and long transcoding delay.

1.2 Motivations and Challenges

The overall motivation of this thesis is to design live video streaming systems that meet the aforementioned trends, in particular both user-generated live videos, and rate-adaptive live videos. Based on the observation of above trends, our objectives are as follows.

- Our first objective is to design a system that allows users to broadcast their own generated live videos by using the P2P approach.
- The second objective of the thesis is to build a live streaming system that provides rate adaptive videos over the CDN network.

In both works, we try to solve the problem of transmitting live streams in the delivery infrastructure (P2P overlay and CDN delivery network) under the bandwidth constraint of the relay nodes (peers and CDN intermediate equipments). As we stated in Section 1.1, the amount of video traffic overwhelm the Internet, and it is expected to still increase. As a result, a major challenge is the bandwidth availability in the delivery infrastructure for both systems. Our task is to find bandwidth-efficient solutions and design delivery policies for underprovisioned system. In the following, we discuss in detail the bandwidth challenge faced by each individual system.

1.2.1 Bandwidth constraint for P2P user-generated live video sharing system

User-generated live video sharing system consists of multiple P2P video streaming overlays. For each video, a P2P overlay is formed by the video broadcaster and users (peers) watching this live video. The video broadcaster emits a user-generated live video, while peers participate in diffusing the video to other peers in the same overlay.

Typically, in such services, a user can watch several live videos simultaneously. This feature is well appreciated by users. Example scenarios include: (*i*) multi-channel TV, where a user can watch several channels simultaneously [WXR11], (*ii*) multi-camera streaming, where a client can watch the same event from different views [NBC12], and (*iii*) MMOG player cooperation, where players from the same team can stream the video of their game to each other [ABH+11]. However, this feature threatens the availability of bandwidth resources. In P2P streaming, it is now widely accepted that the bandwidth resource bottleneck comes from the upload bandwidth at the peers. The connection of a peer to the Internet is limited, and this connection has to be shared among all the overlays to which this peer participates. Consequently, a major challenge is the bandwidth allocation problem, which is to find an upload bandwidth allocation of each peer to its watching videos (P2P overlays).

In the context of extensive video sharing, the system can be underprovisioned. That is, the aggregate upload bandwidth of all peers is not sufficient to support the diffusion of all videos. Despite the vast volume of research on the topic of P2P streaming, resource deficit in P2P has been constantly ignored. At some point scientists did face the reality of resource deficit [SIB12, WLZ11a], they proposed to deploy additional servers as bandwidth helpers to prevent the system from becoming underprovisioned. This solution however does not accommodate well a scenario with many small-size overlays, as it would require reserving and managing a large number of Virtual Machines, each generating a small amount of traffic. Thus, another challenge is to define bandwidth allocation strategies for underprovisioned system, for example, how to share the bandwidth deficit among the overlays.

1.2.2 Bandwidth constraint for live rate adaptive streaming over CDN

The recent DASH standard addresses the problem of streaming videos to a vast set of heterogenous devices, however, it is extremely demanding in the core infrastructure: the high-level channel stream is indeed a collection of smaller video streams, each corresponding to a different representation of the video. For example, the Netflix HD videos are encoded into up to 14 representations with accumulated rate over 20 Mbps per channel stream [AGH+12].

The consequence is that CDNs meet a huge scalability issue. For instance, the worldwide leader of CDN provider, namely Akamai, recently announced that its infrastructure will "*have to expand by a factor of* 100 *times in the next five years just to keep up with the demand for real-time video*" [Ing12]. The problem of scalability is more challenging for live streaming systems than for VOD, for which it is possible to delay bulk transfers at off-peak hours [LSYR11]. Live streaming requires instead the CDN provider to provision a delivery infrastructure in advance, i.e. to make sure that equipments in the CDN infrastructure are able to transmit streams from *origin servers* to *edge servers*, and then to the end-users. Previous works in the area [ASV11, AMM+11, ZAB+12] have highlighted that the upload bandwidth of intermediate equipments is the most critical resource to provision.

Consequently, the main challenge for today's CDN is, again, to design bandwidth efficient video delivering scheme to deal with the underprovisioned CDN delivery infrastructure. In managed infrastructures, such as CDNs, tree-based overlays have proven to be efficient and robust delivery mechanisms [ASV11, AMMS03]. Specifically, the problem is to build a set of delivery trees in the CDN core network to maximize the number of delivered streams, subject to the upload capacity constraints of the CDN intermediate equipments and the topology of the CDN.

Rate adaptive streaming is designed to maximize QoE of users with diverse devices and access links. A recent report about user experience on playing online videos reveals that users are experiencing poor video quality: "*60% of all streams experienced quality degradation*" [Con]. The status is even worse for live videos, which suffers the poorest video quality among live video, short VOD and long VOD: the percentage of High Quality (HQ) videos decreases from 60% at the beginning of 2012 to 35% at the end for online live videos. This observation illustrates another challenge as designing

smart user satisfaction oriented video delivery scheme.

1.3 Contributions

According to the above discussions on objectives and challenges, our efforts have focused on the management of scarce bandwidth resources in live video streaming for both P2P-based and CDN-based systems. We list the contributions of this thesis as follows.

1.3.1 A Multioverlay Peer-to-Peer Live Video Sharing System

This work is involved in the CNG (Community Network Game) project [CNG], funded by the European Commission's Seventh Framework Programme. The CNG project focuses on diffusing User Generated Content (UGC) (videos), from one user to many other users by using the P2P streaming technology in MMOGs.

In this work, we designed a multioverlay P2P live video streaming system. We focus on the bandwidth allocation problem, *i.e.* users who watch multiple video must decide how to share their uplink bandwidth among the concurrent overlays. Typically, we pay attention to the provisioning of overlays. An overlay is overprovisioned if the bandwidth reserved to it is higher than its streaming demand, and underprovisioned otherwise. Thus, bandwidth is said to be *wasted* if it was allocated to overprovisioned overlays, although it could be allocated to underprovisioned ones. Our first contribution is the formulation of the optimum bandwidth allocation problem where the objective is to minimize the waste of bandwidth. We solve it through a maximum-flow problem in a bipartite flow network. Therefore, an optimal solution can be found in a time that is polynomial of the number of users in the system.

Our second contribution is the design of several bandwidth allocation strategies, where the objective is to share the bandwidth deficit among overlays for globally underprovisioned system. We proposed several policies (*e.g.*, prioritizing video diversity, popular videos, preferred videos, and premium users) and we showed that it is possible to find the optimal solution through a minimum-cost maximum-flow problem in the same bipartite flow network. Solutions can be found with existing near-polynomial algorithms. A fairness-based strategy that allocates bandwidth based on the overlay streaming demands are solved by the dual decomposition method. This method leads to a distributed algorithm. At last, we also proposed mechanisms for the practical relevance of the system. The performances of our proposal have been validated by a large-scale real-trace based dynamic simulation.

Our contributions cover the gaps of the previous work [WLL08, WXR09, WXR11] in the following aspects. Firstly, previous work are oblivious to overlay provisioning. Thus, they lead to biased results such that some overlays may suffer from bandwidth deficit even when the system is overprovisioned [WXR09, WXR11]. Secondly, we defined policies for underprovisioned system. In previous work, system underprovisioning is either ignored [WXR09, WXR11] or totally not acceptable [WLL08]. At last, we compared our work to the most relevant one [WXR09, WXR11], the evaluation shows the advantages of our approaches.

1.3.2 Discretized streaming model for delivering live rate adaptive videos over CDN

Our second contribution is the formulation of a new model, namely *discretized streaming model*, for the delivery of rate-adaptive live streams in CDN core network. This contribution fills a critical gap in the scientific literature related to CDN and live streaming.

Indeed, too little attention has been paid to live streaming in CDNs. A recent survey [Pas12] has referenced 438 significant papers related to content delivery through P2P, peer-assisted networking and CDNs; however, only two citations referenced live streaming in CDNs [AMMS03, AEVW04]. Most recent works related to live streaming in CDNs [AMM+11, NSS10, ASV11] have dealt on a topic, which is, in our opinion, less important today. Their goal is reduce the transmission cost of video delivery on peering links between users and surrogates. However, modern CDNs rely on edge servers that are located within the network of Internet Service Providers (ISPs), and on peering agreements with these ISPs [net]. As a matter of fact, the bandwidth cost to make the traffic transit across different networks has significantly decreased [Kro11], to a point that it is no longer the main issue. On the contrary, we believe that the major concern for current CDN providers is the bandwidth availability threatened by the rapidly increasing traffic volume. Moreover, none of the previous work target rate-adaptive streaming, while rate-adaptive streaming is a severe threat for the capacity of CDN core networks.

In the meantime, a series of works have dealt with the streaming capacity of networks [SLC+11, ZLW11, NL11, KS11]. The goal is to determine the maximum bit-rate that can be delivered to all nodes. These work are based on elastic video bit-rate and assume infinitely divisible data streams. However, in the context of rate-adaptive live streaming, the video bit-rate of representations are pre-defined. Each stream has to be either delivered in its entirety, or not delivered at all. The throughput of the network is maximized by the number of delivered streams, rather than the maximum deliverable bitrate. Thus, we have designed a new model, which aims to maximize the number of delivered streams in a network.

For this work, we have the following contributions. We first give a formal formulation of the general problem by *Integer Linear Programming* (ILP) and prove that it is NP-complete. The general problem is formulated as maximizing the utility of delivered streams by a set of delivery trees. The NP-completeness claim implies that it is currently impossible to implement an optimal solution for the general case. Thus, we focus on a practical scenario, which corresponds to today's CDN implementation of live streams. We present an algorithm, which is fast, easy to implement, and near optimal. We provide formal theoretical approximation bounds, which are shown to be negligible for the regarded configuration. At last, we evaluate the approximation ratio of the algorithm by a numerical simulation.

1.3.3 A user-centric live rate adaptive streaming system

The third part of my thesis further explores the discretized live streaming model. In this work, we specialize the general discretized streaming model into a user-centric

one. As the CDN capacity is dedicated to finally earn user satisfaction, we propose a user-centric discretized streaming model that maximizes the user QoE based utility of delivered streams.

The first contribution can be regarded as a theoretical foundation for this work. We first define a model that enables the CDN provider to objectively estimate user satisfaction on each representation in the context of rate adaptive streaming. This model allows the definition of user QoE based stream utility for the discretized streaming model. We present several possible objective functions and focus on the joint objective that guarantees *max-min fairness* on user satisfaction and at the same time maximizes the overall user satisfaction. We formulate an ILP model, which jointly decides: (1) the representation that should be sent to the edge servers, (2) a set of delivering trees from origin server to edge servers and (3) the assignment of end-users to edge-servers. These three points correspond to the three main mechanisms of CDN for live streaming: (1) content placement, (2) content delivery and (3) user assignment. The evaluation of the ILP model on a set of toy-CDN infrastructures demonstrates the benefits of the discretized streaming model comparing to previous approaches [SLC+11, ZLW11, NL11, KS11].

Then, the second contribution substantiates the user-centric discretized streaming model into a practical system. By revisiting the three main CDN mechanisms, the system is able to efficiently utilize the CDN infrastructure to deliver live rate-adaptive video streams to viewers in dynamic and large-scale CDNs. The system contains three components. Each component targets one of the three aforementioned CDN main mechanisms:

- A user assignment component assembles users with similar demands.
- A content placement component decides where to place the representations by computing the utility of representations for each edge server.
- A content delivery component builds a multi-tree overlay that exploits the CDN infrastructure capacity with regard to the content utility.

The performance of the system is validated by large-scale real-trace based simulations. The results show that the system could achieve high user satisfaction with limited CDN infrastructure in a dynamic environment.

1.4 Organization of Dissertation

The rest of this dissertation are organized as follows.

Chapter 2 surveys related work of live streaming for both P2P and CDN systems. We aim to give some background to help readers to understand the technical part of the thesis. We identify several P2P video streaming techniques and we discuss the choice of the techniques that we further use in the multioverlay P2P video sharing system. We also describe current systems for the delivery of live video streams over CDN. At last, we introduce rate-adaptive mechanisms and the transmission of live rate-adaptive streams in CDN.

Chapter 3 is about the multioverlay P2P live video sharing system. We first propose the system architecture and the corresponding model. Then, we formally formulate the bandwidth allocation problem. We first show that the resource allocation with minimum resource waste is equal to the maximum flow in a bipartite flow network model. We further propose several bandwidth allocation strategies for globally underprovisioned system to share the bandwidth deficit among overlays. Then, we provide implementation details to show the practical relevance of the system. Finally, the system performance is validated through a set of large-scale real-trace based simulations. The results shows that resource waste is minimized. Especially, peers could achieve better video quality for both overprovisioned and underprovisioned systems comparing to the main previous work in this topic.

Chapter 4 is about the formulation of the general discretized streaming model for delivering live rate-adaptive streams in CDN. The optimization problem is to maximize the average delivered stream utilities. We provide an ILP formulation and prove that the problem is NP-complete. Then, we focus on a practical scenario which makes sense to CDN providers. For this specific case, we present a fast near-optimum algorithm. The algorithm is analyzed through both theoretical analysis and numerical simulation. The simulation result shows that the approximation ratio is negligible for reasonable CDN configurations.

Chapter 5 proposes the user-centric discretized streaming model for delivering live rate adaptive streams in CDN. We first specialize the general discretized streaming model into a user-centric one by defining user satisfaction based utilities. The objective of the optimization problem is to maximize the overall satisfaction for a user population, while at the same time guarantees max-min fairness on user satisfaction. We formulate an ILP model and demonstrate the benefits of the discretized streaming in a toy-CDN simulation comparing to traditional approaches. Then, we present the practical implementation of the model: a system which delivers live rate adaptive streams in large-scale and dynamic CDNs. To prove the performance of the system, a set of large-scale real-trace based simulations are conducted. The simulation results show that the system could maintain high user satisfaction with limited CDN infrastructure cost.

Chapter 6 summarizes the whole work and gives possible directions on future work.

Chapter 2

Online Live Video Streaming

This chapter presents the state-of-the-art of online live video streaming. Firstly, we discuss in Section 2.1 the requirements and challenges that are specific to online live video streaming. Then, we introduce in detail the two common technologies that are currently used to provide live streaming services: P2P and CDN.

- P2P-based live streaming techniques are discussed in Section 2.2. We first present a taxonomy of solutions in terms of P2P overlay structure. Then, we relate a discussion to our multioverlay P2P live video sharing system.
- Then, we discuss CDN-based live streaming in Section 2.3. After an overall perspective, we introduce in detail one specific rate adaptive streaming protocol—the DASH protocol. Then, we take the DASH protocol as an example to illustrate the delivery of rate adaptive live streams in the CDN infrastructure.

2.1 Live video streaming

The current online video services can be roughly classified into two classes: VoD and live videos. Some researchers have explored a third class, which is referred to as catch-up TV or time-shifted streaming [LS10, LS11, HBC+11, DN08]. The live streams is recorded on the fly and is then offered in an on-demand service. Even though the demand for these services is growing, the related literature can still not be compared with VoD and live videos.

VoD corresponds to the services that allow users to re-access the same video content multiple times at any moment. Hence, in VoD, different users have different playback positions at the same time, even for the same video content. Whereas, live video streaming broadcasts real-time videos to users with the same playback position simultaneously. The two mediums have some strategic differences. Live streaming is especially useful for videos requiring tight timeliness: breaking news, hot sports, real-time cooperation, etc. In the following, we list several characteristics of the live video streaming services.

- **Real-time.** The play-out delay is defined as the time between the generation of the content on the content provider (or video source) and its reproduction

on receiver players. The timeliness of live video streaming requires relatively low play-out delay, ranging from few seconds to few tens of seconds. Besides, to fluently play the video, the play-out delay is required to be maintained. For example, increasing the delay can result in video freezing and consequently in low user experience.

- **Synchronism.** The group of users watching the same video content have synchronous video play-out delay. Moreover, end users can start to watch the video, as well as stop watching the video at any moment. Consequently, the group of users may change rapidly, and system dynamics should be carefully considered.

- **Bandwidth constraint.** Live video streaming is demanding on resources (for example the bandwidth resource) although it is possible to delay bulk transfers at off-peak hours in VoD [LSYR11]. The timeliness of live streaming requires provisioning bandwidth. Hence, the efficiency of bandwidth management is one of the first requirements of live streaming systems.

As previously said, two categories of solutions exist for online live video streaming: P2P and CDN. Both solutions aim to enhance the scalability of the traditional client/server (C/S) architecture. In the C/S architecture, streams are directly sent from the server to each client. This model has limited capacity, and is error-prone to the single point of failure (SPOF). In P2P systems, the end-users contribute to the delivery with their own computing and networking resources, so the total system capacity scales up with the size of the system. In CDN, a large number of dedicated servers are deployed at carefully selected locations in the Internet. These servers cooperate with each other to serve content requests from end users, and increase the fan-out of the CDN *origin* server. These two techniques are complementary to each other. They both have pros and cons:

- P2P networks do not need any central server. Or, at most a central actor (such as the *tracker*) with light traffic load is sufficient. However, the quality of the service provided by a P2P network cannot be fully guaranteed. The main challenges of P2P systems are as follows. First, peer churn can make the network instable. Second, the system has to deal with the heterogeneity of peers. Moreover, in distributed environments, user privacy and security can be important issues and must be treated carefully. Last, there is no guarantee that peers can provide sufficient aggregate resources (*e.g.* upload bandwidth), consequently, the delivery network can be underprovisioned.

- CDNs can provide guaranteed streaming services with low delay and jitter. However, this comes from provisioning the CDN with sufficient capacity at a non-negligible cost. The cost of the CDN operators includes placing a large number of edge servers, provisioning a delivering network, and pay for the traffic load on the peering points. Currently, the development of peering agreements between CDN and network operators has reduced the importance of transmission cost in CDN. In addition, some CDN operators (e.g. AT&T Inc. and Level 3) have built their own networks to further lower down the in-house transmission cost. But still, the cost of maintaining and operating this infrastructure is a constant concern.

Based on the above discussion, P2P-based and CDN-based live streaming service providers have different optimization objectives during the system design phase. Typically, for P2P, the target is to maximize the quality of the service perceived on end users by fully utilizing the resources provided by each peer. Whereas for CDN, the objective of the CDN provider would be to find bandwidth efficient delivery scheme to provide satisfactory services to end users with limited infrastructure cost. In the following, we will fully introduce the state-of-the-art of the two techniques. We still focus on live video streaming. Moreover, after the introduction of each technique, we relate a discussion to the objectives of this thesis: (1) multioverlay P2P systems, and (2) rate-adaptive live streaming CDN systems.

2.2 Peer-to-Peer Live Video Streaming

2.2.1 Overall Perspective

In a P2P network, peers not only download data from, but also upload data to the other peers in the network. Obviously, P2P brings two key advantages to delivery systems. Firstly, since peers provide their own resources, the total capacity of the system increases linearly with the size of the audience. Secondly, although most commercially running P2P networks rely on a central server (such as tracker, bootstrapper or update server), the decentralized nature of P2P systems enhances the robustness of the system by mitigating the problem of having a single point of failure. It is thus not surprising that P2P architectures have been successfully used for content distribution. Especially, P2P techniques for online live video streaming have been investigated for more than a decade. Multiple systems have been designed and implemented (*e.g.* Coolstreaming [ZLLsPY05], PPlive [ppl], UUSee [UUS], SopCast [Sop], GnuStream [JDXB03] and the list is endless). These systems have demonstrated that P2P is a cost effective, highly scalable solution for online live streaming.

In a P2P streaming system, peers form an application layer network called *overlay* network. Then, content is delivered through this overlay network. P2P streaming systems can be roughly classified into two categories of overlay structures: *tree-based* overlays and *mesh-based* overlays. In the following, we will discuss each class separately.

2.2.2 Tree-based P2P Systems

Tree-based overlays follows the basic idea of traditional IP multicast. Peers participating to the same video streaming session are organized into a tree structure, which is rooted on the video source. Then, the video content is pushed from the root down to the leaves along the tree. This method is also referred as *application layer multicast*. The tree structure is optimal in terms of end-to-end delays. Therefore, many early P2P streaming systems have been designed with a tree-based overlay, including ALMI [PSVW01], SALM [BBK02], ESM [CRZ00], and Overcast [JGJ+00].

We take the example of ESM [CRZ00] to detail the construction of data delivery trees over a real underlying communication network. The basic idea of ESM is to form an overlay layer that covers the video source and all peers. Then, on top of the peer

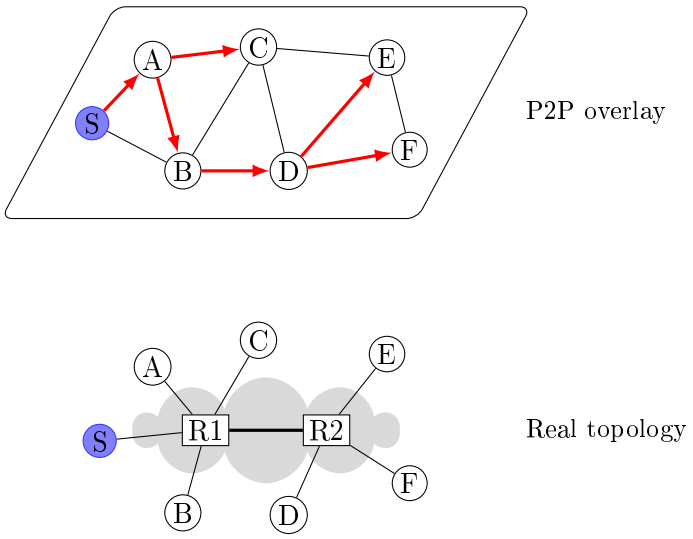


Figure 2.1: Tree-based P2P overlay

network, a spanning tree is constructed. We show this process in Figure 2.1 which includes one video source (S) and six end hosts (A to F). On the bottom part of Figure 2.1 the underlying communication network is shown: end hosts are connected to each other through two routers. Then, based on the topology of the network, the system builds an application layer overlay (as shown in the top part of Figure 2.1) such that (1) each peer has limited number of neighbors and (2) a link between two peers has a good enough performances, e.g. in latency and bandwidth. On top of this overlay, a spanning tree construction algorithm can be used to build data delivery tree (the red arrows in Figure 2.1).

Many tree-based systems have been proposed in the literature. For example ALMI [PSVW01] creates a minimum spanning tree, in which the cost of each link is an application specific metric (such as delay, or distance). However, to the best of our knowledge, no commercial large-scale P2P video streaming system utilizes the tree-based structure. The reason is mainly the high tree maintenance cost and bandwidth inefficiency. Indeed, tree structure are vulnerable to system dynamics and peer churn. A peer departure will affect all its descendants. Besides, it is also vulnerable to peer bandwidth variation. Peer upload bandwidth determines the number of children in the tree. When the upload bandwidth of a peer changes, some of its children are no longer able to be connected to the tree. As a result, the tree structure has high maintenance cost. Moreover, the tree structure is also inefficient in terms of bandwidth utilization. In trees, the leaf peers cannot contribute any of their upload bandwidth resource to the system. Since many peers are leaves in the tree structure, this significantly reduces the bandwidth efficiency.

To overcome the aforementioned drawbacks of the tree structure, systems based on multiple trees, such as SplitStream [CDK+03] have been proposed. The key idea in SplitStream is to split the content into k *stripes* and to multicast each stripe using a separate tree. Splitting the original video into k stripes corresponds to applications using rateless coding (*e.g.*, Raptor codes [Sho06] and LT codes [Lub02]), or multiple

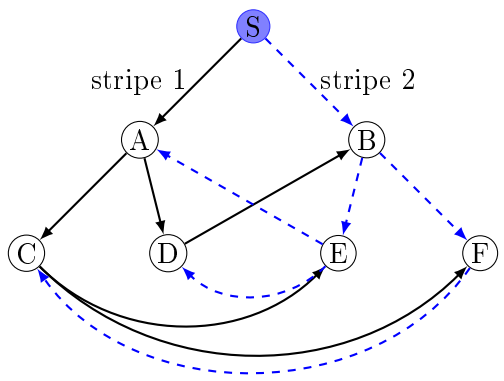


Figure 2.2: Multitree-based P2P overlay

description coding (MDC) [WRL05]. As shown in Figure 2.2, the source has two stripes to deliver to six end hosts. The first tree (shown in black arrows) carrying stripe 1 are delivered through hosts A, C and D, whereas the second tree (shown in blue arrows) are delivered through the other three end hosts B, E and F. The benefits of using multitree structure are twofold: (1). It makes the system more robust to peer churn. (2). Resource of the system are better utilized.

Despite the advantages of multitree structure, it is still tree-based. Thus, it suffers from some of the main drawbacks of the tree-based structure. In particular the construction and maintenance of multiple-trees are still costly due to peer churn. behaviors.

2.2.3 Mesh-based P2P Systems

As the name suggests, in mesh-based P2P streaming systems, peers relationship can be represented as a mesh. That is, unlike in tree structures where peers can only get the whole data from one peer (their parent in the tree), peers in mesh overlays are able to connect to and retrieve data from multiple other peers, namely their *neighbors*. Peers relationship is maintained in a dynamic way. New neighbors can be added and responseless neighbors can be removed. This design enhances the robustness of the system against peer churn: if a peer's neighbor leaves, the peer can still download video content from the remaining neighbors.

We illustrate in Figure 2.3 how peers in the same video session form a mesh overlay. The system includes a *tracker* server to keep track of the active peers. As soon as a peer enters the system, it first contacts the tracker to register its information (step 1 in Figure 2.3). Then, the tracker responds with a list of peers who are assumed to be active at that moment in the same streaming session (step 2). After receiving the peer list, the peer tries to establish connections to these peers by sending connection requests (step 3). If the remote peer approves, a positive reply message is sent (step 4). Then, the connection is established and both peers add each other to their neighbor list. This process continues until the peer is connected to a sufficient number of neighbors. For system dynamics, if a peer gracefully leaves the system, it notifies the tracker and its neighbors. For unexpected departure (such as crash), the tracker and

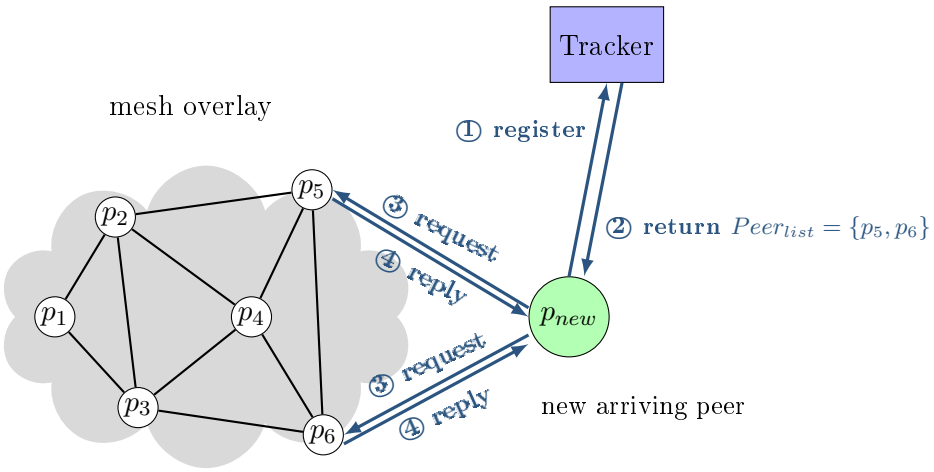


Figure 2.3: Formulation of a mesh-based P2P overlay

peers can also remove responseless peers. In order to maintain good connectivity, peers continues refreshing their neighbor list, and new neighbors can be added if necessary.

A key idea in the mesh-based P2P system design is to cut the original video into a sequence of small video portions with a short period (*e.g.* 1 second). The video portion are named as video *chunks*. In tree-based system, the video can be seen as a flow pushed from the root of the tree down to the leaves, whereas in mesh-based system, small video chunks are pulled in parallel from multiple neighbors and assembled into the original video by the peer for display. Each chunk has a unique id (or sequence number) which implies its playback time. For example, lower id chunk will be played earlier. Peers buffer the received chunks in order to (1) put them back in order for playing, and (2) offer the received chunks for its neighbors to download. To transmit the video data, each peer maintains a *buffer map* which indicates its chunk availability and exchanges it with all its neighbors periodically. By receiving the buffer maps of it neighbors, a peer can decide which chunks and from which peers it should request.

Mesh-based systems suffer from two potential drawbacks: First, frequent buffer map exchanges result in higher overhead. Second, pulling each individual video chunk can introduce additional transmission delays. However, the advantages of mesh-based systems, especially the ability to adapt to system dynamics, finally leads to successful deployment in the real-world. As far as we know, large-scale commercial running P2P live video systems utilize mesh-based overlays. A simulation study [MRG07] also revealed that mesh-based systems have better performance than tree-based systems. The advantages of mesh-based systems include robustness, scalability and bandwidth efficiency. For robustness, peer churn can be recovered through quick neighbor discovery with gossip protocols. Also, mesh-based overlays are immune to peer upload bandwidth variation. For scalability, except the tracker, each peer is required to just maintain a local awareness of other peers, thus the system can easily scale up. For bandwidth efficiency, each peer provides its upload bandwidth for its neighbors to download available chunks, which results in a high utilization of the bandwidth.

Notation	*Tree-based*	*Mesh-based*
Complexity	high	low
Source-to-end delay	low	high
Overhead	low	high
Peer churn	vulnerable	immune
Bandwidth variation	vulnerable	immune
Maintenance cost	high	low

Table 2.1: Comparison of tree-based and mesh-based P2P streaming protocols

Multiple mesh-based P2P live video streaming systems have been implemented. The most popular systems are probably CoolStreaming and PPLive [Pas12, HLL+07]. The Coolstreaming system, which was released in summer 2004, represents the first successful large-scale P2P live streaming system [LXQ+08]. CoolStreaming, which is based on a mesh network, has been further extended in [XLKZ07]. PPLive is one of the most popular large scale P2P live video streaming applications. Measurements [HLL+07] have revealed that PPLive follows the major design principle of mesh-based P2P live video streaming system: a gossip-based protocol for peer management and channel discovery; a mesh-pull P2P architecture for high quality video streaming.

We summarize the pros and cons of both tree- and mesh-based approaches in Table 2.1. The tree-based approach pushes the video flow along the tree, thus it exhibits relatively low source-to-end delay. Moreover, the control message overheads is lower than that of the mesh design. On the other hand, the mesh-based approach outperforms the tree-based one on the other aspects. The design of the mesh-based P2P protocol is arguably simpler than that of the tree-based. Moreover, it is robust to system dynamics such as peer churn and peer upload bandwidth variation, and finally has lower maintenance cost.

2.2.4 Multioverlay P2P Systems

A multioverlay P2P live streaming system consists of multiple peer-to-peer overlays, each for one video's diffusion. Actually, almost all P2P live video streaming systems do their best to provide as many live videos (channels) as possible, thus they all consist of multiple P2P overlays. As for our example in Section 1.2, multioverlay systems are useful in many scenarios.

A number of work in the literature have also considered scenarios where a peer can watch several P2P live streams simultaneously. In [LZL11], the authors proposed a multi-swarm P2P video conferencing system. In such a system, there are multiple users simultaneously distribute their video streams to multiple receivers. At the same time, a user can receive multiple streams from the same conference group. In [WL07, WLL08], the authors considered a scenario that the upstream peers or the streaming servers distribute several videos simultaneously. From another view points, these nodes participate in multiple P2P networks, hence they are the receivers of multiple concurrent videos. In [WXR09, WXR11], the author also studied a multi-view P2P live streaming system, where a user can simultaneously watch multiple

channels. At last, we also discussed in [ABH+11] an application scenario of the multioverlay P2P system where MMOG players from the same team stream the videos of their game to each other for game cooperation purpose.

One of the thesis targets is the design of a multioverlay P2P live video streaming system to share a large number of user-generated live videos to a large-scale end users. On the system level, two design problems have to be addressed: the intra-overlay video streaming problem and the inter-overlay bandwidth allocation problem, which is the decision that peers have to take about sharing their upload bandwidth into concurrent overlays. As we discussed in previous sections, the intra-overlay video streaming problem has been well investigated in the literature for more than a decade. Plenty existing P2P live streaming protocols can be deployed directly. In the following, we explain our idea of solving these two problems independently and the rationale behind such a design.

Inter-overlay Bandwidth Allocation vs. Intra-overlay Video Streaming

The existing work solving bandwidth allocation problem in the multioverlay system can be roughly classified into two classes. The first class solves the intra-overlay video streaming problem and the inter-overlay bandwidth allocation problem all-at-once [LZL11, WL07, WLL08]. It means that their solution determines the bandwidth allocated to each overlay on each peer, as well as the overlay construction for each P2P network. In [LZL11], the authors investigated optimal bandwidth sharing strategies for a multi-swarm video conferencing application. To compensate bandwidth deficit, the authors introduced bandwidth *helpers* (servers that provide upload bandwidth). The objective is to find the optimum video bit rates of video sources and helper sharing such that the utility of the system is maximized. Moreover, they use specific tree structure for intra-overlay video delivery. The construction of the tree is based on the allocated peer upload bandwidth for the overlay. In [WL07, WLL08], the inter-overlay bandwidth allocation problem is solved with a distributed dynamic auction game. Each upstream peer dynamically and locally organizes an auction game in which the downstream peers bid for its upload bandwidth. According to the results of the auction games, peers form a mesh-based overlay. This proposal requires a specific media distribution mechanism based on network coding.

We believe contrarily that a clear separation between inter-overlay resource management and intra-overlay video diffusion brings more benefits to the system.

- Firstly, existing P2P systems can be used. One of the motivations is to leverage the advances that are obtained through research. For example, some works have designed promising solutions for the P2P delivery of video in underprovisioned overlay [LSR+09]. The implementation of such P2P systems for the intra-overlay would be appropriate when the system is underprovisioned.

- Secondly, separating these two problems can enhance the system robustness in a dynamic environment. In multioverlay P2P system, besides regular peer churn, system dynamics also include the variation of allocated peer upload bandwidth into the overlays. In a dynamic system, peer joining, leaving, and switching videos lead to environment changes. Consequently, peers should reallocate their

upload bandwidth according to the new settings. In all-at-once solutions, such changes lead to the reconstruction of the overlays.

- Last, solving the two problems together brings additional complexity. The problem formulation for such a double-objective problem is more complex. On the contrary, separating the two problems can simplify the problem.

Our main idea is to first solve the inter-overlay bandwidth allocation problem, then, upon the allocated bandwidth into each overlay, to use a state-of-the-art P2P video streaming protocol stands for constructing P2P overlays and streaming the videos. This idea is similar to the work presented in [WXR09, WXR11], which represent the second class of approaches in the literature. This design requires the P2P video streaming protocol to well adapt to system dynamics, such as peer churn and peer bandwidth variations. Based on the discussions in previous sections, two possible solutions are available: the tree-based P2P streaming protocols and the mesh-based ones. As previously said, mesh-based P2P streaming protocols are a better fit for the intra-overlay video streaming. Such a design perfectly combine the two problems, inter-overlay bandwidth allocation and intra-overlay video streaming, and reduce the complexity of the whole system.

System Underprovisioning

At last, it is worth to mention that none of the existing works deal with system underprovisioning. If a P2P network can receive enough reserved upload bandwidth (equal to or higher than the streaming requirement) from its peers, it is said to be *overprovisioned*, otherwise, it is *underprovisioned*. In other words, an underprovisioned P2P network endures bandwidth deficit. In the context of intensive user-generated video sharing, multioverlay P2P systems can be underprovisioned. That is, the aggregated peers upload bandwidth is not sufficient to deliver all the videos to all the peers in the system. In this case, to fully utilize peers upload bandwidth, a consequent optimization goal is to minimize the bandwidth deficit: peers should allocate their upload bandwidth to the underprovisioned overlays instead of overprovisioned ones. We will further detail this optimization goal in Chapter 3.

The study of underprovisioned P2P overlays represents a new research topic that has not received enough attention. In general, scientists have first tried to find a way to provision the system so that they are no longer facing underprovisioning. For example, the lack of bandwidth resources can be tackled by provisioning on-the-fly some additional servers to assist in the diffusion of the video [WLZ08, SIB12, LZL11]. This approach is compatible with our multioverlay system. Our goal of minimizing the bandwidth deficit of the system also achieves the minimum upload bandwidth consumption on the server side. Another approach is to authorize peers to contribute to an overlay they have not subscribed to [WLLR10]. This approach works only if the overall system is overprovisioned. Moreover, in the context of live video sharing in social network applications, privacy concerns make it unacceptable.

For multioverlay P2P video streaming system, the aforementioned related works did not investigate system underprovisioning. In [LZL11], bandwidth helpers are introduced to avoid the system becoming underprovisioned. In [WL07, WLL08],

the system cannot even bear underprovisioning, because their algorithm requires the system to be overprovisioned in order to converge. In the most relevant work [WXR09, WXR11], the authors propose a bandwidth allocation protocol, named DAC, which fairly allocates upload bandwidth based on each overlay streaming demand. However, this approach is oblivious to each individual overlay's provisioning information. Thus, the protocol leads to biased results such that most resources are allocated to the overlays with high demand, although the excess resources could be allocated to other underprovisioned ones to alleviate bandwidth deficits in these overlays. As a result, the overlays with low demand may suffer from bandwidth deficit even when the system is overprovisioned. Our multioverlay P2P video sharing system is further detailed in Chapter 3. We compared our solutions to this most relevant work. A detailed comparison is available in Section 3.7.

2.3 Live Video Streaming over CDN

2.3.1 CDN: Overall Perspective

CDN is actually the main method of content delivery in today's Internet, including web objects, applications, on-demand streaming data, and live streaming media. It relies on replicating the content on a set of CDN edge servers to provide accelerated and reliable content delivery to a large number of end users. It basically increases the fan-out of the main server.

The CDN providers implement three main algorithms for their system: *content placement*, *content delivery*, and *user redirection* algorithm. Content placement decides where to replicate content on different edge servers. It works in combination with the user redirection mechanism. The basic idea is to replicate content, and re-direct content requests to one of the replicas according to some selection strategy.

Content Placement

There are mainly three content placement approaches: push, non-cooperative pull, and cooperative pull. The push methods proactively replicate the content on edge servers before they are actually requested. Such a method can result in inefficient system performance and bandwidth wastage. In non-cooperative pull methods, edge servers fetch the content from sources only when a client request cannot be satisfied. At last, cooperative pull approach further improves the second one by edge servers cooperations for downloading a missing content. The pull based approaches are largely utilized for VoD services [DMP+02, Pas12]. However, for live streaming, CDN should switch from the pull-based strategy to a push-based one [ASV11]. A consequent challenge for live content placement is to accurately predict future user requests [WLZ11b, WLC12].

Content Delivery

A CDN delivery network is composed of a set of communication devices and a set of directed communication links. The previous theoretical works related to live streaming in CDNs [AMMS03, AEVW04, NT05, AMM+11, ASV11, NSS10] have highlighted

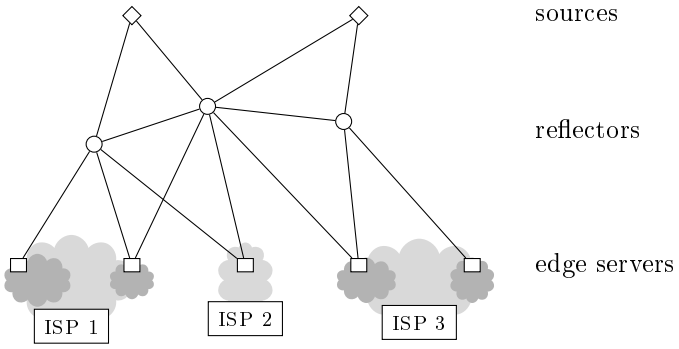


Figure 2.4: CDN delivery network

the main characteristics of these networks, in particular the 3-tier topology (source servers, reflectors and edge servers), and the restriction on the upload capacity of the equipment. We show in Figure 2.4 this 3-tier architecture of the CDN delivery network. Basically, the CDN delivery network includes the following three types of equipments: a relatively small number of *sources*, which directly receive the raw videos from the content provider; a medium size network of *reflectors*, which links the sources to the edge servers; and a large number of *edge servers* located in different Internet Service Providers (ISP). For live streaming, the tree-based structures have been proven to be efficient and robust in such managed infrastructure [AMMS03, ZAB+12, ASV11]. These methods construct content delivering trees rooted on the CDN sources with CDN edge servers as leaves to constantly push the live content through the CDN delivery network.

User Redirection

Request redirection transparently redirects a client request for a content to its actual location in a CDN edge server. Transparent means that users get the requested content as if they got them from the server of the content provider. There are two techniques for redirecting users to proper edge servers: uniform resource locator (URL) rewriting and domain name system (DNS) redirection. With URL rewriting, the URL in the original request are rewritten to the CDN server addresses. The DNS redirection is used by Akamai, the world wide leader of CDN providers. A client issues a request to a local DNS server. The local DNS then points to a hierarchy of DNS servers which manage the URL space of the CDN. Finally, the CDN DNS servers select an edge server according to some strategies to redirect the user. The redirection strategy is the core of this redirection process. Studies have shown that the redirection depends on a mix of network proximity, load-balancing and business issues [AJZ11, TFK+11b, PB12, JSZ12, TL12].

2.3.2 CDN Architecture for Live Streaming

We illustrate in Figure 2.5 the traditional architecture for the delivery of live video streams in the Internet. The content providers transmit their video content to the CDN providers for delivery. The CDN providers push the video streams to a large

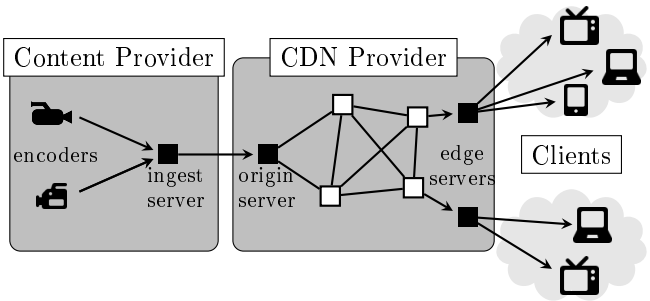


Figure 2.5: Live stream delivery over CDN: the main actors

set of CDN edge servers that are located near to the end users. On the end of this chain, a large number of heterogenous end users consume the streams through a wide range of electronics (from high definition TVs to smartphones).

The central actor is the CDN provider which links content provider to the content consumer. Thus, the goal of a CDN is to offload the traffic originated to the content provider's infrastructure, and to make the content widely available to end-users with high performance by its widely deployed servers. A typical live stream delivery within the CDN network contains the following three phases, with each corresponds to one type of the CDN equipments:

- **Phase I: Transcoding.** Transcoding is performed by CDN sources. The source nodes receive the raw live stream data from the content provider. Then, they are responsible for coding it into a set of live streams and then forward these streams to the reflector nodes.

- **Phase II: Multiplication.** The sources themselves cannot achieve the required output capacity to deliver the stream to all edge servers. The reflectors aim to increase the fan-out of the sources. They multiply the received stream data and forward the stream data to the edge servers.

- **Phase III: Delivery.** At last, the edge servers receive the live streams and offer them to the clients inside their respective Internet Service Providers (ISPs).

In comparison to the infrastructureless approaches (such as P2P-based approaches), CDN provides guaranteed video services with high performance, for example, with low delay, high video bit rate, and low jitter. However, this comes from provisioning the CDN network with high cost. For CDN providers, a major challenge today is the contradiction between the rapidly increasing traffic volume and the deficient network capacity (for example, the underprovisioned CDN delivery network). Moreover, the rate-adaptive streaming technique (such as the DASH protocol) has been widely deployed. One objective of this thesis is the design of bandwidth efficient solutions for live rate adaptive streaming in underprovisioned CDN delivery network. In the following, we will first introduce the mechanism of rate adaptive streaming by discussing in detail the DASH protocol. Then, we will discuss the process of live rate adaptive streaming in CDN delivery network and related works on this domain.

Representation	Bit rate(kbps)	Resolution
Representation 1	330	320x180
Representation 2	700	640x360
Representation 3	1500	640x360
Representation 4	2500	1080x720
Representation 5	3500	1080x720

Table 2.2: Representations in Akamai adaptive video streaming

2.3.3 Dynamic Adaptive Streaming over HTTP (DASH)

Today's video services are accessed from a wide range of devices, from smartphone, tablet to wired PC and connected TV. It is common for mobile users to enjoy the online video entertainment through wireless connections (3G, WiFi). A recent Cisco report shows that mobile video traffic exceeded 50 percent by the end of 2012 [cisa]. Moreover, the statistics collected during London Olympic Games shows that half of video requests are released from mobile devices [nbc]. Although the throughput of mobile network connection is increasing, it is still relatively low for high bandwidth consuming online video services, especially for HD videos. The average mobile network throughput for smartphones in 2012 was 2064 kbps, up from 1211 kbps in 2011 [cisa]. Wired users are generally equipped with larger broadband connections and capable to enjoy a video resolutions up to high definition (1920x1080), whereas the wireless users are limited to a low video resolution by their devices and lower wireless connections. A challenge is to maximize the Quality of user Experience (QoE) of diverse types of users.

Rate adaptive streaming has been designed to improve the user experience of multimedia streaming services in a heterogeneous context. It provides an efficient and easy solution to stream multimedia to diverse users, connected to the Internet with different types of connections, by using the existing HTTP protocol. The recent Dynamic Adaptive Streaming over HTTP (DASH) standard [Sto11], specified by the 3rd Generation Partnership Project (3GPP) and the Moving Picture Experts Group (MPEG), is a new step toward a broader adoption. Without loss of generality, we will introduce the DASH streaming protocol in detail.

The basic idea of DASH is that each video is cut into multiple small *segments*, and each segment holds several video *representations*. Each representation is associated with a certain quality, thus a certain encoding and a certain bit-rate. For example, we list in Table 2.2 the representations utilized in Akamai HTTP-based adaptive HDTV live streaming service [CM10]. Each video is encoded into 5 versions. The video bit rate of the lowest representation is 330 kbps, which corresponds to a low resolution. Whereas the highest representation represents the HD 1080p version of the video with bit rate up to 3500 kbps.

During the transmission of the video, end users adaptively choose the proper representation for each segment, that is, the representation that maximizes the video quality that is allowed by both her network connection and device. Thus, the video bit rate is adaptively and dynamically determined by the last-mile available upload

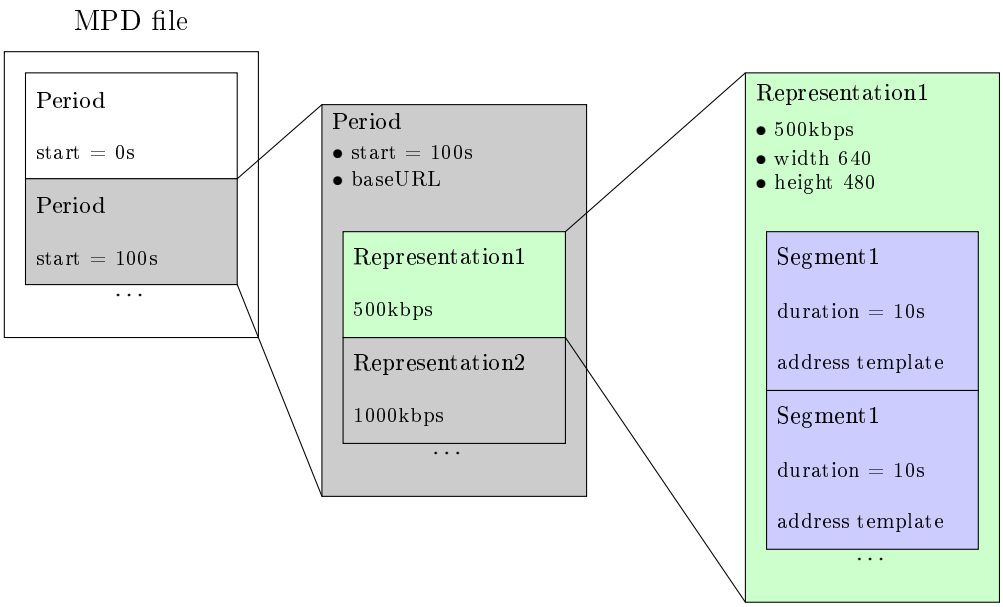


Figure 2.6: Media Presentation Data file

bandwidth on each user. Rate adaptation algorithms also represent a major research direction in the literature for DASH live streaming [ABD11, MLCC12, LBHG12].

DASH leverages on the XML-formatted media presentation description (MPD) file to indicate the users the detailed information about segments and available representations. To play the video content, the DASH client first requests the MPD file from the server. By parsing the MPD file, the client knows where to access each video data and sends consecutive HTTP requests. Typically, the MPD file contains the following information [Sto11], and the structure of the MPD file is further illustrated in Figure 2.6.

- First of all, the MPD file consists of a sequence of *periods* by specifying the period start time.
- Each period states multiple *representations*. Each representation corresponds to a different user choice on bit rate, resolution, and encoding, etc.
- Within each representation, a set of *segments* are listed. Segments is a data unit that can be uniquely referenced by an HTTP-URL.
- Within each segment, the segment info details how to access the data by showing the URL of the data unit.

DASH is also an HTTP-based streaming techniques, which avoids NAT/firewall traversal issues and also has the advantage of reusing the widely deployed standard HTTP caches (*e.g.* existing Internet CDNs). Thus, this technique is appropriate for CDN. Nowadays, there are plenty of commercial running adaptive streaming services:

```
<?xml version="1.0" encoding="utf-8"?>
<MPD
    type="Live"
    minBufferTime="PT3S"
    availabilityStartTime="2010-04-26T08:45:00-08:00"
    minimumUpdatePeriodMPD="PT5M0S"
    timeShiftBufferDepth="PT1H30M0S"
    xmlns="urn:3GPP:ns:PSS:AdaptiveHTTPStreamingMPD:2009"
>
<Period segmentAlignmentFlag="true">
    <Representation bandwidth="128000">
        <SegmentInfo duration="PT10S" baseURL="mt500">
        <InitialisationSegmentURL sourceURL="fifa128seg_i.3gp"/>
        <UrlTemplate sourceURL="$Index$.3gs"/>
        </SegmentInfo>
    </Representation>
    <Representation bandwidth="512000">
        <SegmentInfo duration="PT10S" baseURL="mt1000">
        <InitialisationSegmentURL sourceURL="fifa512seg_i.3gp"/>
        <UrlTemplate sourceURL="$Index$.3gs"/>
        </SegmentInfo>
    </Representation>
</Period>
</MPD>
```

Figure 2.7: A live MPD file example

- IIS Smooth Streaming is HTTP-based adaptive streaming platform provided by Microsoft [IIS].
- Adobe Flash Dynamic Streaming solution is a web-based service available to all devices running a browser with Adobe Flash plugin [ado].
- Apple HTTP Live Streaming is a client-side HTTP adaptive live streaming solution released by Apple for its products [appa].
- Akamai HD Network is the HTTP-based adaptive HDTV live streaming service provided by the worldwide leader of CDN, Akamai [aka].
- Netflix adopts the DASH protocol for streaming online movies and TV shows [AGH+12].

2.3.4 Live DASH over CDN

The DASH streaming protocol supports various video services, including on-demand and live video steaming. In comparison to on-demand video services, live streaming has special requirements and challenges. In live streaming, the content is generated on

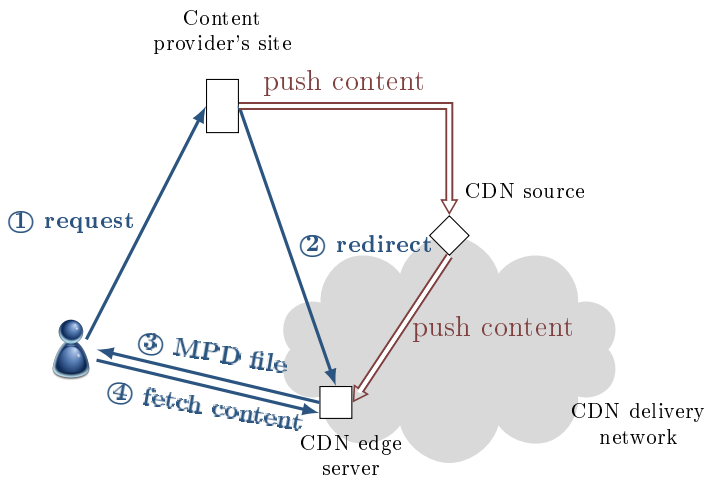


Figure 2.8: Distributing live rate adaptive streams in CDN

the fly. DASH leverages on MPD updates to cope with real-time content generation. To start the stream, the client first gets an initial MPD file that describes a certain periods of accessible segments for that moment. As new content generates, the client updates the MPD file and continues watching the up-coming video. Besides, DASH synchronize on Universal Time Clock (UTC) time in the MPD file for live streaming to provide consistent information to clients.

Figure 2.7 depicts an example MPD file for live DASH video taken from [Sto11] and [LEF+11]. The `type` field is assigned to live to indicate that the video is a live one. The `minimumUpdatePeriodMPD` field gives the duration for user MPD update. The user should check for MPD updates at the indicated interval. The new MPD contains the same metadata for future media. Commonly, the update interval is longer than segment duration so that an MPD file contains the information for a certain amount of segments. In the example, the segment duration is 10 seconds, whereas the MPD update duration is set to 5 minutes. The `availabilityStartTime` announces the start time for the first video segment (the start time of the first period) in the MPD file. As previously said, the global UTC time is used for synchronization propose.

We illustrate the process of live rate adaptive streaming in Figure 2.8. In the user side, when a user clicks on the thumbnail of a video on the content provider's site, the user sends an HTTP GET request for the content. This request is redirected to an CDN edge server according to the CDN redirection strategy. The selected edge server releases a MPD file to the user. By parsing the MPD file, the user is able to fetch the live video content for display. When necessary, a MPD file update request is sent, and new MPD file is replied. For the CDN side, the content provider pushes the content to be delivered to the CDN source server. For live streaming, the CDN works in a push way such that the video flow is pushed continuously from the source to the edge servers through the CDN delivery network.

There are several challenges in the above process. First of all, the sharp growth of video traffic is a severe threat for the CDN infrastructure. Besides, the multi-representations characteristic of the rate adaptive streaming aggravates such capacity

problem, since for a single channel the whole set of representations (frequently more than 20Mbps) should be delivered to the edge servers [AGH+12]. As a result, the CDN delivery network can be underprovisioned: the network could not afford the delivery of all the required streams to all the edge servers. In this context, the objective of the CDN providers for live streaming is to maximize the throughput of their network under the restriction on the upload capacity of the CDN forwarding equipments. In the following, we will survey some of the relevant literature in related areas.

Related Works

A surprisingly low amount of work are related to live video streaming in CDN networks. The earliest works [AMMS03, AEVW04] only deal with one stream. Multiple streams delivery is more complex because the forwarding nodes should determine how to allocate their upload bandwidth into the multiple streaming sessions for a global optimization objective. Some more recent works [AMM+11, NSS10, ASV11] consider the node upload bandwidth limitation and multiple streams. Their objective is to reduce the bandwidth cost subject to the resource constraints. This objective is outdated since CDNs and ISPs develop peering agreements, which reduce the importance of the bandwidth cost. On the other hand, the major concern for current CDN providers is the huge volume of video traffic to deliver, and the transmission burden on its limited CDN infrastructure. Thus, an appropriate objective for current CDN providers is to maximize the throughput of the CDN delivery network. The relation between edge servers and end users in the context of DASH is also widely studied in the literature [LBHG12, CMP11]. These studies could complement our work, which only deals with content delivery within the CDN infrastructure.

The problem of maximizing the throughput of a network, subject to the upload capacities of the nodes has been addressed as the *streaming capacity* problem in the context of P2P networks [SLC+11, ZLW11, NL11]. Their goal is to maximize the deliverable bit rate of the network. In other words, their model is based on *elastic* video bit rate. However, in the context of rate adaptive streaming, the bit rate of the representations are well predefined (such as in Table 2.2). Hence, the throughput of the network is maximized when the number of delivered streams is maximized. We refer to such optimization problem as *discretized streaming capacity* problem. It is introduced in Chapter 4. This optimization problem provides a theoretical foundation for optimal live rate adaptive streams delivery in the CDN infrastructure.

For live streaming in CDN, the tree structure is efficient and robust because the infrastructure is under the control of the CDN provider. The delivery trees are rooted on the CDN sources, and push the content flow down to the leaves, which covers the CDN edge servers. Thus, we aim to construct a set of delivery trees that achieve the above-mentioned optimization goal. There are two set of related work: the multiple tree packing problems and the minimum Bounded Degree Spanning Tree (BDST) problem.

The multiple tree packing problems have been studied in the context of peer-assisted systems [RBS12, SIB12]. The problem is to minimize the amount of additional resources to serve all peers in a P2P system. This problem is different in CDNs, which are self-sustained networks. The missing resources cannot be compensated,

rather, bandwidth resources need to be used in the best possible way. Numerous work have studied multi-tree packing for P2P application layer multicast protocols (see [HA+07] for a survey). The goal is to span *all* nodes under application related optimization objective (*e.g.*, to minimize tree height, or to reduce controlling overhead). However, in the context of CDN, each delivery tree only need to cover a set of edge servers (not all CDN equipments) in the network who require the stream. Such requirement is determined by the CDN content placement algorithms.

The BDST problem aims to determine a minimum-cost spanning tree while no node should have more than m children (see [Goe06]), which is NP-complete for any $m \geq 2$. Related variations of this problem feature non-uniform degree bounds [KR05]. Again, these work aim at spanning all nodes in the network while optimizing an objective function, while we aim at maximizing the number of spanned nodes under a node degree constraint. The only related work in this aspect is [BB05], which study the minimum spanning tree with at least k nodes in a weighted graph, but this work do not target the maximal k. Furthermore, these works do not deal with packing several trees and the resource allocation problem that such packing introduces.

The discretized streaming model for live rate adaptive streaming is introduced in Chapter 4. The objective of the problem is maximizing the utility of delivered streams from CDN sources to CDN edge servers through a set of delivery trees. Then, this theoretical work is further extended in Chapter 5. We propose a practical CDN system that efficiently delivers live rate adaptive streams in a large-scale and dynamic environment.

Chapter 3

Multioverlay P2P Video Sharing: Resource Allocation in Under-Provisioned System

3.1 Introduction

3.1.1 Multioverlay P2P Live Video Sharing System

In this chapter, we introduce our multioverlay P2P video sharing system. The architecture of the system is shown in Figure 3.1. As the figure depicts, the system consists of multiple P2P live video streaming networks. Each P2P network contains one *source* (the video generator) and all *peers* (the video receivers) that have subscribed to its live stream (channel). For example, in the figure, three sources are sharing their videos to a set of peers. A centralized actor, say a management server, is required to orchestrate the multiple overlays and perform necessary computations.

In multioverlay systems, a user can watch multiple live videos simultaneously. For example, in Figure 3.1, p_1 watches the videos from s_1 and s_2, while p_2 watches all the three videos. For such multi-watching characteristic, several possible application scenarios have been introduced in Section 1.2, and some related work are discussed in Section 2.2.4. As we discussed, besides the intra-overlay streaming problem, the system must also solve an inherent inter-overlay bandwidth allocation problem such that the multiple watching peers must decide how to share its uplink capacity among these concurrent overlays. In Section 2.2.4, we have stated our design principle: for intra-overlay video streaming, any state-of-the-art mesh-based P2P video streaming protocol can be deployed, and streaming overlays are constructed based on the allocated bandwidth. Then, in this chapter, we fully describe how we solve the inter-overlay bandwidth allocation problem.

For the bandwidth sharing problem, we especially focus on the provisioning of overlays. The provisioning of an overlay is calculated as the difference between the overlay demand (the amount of bandwidth required to serve all peers in this overlay) and the overlay capacity (the amount of bandwidth actually reserved). In the multioverlay context, resources can be *wasted* if they are allocated to overprovisioned

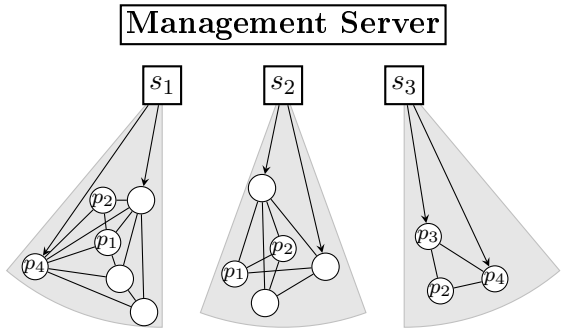


Figure 3.1: Three sources in a multioverlay live video system.

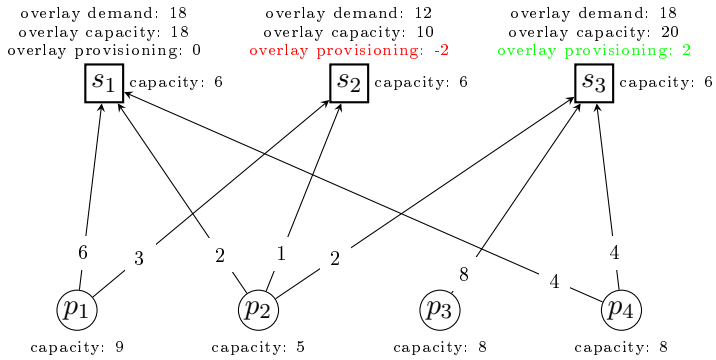


Figure 3.2: Example of a sub-optimal bandwidth allocation.

overlays, although they could be allocated to underprovisioned ones. We illustrate such bandwidth wastage in Figure 3.2. In the figure, an arrow from a peer to a source means that this peer reserves some resource units for this overlay. For example, peer p_2 reserves two resource units to the overprovisioned overlay s_3 although it could reserve them for the underprovisioned overlay s_2. This problem is especially critical when the system is underprovisioned, that is, when the aggregate upload bandwidth of all peers cannot meet the demand to deliver all the videos. In the context of extensive live video sharing, the system has to deliver a large number of videos, under the constraint of limited total peers upload bandwidth. As a result, the system can be underprovisioned. Our simulations also confirm that for realistic parameter settings, the system is often underprovisioned.

3.1.2 Our Contribution

Our first objective is to find bandwidth allocation solutions that minimize the waste of resources. We show that the problem can be transformed into the computation of a maximum flow in a bipartite graph. The maximum flow problem has been extensively studied in the literature, and fast polynomial algorithms exist. Hence, the optimal bandwidth allocation can be computed rapidly for a large scale system.

The second contribution is a set of bandwidth allocation strategies for underprovisioned system which share the inevitable resource deficit among overlays so that

a pre-determined policy is satisfied. Several bandwidth strategies can be tackled by introducing a cost function to our maximum-flow problem. This is a generic solution because different cost functions can be designed for various strategies. We enumerated four possible strategies: minimize the number of underprovisioned channels, prioritize the most popular channels, prioritize fee-paying users, and prioritize videos that are globally more frequently watched as preferred videos. The computation of a minimum-cost maximum flow finds allocations that are optimal in terms of resource waste and correspond to the best allocations with respect to the policy of the service provider.

Our third contribution is a distributed algorithm that allocates the bandwidth based on *proportional fairness* (bandwidth is allocated according to overlays demand). We show that this objective is equivalent to the maximization of the utility of flows in a bipartite flow network. We solve this optimization problem with the dual decomposition method. This method allows us to decompose the optimization problem into a set of subproblems and to solve it in a decentralized way with a distributed algorithm.

Finally, we present a set of simulations where we compare our strategies to the algorithms presented in the main previous work [WXR09, WXR11]. The simulation results shows that, when the system is overprovisioned, our strategies make sure that all peers receive high quality videos. In case the system is underprovisioned, our strategies guarantee that a large part of peers enjoy high quality videos, according to the policy given by the service provider. To the best of our knowledge, this is the first time that the quality of experience of users for diverse bandwidth allocation strategies in underprovisioned systems is studied.

The related work is discussed in Section 2.2.4. To sum up, our work improve the existing research in the following aspects: (*i*) none of the current work have proposed policies for the management of underprovisioned systems; (*ii*) their simulations considered only a small number of overlays (four overlays in most cases) with a very simple multioverlay structure. For example, in [WLL08, WLZ11b], each peer participates to all overlays, and in [WXR09] three simple overlay structures (chain, star and mesh) are studied. In our work, we simulated a large number of overlays (about 200), and the multioverlay structure is formed with a realistic simulation model.

The remainder of this Chapter is organized as follows. Section 3.2 defines the system model. Section 3.3 presents our bipartite flow network model and shows that the resource allocation with minimum total underprovisioning is equal to the maximum flow in the model. Section 3.4 proposes several bandwidth allocation strategies defined by properly defined cost functions in the flow network. Section 3.5 introduces the fair bandwidth allocation strategy and solves it using the dual decomposition method. Section 3.6 discusses the practical relevance of the system. Section 3.7 evaluates the performance of the system. Finally, conclusions are presented in Section 3.8.

3.2 System Model

We first give the notations used throughout this chapter (see Table 3.1). The system includes a global server. Its role is to authorize, or not, a peer to watch a video

emitted by a source, and to compute an optimal bandwidth allocation.

Sources. The set of sources is denoted by S. A source s is associated with an overlay G_s, which contains the set P_s of all peers that have subscribed to this overlay. To avoid confusion, $s \notin P_s$.

Peer-to-Peer Streaming. The intra-overlay P2P streaming system is out of the scope of this work. Our system is independent of intra-overlay structures. As we have discussed, any state-of-the-art mesh-based P2P live video streaming system can be used.

Peer Uplink Management. The set of all peers is denoted by P. We denote by $G(p)$ the set of sources from which the peer p receives a video. Every peer can use its uplink to transfer the chunks it received to other peers in the same overlays. The upload capacity of p is denoted by B_p while the upload capacity that p has reserved to serve video chunks in the overlay G_s is denoted by b_p^s. Clearly, we have the following constraint on peer upload bandwidth:

$$\sum_{s \in G(p)} b_p^s \leq B_p$$

Note that a source $s \in S$ can also be a receiver of another overlay as a casual peer. Yet, we assume that s reserves all of its upload bandwidth to its overlay G_s.

Overlay Capacity and Demand. The capacity of an overlay G_s is denoted by C_s:

$$C_s = \sum_{p \in P_s} b_p^s + B_s$$

That is, C_s is the aggregated upload bandwidth allocated from peers to G_s, plus the capacity of the source. The demand of an overlay corresponds to the smallest overlay capacity required to satisfy all peers in P_s. In a real system, the overhead resulting from the control traffic of P2P streaming protocols cannot be neglected. Therefore, the demand D_s of an overlay G_s contains two parts. The first part is the bandwidth required to stream the video to all peers. It is equal to $|P_s| \cdot d_s$, where d_s denotes the bit rate of the video emitted by s. The other part is the bandwidth used by the streaming protocol. This is equal to $|P_s| \cdot o_s$, where o_s represents the average overhead in the overlay G_s. As a result, the capacity of an overlay is calculated as:

$$D_s = |P_s| \cdot (d_s + o_s)$$

Overlay Provisioning. The provisioning Δ_s of a given overlay G_s is the difference between its capacity C_s and its demand D_s:

$$\Delta_s = C_s - D_s$$

An overlay is said to be underprovisioned when Δ_s is negative. The average upload capacity is smaller than the video bit rate, so some peers in this overlay are unable to watch the video at full quality. The smaller the provisioning, the worse the video quality experienced by the peers. On the other hand, the overlay is overprovisioned when Δ_s is positive. We define the notion of overlay underprovisioning as $|\Delta_s|$ if the overlay is underprovisioned and zero, otherwise.

P, S	set of peers, set of sources
G_s, P_s	overlay of source s and set of peers in G_s
B_p	upload capacity of a peer p
b_p^s	upload capacity reserved by p for G_s
$G(p)$	set of sources to which peer p subscribed
d_s	video bit rate of the video emitted by s
o_s	the average overhead of P2P streaming protocol in overlay G_s
D_s, C_s	demand and capacity of G_s
Δ_s, Δ_s^r	provisioning and relative provisioning of G_s

Table 3.1: Notations used in Chapter 3

Overlay Relative Provisioning. The relative provisioning Δ_s^r of a given overlay G_s is defined as the overlay provisioning divided by the number of peers in the overlay, that is:

$$\Delta_s^r = \frac{\Delta_s}{|P_s|}$$

System Provisioning: A system is said to be underprovisioned if the aggregate peer upload bandwidth cannot fulfill the demand of streaming all the videos, that is:

$$\sum_p B_p < \sum_s D_s$$

Otherwise, it is said to be overprovisioned.

3.3 Minimizing Under-provisioning: Bipartite Network Flow

In order to solve the bandwidth allocation problem, we build a theoretical abstract structure, which is a bipartite flow network $N = (V, E)$ according to source-peer relationship. For example, we represent in Figure 3.3 the bipartite graph related to the scenario of the system shown in Figure 3.1. The set V contains a virtual fountain l, a virtual sink q, the set P of all peers in the system, and the set S of all sources. Thus $V = P \cup S \cup \{l, q\}$.

The set of directed edges E gives the source-peer relationship. It includes three subsets. The first one, $E_1 = \{(l \to p) : p \in P\}$, contains edges from the fountain to each peer p with a maximum capacity of B_p. The second one, $E_2 = \{(p \to s) : p \in P, s \in G(p)\}$, contains edges from p to s if p subscribes to s with infinite maximum capacity. The third set, $E_3 = \{(s \to q) : s \in S\}$, contains edges from each source s to the sink with a maximum capacity equal to $D_s - B_s$, the overlay demand minus the source capacity.

This flow network is used by the bandwidth allocation algorithm. The capacity of edges of peers and sources indicate the limitation in the amount of bandwidth resources that a peer can reserve and a source needed. The flows on edges from peers to sources represent the allocation of resources.

Our first goal is to minimize the total underprovisioning. Given demands and capacities, the resource allocation should ensure that no resource is allocated to an

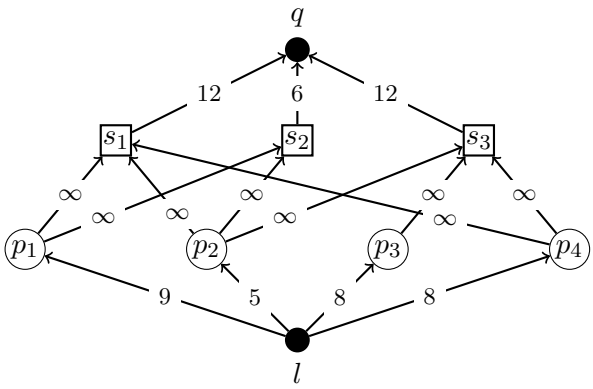


Figure 3.3: Bipartite Flow Network

overprovisioned overlay when it could have been allocated to an underprovisioned one. Let S^+ (respectively S^-) be the set of sources (overlays) with a positive (respectively negative) provisioning. We look for an uplink sharing among the overlays so that the total underprovisioning is minimum. Hence, our first goal is to minimize $\sum_{s \in S^-} |\Delta_s|$.

Proposition 1 *The sum of underprovisioning $\sum_{s \in S^-} |\Delta_s|$ is minimum if and only if the maximum flow is achieved in the flow network.*

Proof. We denote by $f_{s,q}$ the flow on the arc $(s \to q)$. The cut-set between $V \backslash \{q\}$ and $\{q\}$ bounds a flow $|f| = \sum_{s \in S} f_{s,q}$. For each source s, the absolute underprovisioning $|\Delta_s|$ is equal to $D_s - B_s - f_{s,q}$. For a source s in S^+, we have $D_s - B_s - f_{s,q}$ equals to zero because the flow $f_{s,q}$ cannot be greater than $D_s - B_s$. Thus,

$$\sum_{s \in S^-} |\Delta_s| = \sum_{s \in S^-} (D_s - B_s - f_{s,q}) = \sum_{s \in S} (D_s - B_s - f_{s,q}) = A - \sum_{s \in S} f_{s,q}$$

where A is a constant. Minimizing $\sum_{s \in S^-} |\Delta_s|$ is equivalent to maximizing $\sum_{s \in S} f_{s,q}$. Moreover, (i) edges from l to P furnish the system with all capacities $C = \sum_p B_p$, (ii) edges from P to S follow the rule of the bandwidth allocation. Hence, the overall underprovisioning $\sum_{s \in S^-} |\Delta_s|$ is minimized if and only if the flow is maximized.

The max-flow problem has been studied extensively in the literature. The Goldberg-Tarjan algorithm [GT88], which is based on a method called *preflow-push*, is one of the most famous algorithms. The principle is to *push* a flow to the nodes that are estimated to be closer to the sink. The estimation is measured by a *distance label* maintained on each node. This preflow algorithm in a bipartite graph has been especially studied in [AOST94]. In an *unbalanced* bipartite graph, the preflow algorithm can be substantially sped up. A number of works [GGSP95, AGM+05] also designed distributed algorithms. These previous works offer opportunities to develop distributed methods that minimize the total underprovisioning. In particular, these algorithms can be used to implement fast algorithm on a cluster of servers for the management of large-scale systems.

3.4 Cost Function Driven Bandwidth Allocation Strategies

When the system is underprovisioned, there is often more than one maximum flow. So there is often more than one resource allocation that minimize the total underprovisioning. In this section, we show that in the flow network, with dedicated designed cost functions, the minimum-cost maximum flow can drive the distribution of upload bandwidth among underprovisioned sources. In other words, we are able to choose one allocation among all optimal resource allocations so that the chosen one matches a given pre-determined policy.

3.4.1 Minimum-cost Maximum-flow Problem

In the minimum-cost maximum-flow problem, each edge e in the flow network is associated with a certain $cost(e)$. The cost of sending a flow is calculated as $cost(e) \cdot f_e$. The minimum-cost maximum-flow problem is defined as, out of all maximum flows in the flow network, finding the one with the minimum sum of cost $\sum_e cost(e) \cdot f_e$.

It has been shown that the minimum-cost maximum-flow problem can be equivalently converted into the minimum-cost circulation problem by adding an edge from the sink to the source with infinite capacity and large enough negative cost [AMO93]. Then, the latter problem can be solved in polynomial time [Tar85, GT89]. It can also be solved by linear programming because the objective and constraints are all linear.

In our flow network model $N = (V, E)$, we propose to define a cost function for edges in E_3. As the minimum-cost maximum-flow problem aims at minimizing the sum of flow cost, the edges associated with a lower cost will be prioritized in the bandwidth allocation. Consequently, different bandwidth allocation strategies can be applied by using correspondingly defined cost functions. This approach is generic in the sense that various cost functions can be designed, which result in various resource allocations. We present later four distinct bandwidth allocation strategies.

3.4.2 Strategy I: Prioritize Overlay Diversity

The goal of our first strategy is to satisfy the maximum number of overlays. Here an overlay is said to be satisfied if its normalized relative provisioning is positive, or slightly negative. Thus, we define our first strategy as one that minimizes the sum of relative underprovisioning: $\sum_{s \in S^-} |\Delta_s^r|$. Overlays with lower population should be prioritized because they have less demand and are easily satisfied. Thus unpopular overlays should be assigned with smaller cost. Let us define the cost function as:

$$cost_1(e) = \begin{cases} 1, \text{If } e \in E_1 \cup E_2 \\ 1 - \frac{1}{|P_s|}, \text{If } e \in E_3 \end{cases}$$

Then Proposition 2 shows that our first strategy is given by the minimum-cost maximum flow corresponding to $cost_1(e)$.

Proposition 2 *The minimum-cost maximum flow corresponding to $cost_1(e)$ minimizes the sum of relative underprovisioning $\sum_{s \in S^-} |\Delta_s^r|$.*

Proof.

$$\begin{aligned}\sum_e cost_1(e)\cdot f_e &= \sum_{e\in E_1} f_e + \sum_{e\in E_2} f_e + \sum_{e\in E_3}(1-\frac{1}{|P_s|})\cdot f_e\\ &= 2|f_{max}| + \sum_{e\in E_3}(1-\frac{1}{|P_s|})\cdot f_e\\ &= 2|f_{max}| + \sum_{e\in E_3} f_e - \sum_{s\in S}\frac{D_s-B_s-|\Delta_s|}{|P_s|}\\ &= 3|f_{max}| - \sum_{s\in S}\frac{D_s-B_s}{|P_s|} + \sum_{s\in S^-}\frac{|\Delta_s|}{|P_s|}\\ &= A + \sum_{s\in S^-}|\Delta_s^r|\end{aligned}$$

where A is a constant. Thus, minimizing $\sum_e cost_1(e)\cdot f_e$ is equivalent to minimizing $\sum_{s\in S^-}|\Delta_s^r|$.

3.4.3 Strategy II: Prioritize Overlay Popularity

Another reasonable strategy is to prioritize the most popular overlays since they are required by a large number of peers. On the opposite to the first strategy, the number of unsatisfied overlays should be maximized in order to better serve the most popular ones. The second strategy is defined as the one that maximizes the sum of relative underprovisioning $\sum_{s\in S^-}|\Delta_s^r|$. Let us define the cost function as:

$$cost_2(e) = \begin{cases} 1, \text{If } e\in E_1\cup E_2\\ \frac{1}{|P_s|}, \text{If } e\in E_3\end{cases}$$

Then our second strategy is given by the minimum-cost maximum flow corresponding to $cost_2(e)$.

Proposition 3 *The minimum-cost maximum flow corresponding to $cost_2(e)$ maximizes $\sum_{s\in S^-}|\Delta_s^r|$.*

Proof. The proof of Proposition 3 is similar to that of Proposition 2:

$$\begin{aligned}\sum_e cost_2(e)\cdot f_e &= \sum_{e\in E_1} f_e + \sum_{e\in E_2} f_e + \sum_{e\in E_3}\frac{1}{|P_s|}\cdot f_e\\ &= 2|f_{max}| + \sum_{s\in S^-}\frac{D_s-B_s-|\Delta_s|}{|P_s|}\\ &= 2|f_{max}| + \sum_{s\in S}\frac{D_s-B_s}{|P_s|} - \sum_{s\in S^-}\frac{|\Delta_s|}{|P_s|}\\ &= A - \sum_{s\in S^-}|\Delta_s^r|\end{aligned}$$

where A is a constant. Thus, minimizing $\sum_e cost_2(e)\cdot f_e$ is equivalent to maximizing $\sum_{s\in S^-}|\Delta_s^r|$.

3.4.4 Strategy III: Prioritize Fee-Paying Sources

Many service operators have adopted a tiered business model, which prioritizes fee-paying users. With this model, sources can be split into two classes: a premium class consisting of fee-paying sources and a second class consisting of non-paying sources. A third strategy that prioritizes the premium class can be obtained with the following simple cost function.

$$cost_3(e) = \begin{cases} 1, \text{If } e \in E_1 \cup E_2 \\ 1, \{e = (s \to t), s \in Premium\ Class\} \\ 2, \{e = (s \to t), s \in Second\ Class\} \end{cases}$$

3.4.5 Strategy IV: Prioritize User Preference

When watching multiple videos, users pay more attention to a subset of *preferred* videos. This preference can be typically estimated by the window size of each video. For example in Multi-view Internet TV, the major screen size video is the focus of users although channels in small windows do not receive much attention. We define a strategy that prioritizes the overlays that are more frequently watched as preferred videos.

We first define a binary variable r_p^s, which is equal to one if p prefers the video emitted from s, and zero otherwise. Then, the normalized frequency of video s watched as preferred video can be measured as $pref_s = \frac{\sum_{p \in P_s} r_p^s}{|P_s|}$. The value of $pref_s$ is closer to one if it is more frequently watched as preferred video. In order to prioritize preferred videos, overlays with higher $pref_s$ value should be associated with higher provisioning. Thus we can define this strategy as maximizing $\sum_{s \in S^+} pref_s \cdot 0 + \sum_{s \in S^-} pref_s \cdot \Delta_s$. Consequently, the objective of prioritizing user-preferred overlays can be equivalently written as minimizing $\sum_{s \in S^-} pref_s \cdot |\Delta_s|$.

The following cost function can be defined with respect to this objective.

$$cost_4(e) = \begin{cases} 1, \text{If } e \in E_1 \cup E_2 \\ 1 - pref_s, \text{If } e \in E_3 \end{cases}$$

Proposition 4 *The minimum-cost maximum flow corresponding to $cost_4(e)$ minimizes $\sum_{s \in S^-} pref_s \cdot |\Delta_s|$.*

Proof. The proof of Proposition 4 is similar to that of Proposition 2:

$$\begin{aligned} \sum_e cost_4(e) \cdot f_e &= \sum_{e \in E_1} f_e + \sum_{e \in E_2} f_e + \sum_{e \in E_3} (1 - pref_s) \cdot f_e \\ &= 2|f_{max}| + \sum_{e \in E_3} f_e - \sum_{s \in S^-} pref_s \cdot (D_s - B_s - |\Delta_s|) \\ &= 3|f_{max}| - \sum_{s \in S} (D_s - B_s) \cdot pref_s + \sum_{s \in S^-} pref_s \cdot |\Delta_s| \\ &= A + \sum_{s \in S^-} pref_s \cdot |\Delta_s| \end{aligned}$$

where A is a constant. Thus, minimizing $\sum_e cost_4(e) \cdot f_e$ is equivalent to minimizing $\sum_{s \in S^-} pref_s \cdot |\Delta_s|$.

3.4.6 Practical Optimization

Previous strategies have a common drawback: they aim at ensuring a null provisioning to the prioritized overlays, although a slightly negative relative provisioning would have a small impact on the overall quality of experience. To address this problem, we introduce a tunable *tolerable video quality parameter* k and say that an overlay has tolerable video quality if its relative provisioning $|\Delta_s^r|$ is smaller than k.

The demand D_s of an overlay can be interpreted as the amount of upload bandwidth required by s to be provisioned as $\Delta_s = 0$. If the system is very underprovisioned, rather than requiring perfect video quality on each source, we only require tolerable video quality. This can be done by tuning the parameter k. As a consequence, the actual D_s is equal to $|P_s| \cdot (d_s + o_s - k)$.

3.5 Fair Bandwidth Allocation Strategy

The bandwidth allocation strategies discussed in Sec 3.4 prioritize some overlays. However, this can result in significant resource deficits in the unprioritized overlays. Hence, we propose a fair upload bandwidth allocation strategy such that bandwidth is allocated based on overlays demand. Due to the non-linear fairness objective, we solve this optimization problem with the *dual decomposition* method [PC07], which decomposes the original global optimization problem into a set of subproblems. Furthermore, we show how the subproblems can be locally solved by each source-peer pair with a distributed algorithm.

3.5.1 Problem Formulation

In the flow network model, the upload bandwidth allocated from peer p to source s can be seen as a flow along the path $(l \to p \to s \to q)$ with a value of b_p^s. Thus, the problem of fair allocation of upload bandwidth in underprovisioned systems can be formulated as maximizing the flow utility with respect to edges capacities, that is,

$$\max \quad \sum_{s,p \in P_s} D'_s \cdot \log(b_p^s) \qquad b_p^s \geq 0 \tag{3.1}$$

subject to

$$\sum_{s \in G(p)} b_p^s \leq B_p \qquad \forall p \tag{3.2}$$

$$\sum_{p \in P_s} b_p^s \leq D'_s \qquad \forall s \tag{3.3}$$

where $D'_s = D_s - B_s$. The utility function $U(b_p^s) = D'_s \cdot \log(b_p^s)$ expresses the following connotations: (i) the objective function in (3.1) is defined as a strictly concave function to make the problem computationally solvable through convex optimization. (ii) the part $D'_s \cdot \log(b_p^s)$ reflects the *proportional fairness* such that bandwidth is allocated according to each overlay's demand. Constraint (3.2) guarantees that flows on edges in E_1 will not violate the edges' capacity, while constraint (3.3) ensures no such violation on edges in E_3.

3.5.2 Dual Decomposition Solution

This global optimization problem can be solved with the *dual decomposition* method [PC07]. In *optimization theory*, large-scale problems with coupling constraints can be decomposed into subproblems by constraint relaxing. The subproblems can be solved in parallel or sequentially, which reduces the complexity of the original problem. Moreover, if each subproblem involves only local variables, a distributed algorithm can be used. The Lagrangian of (3.1) is

$$\begin{aligned} L(\boldsymbol{\lambda}, \boldsymbol{\mu}) &= \sum_{s,p \in P_s} U(b_p^s) + \sum_p \lambda^p (B_p - \sum_{s \in G(p)} b_s^p) + \sum_s \mu^s (D'_s - \sum_{p \in P_s} b_s^p) \\ &= \sum_{s,p \in P_s} U(b_p^s) - \sum_p \lambda^p \sum_{s \in G(p)} b_p^s - \sum_s \mu^s \sum_{p \in P_s} b_p^s + \sum_p \lambda^p B_p + \sum_s \mu^s D'_s \\ &= \sum_{s,p \in P_s} (U(b_p^s) - \lambda^p b_p^s - \mu^s b_p^s) + \sum_p \lambda^p B_p + \sum_s \mu^s D'_s \end{aligned} \tag{3.4}$$

where $\lambda^p \geq 0$ and $\mu^s \geq 0$ are the dual variables to be minimized in the dual problem and $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$ denote the vectors of λ^p and μ^s, respectively. A solution that satisfies (3.2) and (3.3) can always be found. Thus, Slater's strong duality constraint qualification holds, and the original constrained convex optimization problem can be solved via the dual problem (3.4) [BV04]. The master dual problem (3.5) is

$$\min \quad \phi(\boldsymbol{\lambda}, \boldsymbol{\mu}) = \sum_{s,p \in P_s} \phi_{s,p}(\lambda^p, \mu^s) + \boldsymbol{\lambda}^T \boldsymbol{B} + \boldsymbol{\mu}^T \boldsymbol{D'} \tag{3.5}$$

for the dual variables $\boldsymbol{\lambda} \succeq 0$ and $\boldsymbol{\mu} \succeq 0$ where $\boldsymbol{B}$ and $\boldsymbol{D'}$ denote the vectors of B_p and D'_s and

$$\phi_{s,p}(\lambda^p, \mu^s) = \sup_b (U(b_p^s) - \lambda^p b_p^s - \mu^s b_p^s) \qquad \forall s,\ p \in P_s$$

is the dual function obtained as the optimal value of the Lagrangian solved in (3.6).

$$\max \quad U(b_p^s) - \lambda^p b_p^s - \mu^s b_p^s \qquad \forall s,\ p \in P_s \tag{3.6}$$

The subgradient method is a convenient and general approach to iteratively update the dual variables into the optimum value. Let $\hat{b_p^s}(\lambda^p, \mu^s)$ the optimal solution of problem (3.5) with given λ^p and μ^s. Given (λ^p, μ^s), $\hat{b_p^s}(\lambda^p, \mu^s)$ is unique due to the strict concavity of the utility function, and it can be calculated as:

$$\hat{b_p^s}(\lambda^p, \mu^s) = \frac{D'_s}{\lambda^p + \mu^s}$$

Let $(\bar{\lambda^p}, \bar{\mu^s})$ be a feasible dual solution, then the subgradient of $\phi_{s,p}$ can be derived

Algorithm 1: Source s at round $t > 0$

1: Wait for $\hat{b_p^s}(t-1)$ from $p \in P_s$
2: Update $\mu^s(t)$ with $\mu^s(t-1)$ and $\hat{b_p^s}(t-1)$ according to (3.9)
3: **for** $p \in P_s$ **do**
4: Send $\mu^s(t)$ to p

Algorithm 2: Peer p at round $t > 0$

1: Update $\lambda^p(t)$ with $\lambda^p(t-1)$ and $\hat{b_p^s}(t-1)$ according to (3.8)
2: Wait for $\mu^s(t)$ from $s \in G(p)$
3: **for** $s \in G(p)$ **do**
4: Solve problem (3.6) with $\lambda^p(t)$ and $\mu^s(t)$ and obtain $\hat{b_p^s}(t)$
5: Send $\hat{b_p^s}(t)$ to s

as:

$$\begin{aligned}
\phi_{s,p}(\lambda^p, \mu^s) &= \sup_b (U(b_p^s) - \lambda^p b_p^s - \mu^s b_p^s) \\
&\geq U(\hat{b_p^s}(\bar{\lambda^p}, \bar{\mu^s})) - \lambda^p \hat{b_p^s}(\bar{\lambda^p}, \bar{\mu^s}) - \mu^s \hat{b_p^s}(\bar{\lambda^p}, \bar{\mu^s}) \\
&= U(\hat{b_p^s}(\bar{\lambda^p}, \bar{\mu^s})) - (\lambda^p - \bar{\lambda^p})\hat{b_p^s}(\bar{\lambda^p}, \bar{\mu^s}) - (\mu^s - \bar{\mu^s})\hat{b_p^s}(\bar{\lambda^p}, \bar{\mu^s}) \\
&\quad - \bar{\lambda^p}\hat{b_p^s}(\bar{\lambda^p}, \bar{\mu^s}) - \bar{\mu^s}\hat{b_p^s}(\bar{\lambda^p}, \bar{\mu^s}) \\
&= \phi_{s,p}(\bar{\lambda^p}, \bar{\mu^s}) - (\lambda^p - \bar{\lambda^p})\hat{b_p^s}(\bar{\lambda^p}, \bar{\mu^s}) - (\mu^s - \bar{\mu^s})\hat{b_p^s}(\bar{\lambda^p}, \bar{\mu^s})
\end{aligned} \tag{3.7}$$

From (3.7), the subgradient $s(\boldsymbol{\lambda}, \boldsymbol{\mu}) = -\hat{b}(\boldsymbol{\lambda}, \boldsymbol{\mu})$ where $\hat{b}(\boldsymbol{\lambda}, \boldsymbol{\mu})$ is the optimum solution of problem (3.5) for a given $(\boldsymbol{\lambda}, \boldsymbol{\mu})$. Thus, the dual variables can be updated by the following iteration:

$$\lambda^p(t+1) = [\lambda^p(t) - \alpha(B_p - \sum_{s \in G(p)} \hat{b_p^s}(\lambda^p(t), \mu^s(t)))]^+ \qquad \forall p \tag{3.8}$$

$$\mu^s(t+1) = [\mu^s(t) - \alpha(D'_s - \sum_{p \in P_s} \hat{b_p^s}(\lambda^p(t), \mu^s(t)))]^+ \qquad \forall s \tag{3.9}$$

where t is the iteration index, α is a small enough positive step size, and $[\cdot]^+$ is the nonnegative orthant projection.

3.5.3 Distributed Algorithm

The E_1 edge price λ^p can be locally updated by each peer p by flows pass over p, that is, the allocated upload bandwidth from p to its subscribed sources, while μ^s can be updated by each source in the same way. In this section, we discuss the algorithm generated by dual decomposition which solves the global optimization problem in (3.1) locally by message passing between sources and peers coupled by constraint (3.2) and constraint (3.3).

The algorithm consists of two phases: an initialization phase and a running phase. Initially, we set $t = 0$. Since the subgradient method starts from any given $(\boldsymbol{\lambda}, \boldsymbol{\mu})$ and converges to the optimal solution, the initial value of $(\boldsymbol{\lambda}(0), \boldsymbol{\mu}(0))$ is randomly generated. On the source side, each source randomly chooses $\mu^s(0)$ such that $\mu^s(0) > 0$ and broadcasts $\mu^s(0)$ to its peers. On the other hand, each peer randomly chooses a positive $\lambda^p(0)$ and calculates $\hat{b_p^s}(\lambda^p(0), \mu^s(0))$ with respect to $(\lambda^p(0), \mu^s(0))$, and then sends $\hat{b_p^s}(\lambda^p(0), \mu^s(0))$ to the corresponding sources. Then, the algorithm goes into the iterative phase until it converges to the optimal solution. Each iteration is detailed in Algorithm 1 and Algorithm 2.

The algorithm requires message passing between peers and their subscribed sources. For practical relevance, the source-to-peer information can be piggy backed using the video chunk messages, while for the reverse direction, the message integration mechanism can be used on upstream peers. According to the subgradient method, the dual variables $(\boldsymbol{\lambda}, \boldsymbol{\mu})$ will converge to the optimum value with sufficiently small step size α. Moreover, the original primal problem (3.1) can be equivalently solved by the dual problem (3.4), the primal variable also converge into the optimum value.

3.6 Implementation and Practical Details

We provide implementation details to show the practical relevance of our theoretical framework.

3.6.1 Overall Architecture and Peer Dynamics

The main functions related to bandwidth allocation are implemented in a global server called *P2PServer* (as Figure 3.1). Every peer sends a report to P2PServer that contains an estimation of the available bandwidth. Then, P2PServer computes the bandwidth allocation and send it to peers. With the deployment of the P2PServer, centralized bandwidth allocation algorithms, for example, the maximum flow based algorithms presented in Section 3.3 and Section 3.4 can be implemented to calculate the optimum bandwidth allocation.

This approach is feasible in a static environment. However, all P2P video applications face the problem of peer churn and the available bandwidth can vary between two measurements. To cope with system dynamics, we followed the approach proposed in [WXR09, WXR11]. The system time is cut into *sessions*, and the computation is done periodically. The computation involves only peers and sources that exist in the system at the starting point of that session. During the session, new arriving peers allocate their upload bandwidth according to some predefined strategy (e.g., equal allocation to the subscribed overlays). The capacity of the system to handle the dynamic behavior of peers only depends on the choice of session length. We evaluated the received video quality for different session lengths and peer churn rates in Section 3.7. They suggest that a delay of one minute between two re-computations is a reasonable choice.

nb. peers	1,000	5,000	10,000	50,000	100,000
time (in sec)	0.005	0.086	0.311	7.455	31.887

Table 3.2: Computation time for minimum-cost maximum-flow algorithms.

3.6.2 Peer-Server Communication Overhead

We are concerned about the traffic generated by the peer-server communication. However, this traffic has to be seen in light of the huge amount of data needed for the live video streams. For example, if the average number of videos watched by a peer is three, two bytes are used to specify the upload bandwidth reserved to an overlay, and the bandwidth allocation is recomputed every minute, then P2PServer needs to transmit 0.8 bps per peer. If we consider in addition the 54 bytes for the TCP/IP/Ethernet packet overhead, then 0.8 Mbps server upload bandwidth would be needed for 100,000 users. Similarly, if the report sent by a peer to P2PServer includes four bytes to specify the estimated upload bandwidth and four bytes for the peer ID, then only 0.826 Mbps server download bandwidth would be needed for 100,000 users. It shows that, from a network standpoint, the system can be implemented without much fear for scalability.

3.6.3 Algorithm Computation Time

We also studied the scalability in terms of computation. We measured the exact computation time of the *preflow* maximum flow algorithm and a scaling approximation minimum-cost flow algorithm [Gol97]. The measurement was done on a typical server (2×4 cores Intel(R) Xeon(R) 2.67GHz CPUs). The results are average value of 5 runs (Table 3.2). The number of peers increases from 1,000 to 100,000. For each instance, the number of sources was set to 10% of the population. The number of channels watched by a peer was randomly chosen between 1 and 5.

Our measurements demonstrate that a practical implementation of the minimum-cost maximum-flow algorithm can compute the resource allocation for very large instances (100,000 peers and 10,000 sources) in reasonable time. They also confirm that recompute every minute is a reasonable choice. The low peer to server communication overhead and the fast resource allocation algorithm show the feasibility of managing a centralized server to recompute periodically the optimal resource allocation in a dynamic environment.

3.7 Evaluation

To evaluate the performance of the system, especially in dynamic environments, we conducted two sets of simulations: a static scenario simulation and a dynamic scenario simulation. In the static scenario, we calculate the optimal bandwidth allocation of different algorithms for a set of peer-source configurations. This can be seen as calculating the optimal bandwidth allocation of a snapshot of a dynamic system. The system provisioning was changed from an overprovisioned state into an under-

provisioned state by increasing the video bit rate. We show that with reasonable multioverlay structure (number of overlays and overlay popularity), the system can be underprovisioned, which confirms our intuition that underprovisioning is a critical issue for multioverlay systems. The aim of the static scenario simulation is to reveal the behavior of the algorithms and compare them with respect to the video quality experienced by the peers. In the dynamic scenario, we introduced peer churn. The system periodically recomputes the optimal bandwidth allocation every Δt time with respect to the configuration at that moment. We defined three sub-scenarios: a *join-only* scenario, a *leave-only* scenario and a *fully-dynamic* scenario. Then, we evaluated the performance of the dynamic system under different peer churn rates and investigated the impact of the recomputation time interval Δt.

The following algorithms were evaluated:

- **DAC** [WXR09]: It fairly allocates upload bandwidth based on overlays' demands, but it is blind to the provisioning of overlays.
- *Diversity-based* **(Db)**: It corresponds to the strategy that prioritizes unpopular overlays (Section 3.4.2).
- *Popularity-based* **(Pb)**: It corresponds to the strategy that prioritizes popular overlays (Section 3.4.3).
- *Improved diversity-based* **(IDb)** and *Improved popularity-based* **(IPb)**: Versions of **Db** and **Pb** obtained by introducing the tolerable video quality parameter k as 50 kbps for underprovisioned system(Section 3.4.6).
- *Payment-based* **(Pa)**: It prioritizes premium sources (Section 3.4.4).
- *Preference-based* **(Pr)**: It prioritizes more frequently preferred overlays (Section 3.4.5). Each peer randomly chooses a video as its preferred one.
- *Fairness-based* **(Fb)**: It allocates the upload bandwidth based on overlays' demands while being aware of the provisioning of overlays (Section 3.5).
- *Naive* **(Nai)**: It equally allocates the upload bandwidth to the overlays it belongs.

3.7.1 Simulator Platform

To evaluate the proposed bandwidth allocation algorithms, we built a two-level simulator platform as shown in Figure 3.4. The first level simulates an MMOG video sharing system in which users form friendships and share live videos of their game to their friends. Although we simulated MMOG video sharing services, we believe that the results are applicable to any social network-based video sharing service because our simulation model is based on rules that reflect human nature (Pareto rule). In order to conduct realistic simulations, the peer-source watching relationship (which users publish videos and who are the receivers of these videos) is established based on a model drawn from real measurements. Upon the peer-source relationship, the bandwidth allocation algorithms act to allocate users upload bandwidth.

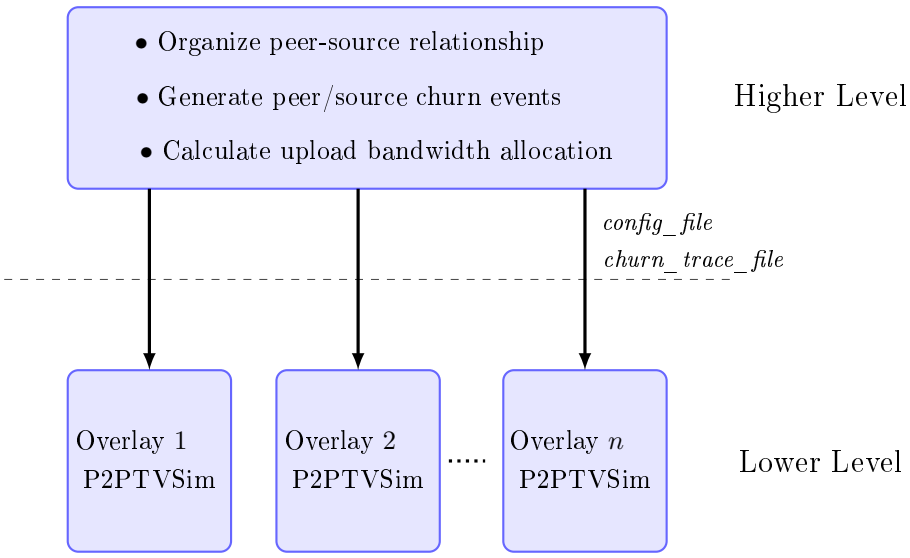


Figure 3.4: The two level simulator platform

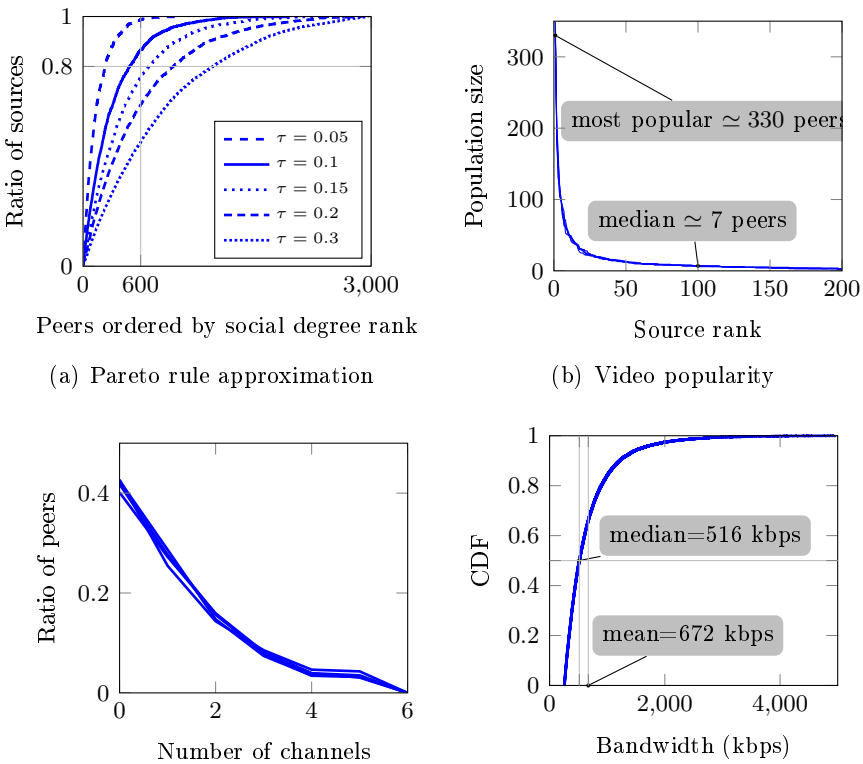
The second level simulates the chunk diffusion of each overlay with *P2PTVSim* [P2P]. P2PTVSim is a peer-to-peer TV simulator which simulates a flow of video chunks from one source to a set of peers. Peers are interconnected in a mesh-based overlay where each peer randomly chooses a number of peers as neighbors. Peers use the latest-best chunk scheduling strategy, i.e., peers push the latest useful chunk to their neighbors according to their upload bandwidth. For simplicity, chunk losses are only caused by the lack of bandwidth. With the same configuration, a set of peers with low upload bandwidth have longer chunk delays and more chunk losses compared to a set of high-bandwidth peers.

The current P2PTVSim simulator is far away from simulating a dynamic video streaming system. We modified the P2PTVSim simulator to support well controlled peer dynamics by loading peer churn events from a peer churn trace file. Thus, the higher level simulator generates, for each overlay, a simulation configuration file, as well as a trace file which contains peer churning events and bandwidth allocation information. Then, the lower level takes these two files as input and simulates the video chunk diffusion in each overlay.

Finally, from the results of P2PTVSim, we also evaluated the video quality at each receiving peer by measuring the average luminance peak signal-to-noise-ratio (PSNR). The PSNR is a standard video quality metric computed as:

$$PSNR(dB) = 10 \log_{10} \frac{255^2}{MSE}$$

where MSE is the mean squared error between the original frame and the reconstructed frame.



(a) Pareto rule approximation

(b) Video popularity

(c) Number of watching videos

(d) Peer upload bandwidth

Figure 3.5: Simulation Setting

3.7.2 Simulation Setting

Peer upload bandwidth followed a log-normal distribution with parameters $\mu = log(380)$ and $\sigma = 0.8$, ranging from 256 kbps to 5 Mbps (Figure 3.5(d)). For the static scenario, the video bit rate varied from 240 kbps to 330 kbps in steps of 30 kbps. The overall system changed from an overprovisioned state into an underprovisioned one. For the dynamic scenario, the video bit rate was set to 330 kbps. The overhead was set to 50 kbps in both cases.

To establish a realistic peer-source watching relationship, there are three key questions to answer:

- *What is the basic user relationship to support video sharing?* In the context of user-generated content, video sharing is likely to occur between friends. Thus, the in-game user social network is the basic framework to support user interaction. Shen and Iosup [SI11] observed that, on average, Xfire users have 60 friends, and about 15% of the players have more than 100 friends. These figures are smaller than observations on Facebook (130 friends on average).

- *Who are the video sources?* Videos are spontaneously published by users, and especially by socially active users [WBS+09]. We defined a mechanism where the probability that a user becomes a source depends on the number of its friends.

- *Who are the receivers of each video?* This directly affects the video popularity distribution. A user decides to watch a video based on its popularity (attractiveness). Measurements in [SI11, KSC+12] show that the popularity of live game videos follows a power-law distribution.

We then build a realistic simulation model in which the peer-source watching relationship is formed in two phases: after the selection of sources, the friends of each source decide to watch this video or not based on its attractiveness. From 100 trace files collected from the Facebook network [TMP11], we selected the social network of "Smith College" as the in-game social network. This network is similar to the one in [SI11]. It consists of 2,970 peers. The average social degree is 65.4, and 20.24% of the peers have more than 100 friends.

We modeled the publication of video according to the Pareto 80-20 rule—80% of videos are published by 20% of the most active users. The measurement in [WBS+09] confirms our assumption that user activity is strongly related to user degree in the social graph (number of friends). We then chose the user social degree as the major criterion of activity and formalized a model in which users decide to publish a video based on a probability related to their social graph degree. That is, a user decides to publish a video based on a probability equal to $e^{-\frac{i}{\tau N}}$, where i denotes its rank in terms of social degree in decreasing order, N denotes the total number of users, and τ is a parameter. As shown in Figure 3.5(a), the Pareto rule is followed when τ takes on values between 0.1 and 0.15. With $\tau = 0.1$, on average, 214 videos were published with five different random seeds.

We modeled the peer watching decision in a way that the resulting video popularity follows Zipf's law. Assume that sources are ranked by their social degree (size of potential audience) in decreasing order, and s_i is the ith source with n_{s_i} denoting its number of friends. We associated to each source s_i an *attractiveness index* α_i, which is equal to the probability with which its friends decide to watch this video. The attractiveness index α_i was calculated such that the channel population follows Zipf's law: $\frac{n_{s_i}}{n_{s_1}} \cdot \alpha_i \sim i^{-b}$. Figure 3.5(b) shows the number of videos and corresponding popularity generated by $\tau = 0.1$ and $b = 0.85$. Due to the limitation of human attention, we set the maximum number of videos a peer can simultaneously watch to five. Figure 3.5(c) shows the corresponding number of watched videos distribution.

We used a 600-frame video by concatenating 300 frames of the Foreman sequence and 300 frames of the Mother and Daughter sequence. Both sequences are in Common Intermediate Format (CIF) format and have a frame rate of 30 fps. The video was compressed with the H.264/AVC encoder at bit rates ranging from 240 to 330 kbps using the H.264 high profile. We used the Group of Pictures (GOP) structure IBBPBBPBBP (10 frames per GOP). Each chunk corresponds to one GOP, so each GOP is played back independently of the other chunks. At the receiver side, we used the standard frame copy error concealment technique to deal with lost frames. With this technique, the last frame of the last decoded GOP is used to represent all frames of a missing chunk.

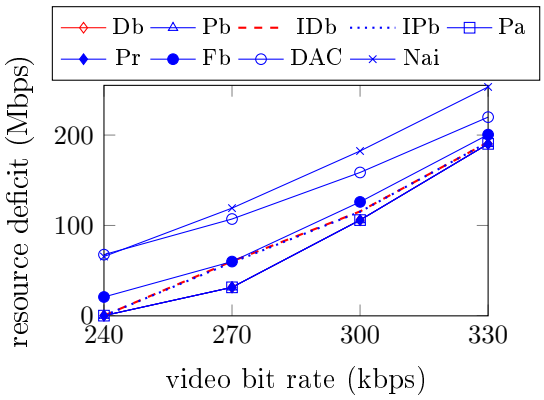


Figure 3.6: Total amount of missing upload capacity.

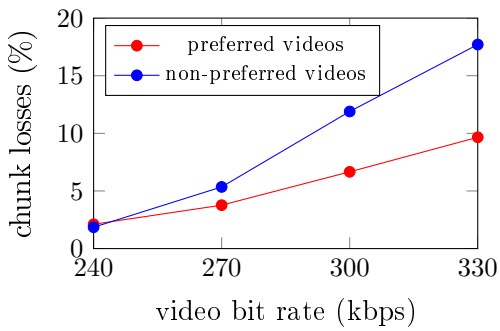


Figure 3.7: Average chunk losses of preferred and non-preferred videos per peer for Preference-based.

3.7.3 Static Scenario Simulation

Provisioning Results

Our simulation results show the median result obtained from five runs. In Figure 3.6, the resource deficit shows the total underprovisioning $\sum_{s \in S^-} |\Delta_s|$. The maximum-flow based approaches (**Db**, **Pb**, **Pa** and **Pr**) minimize the total underprovisioning. For video bit rate 240 kbps, the system is overprovisioned, and **Db**, **Pb**, **Pa**, **Pr**, **IDb** and **IPb** have zero total underprovisioning, which means that every overlay is well provisioned.

We observe that both **IDb** and **IPb** can have a slightly larger underprovisioning than Db and Pb. Indeed, the reduced overlay demand can perturb the computation of a maximum flow in the flow network. Moreover, constant k was set to 50 kbps although the video bit-rate varies, which explains why the difference between curves is not constant.

Fb has a larger total underprovisioning than the maximum-flow based approaches because constraints (3.2) and (3.3) cannot guarantee the maximum flow. Then, peers equally allocate the unallocated bandwidth to all subscribed overlays.

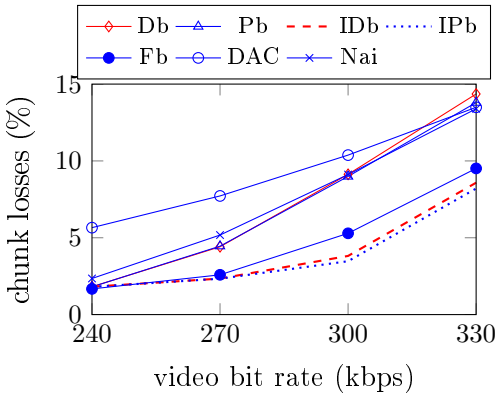


Figure 3.8: Total chunk losses.

The overlay demand follows Zipf's law. Because **DAC** allocates upload bandwidth to overlays according to proportional fairness, the most demanding overlays are highly overprovisioned while leaving the remaining overlays more underprovisioned. Consequently, the total underprovisioning is higher. When the system goes into a more underprovisioned state, the difference between **DAC**, **Fb**, and the maximum-flow based approaches decreases. Because **DAC** always allocates most resources to the high demanding overlays, when the system is very underprovisioned, even these overlays' demands cannot be fulfilled.

We expect that all algorithms (except **Nai**) have the same underprovisioning in very underprovisioned system. However, it is worthless to investigate this situation, because the performance of the system is naturally very poor. Besides, **Nai** has the highest total underprovisioning.

P2PTVSim Results

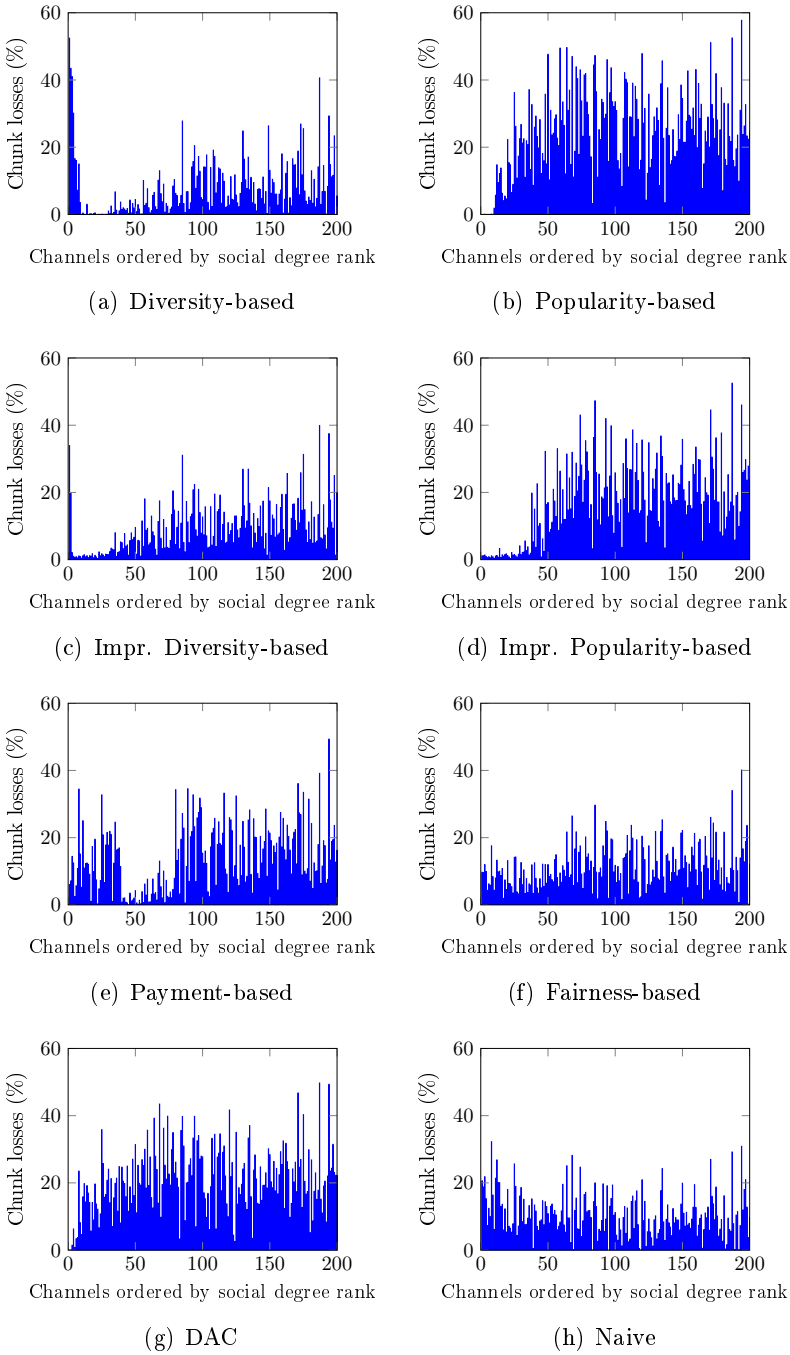
We first show how the bandwidth is allocated among overlays by showing the percentage of lost chunks per peer in overlays ordered according to popularity for video bit rate 330 kbps (Figure 3.9(a) to Figure 3.9(h)). A low chunk loss means an overlay is well provisioned.

In **Db**, channels ranked 10 to 30 have high video quality, while the ten most popular overlays have high chunk losses. On the contrary, **Pb** produces high video quality for the first ten most popular overlays. Some channels ranked after 50 have high chunk losses for **Db**. These are the overlays with few viewers, thus cannot receive enough resources collectively.

We now compare **IDb** to **Db** and **IPb** to **Pb**. In **IDb**, peers in the most popular channels lose fewer chunks, and more channels experience high-quality videos. Similarly, in **IPb**, the first 40 overlays experience relatively few chunk losses.

For the evaluation of **Pa**, the premium class channels were chosen to be the overlays ranked from 40 to 80. These overlays have far fewer missing chunks than the other overlays.

When we compare **Fb** to **DAC**, we observe that **Fb** has fewer chunk losses than



(a) Diversity-based (b) Popularity-based

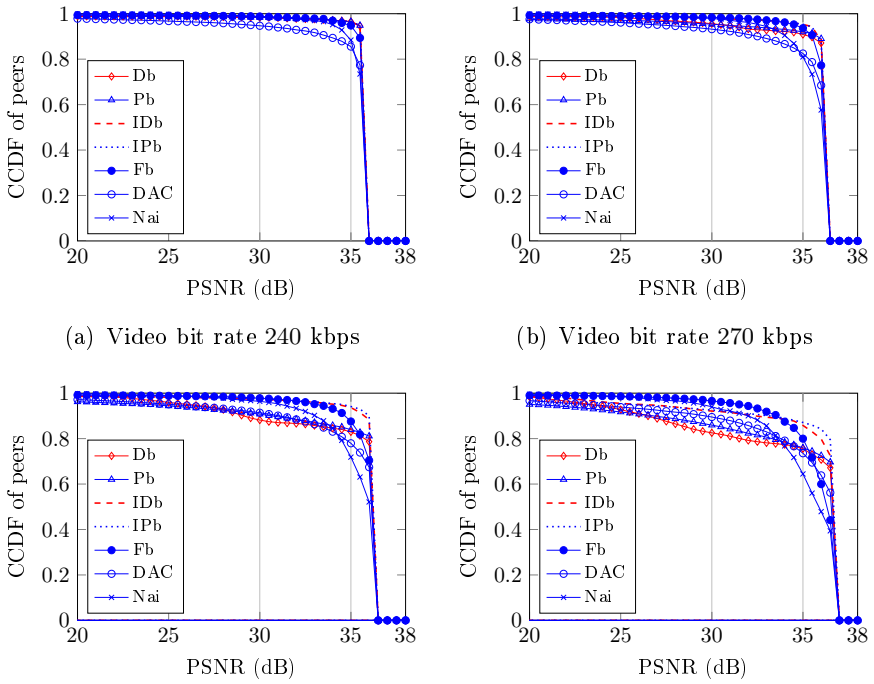
(c) Impr. Diversity-based (d) Impr. Popularity-based

(e) Payment-based (f) Fairness-based

(g) DAC (h) Naive

Figure 3.9: Average chunk losses in overlays. Overlays are ordered by popularity. The video bit rate is 330 kbps.

DAC. This is because more resources from the overprovisioned high demand overlays are allocated to the underprovisioned ones. The **DAC** algorithm allocates most of the upload bandwidth to the first overlays due to their large demands, making these overlays enjoy no chunk loss while leaving the remaining overlays with high chunk



(a) Video bit rate 240 kbps

(b) Video bit rate 270 kbps

(c) Video bit rate 300 kbps

(d) Video bit rate 330 kbps

Figure 3.10: PSNR results.

losses.

The results obtained by **Nai** may seem better than those of the other strategies. The chunk losses of **Nai** are almost uniform. In fact, this strategy takes the worst, as we will see later, all overlays are unsatisfied.

We do not show such a figure for **Pr** because the well provisioned overlays are the most preferred ones, which highly depend on user choice. Instead, we show in Figure 3.7 the average chunk losses of preferred videos and non-preferred videos for every peer. When the system is overprovisioned, peers receive the same video quality for both types of videos. As the resource deficits increase, this strategy ensures that, globally, peers receive higher quality for their preferred videos.

Figure 3.8 shows the total chunk losses in the system. We do not present results for the **Pa** and **Pr** algorithms because their performance highly depends on some uncertain user choices. When the system is overprovisioned, **DAC** has higher chunk losses because the low demanding overlays can be underprovisioned. When the system is in an underprovisioned state, the improved max-flow approaches (**IDb** and **IPb**) and **Fb** have the lowest total chunk losses.

PSNR Results

We now provide objective video quality results by measuring the average luminance PSNR at each receiving peer. To illustrate the distribution of the PSNR, we show the complementary cumulative distribution function (ccdf) of peers (Figure 3.10). We

distinguish three sets of allocations: (1) the maximum-flow allocations (**Db** and **Pb**), (2) their improved counterparts (**IDb** and **IPb**), and (3) **Fb**, **DAC** and **Nai**. We do not present results for the **Pa** and **Pr** algorithms for the reasons mentioned earlier in Section 3.7.3.

When the video bit rate is 240 kbps (Figure 3.10(a)), the system is overprovisioned. The maximum-flow based approaches ensure that almost all peers enjoy high video quality because every overlay is well provisioned. The **DAC** algorithm allocates resources to the most demanding overlays, making some peers experience poor video quality although the system is overprovisioned. The **Fb** algorithm has slightly worse results than the maximum-flow based approaches because the maximum flow cannot be ensured. However, compared to **DAC**, **Fb** balances resources from the overprovisioned overlays into less demanding underprovisioned ones, making fewer peers have poor video quality.

As the video bit rate increases, the system changes into an underprovisioned state. We discuss in detail the results for video bit rate 330 kbps (Figure 3.10(d)). A significant proportion of peers experience high video quality: 70% (respectively 80%) of peers for the maximum-flow allocations (respectively the improved ones). For **DAC** and **Nai**, this percentage is 40% and 60%, respectively. For the maximum-flow allocations, 5% of peers have poor video quality (less than 20 dB) due to their extreme bandwidth allocation strategy. The curve of **Fb** reflects the "fair" strategy: for almost all peers, the PSNR is between 30 dB and 37 dB. This strategy makes sure that no overlay is extremely well provisioned or bad provisioned.

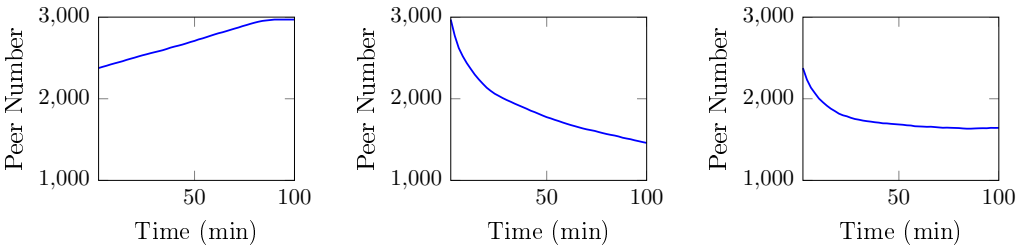
3.7.4 Dynamic Scenario Simulation

We also conducted a set of dynamic scenario simulations by introducing peer churn. When a user enters the system, it first decides to become a source or not based on the mechanism stated in Section 3.7.2. If it decides to become a source, it emits videos during its whole playing session. During its playing session, if some friends emit a video, the peer decides to watch the video or not based on the video attractiveness index. The system periodically recomputes the bandwidth allocation every Δt minutes. For peers that enter the system during two consecutive calculations, the upload bandwidth is equally allocated to the subscribed overlays. We simulated a dynamic system with 100 minutes simulation time.

Peer churn is modeled with the following parameters.

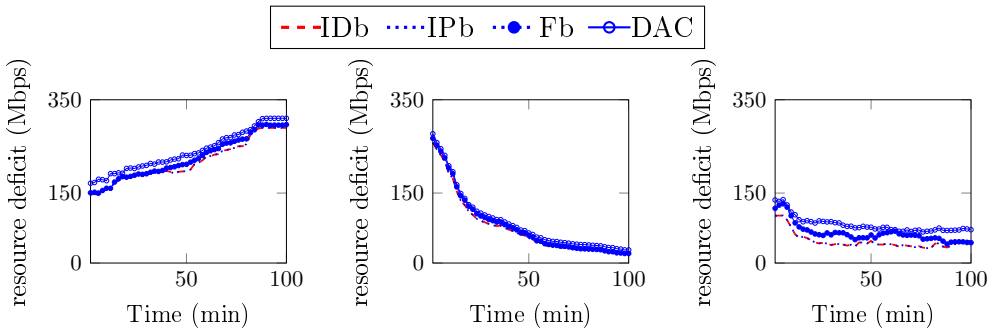
- ratio of peers initially in the system: γ_{ini}.
- peer playing session length distribution: We followed the measurements in [FBS07] and modeled the playing session length by a *Weibull* distribution with $scale = 11.7$, $shape = 0.456$, and minimum session length 30 s.
- peer inter-arrival time distribution: Peers inter-arrivals can be modeled by a Poisson process within a short period of time (one hour) [CHL06]. We generated short-time simulations (100 min), thus, we modeled peer arrivals as a Poisson process with arrival rate λ.

We conceived three scenarios:



(a) Join-only scenario (b) Leave-only scenario (c) Fully-dynamic scenario with $\lambda = 0.15$

Figure 3.11: Number of peers in the system.



(a) Join-only scenario (b) Leave-only scenario (c) Fully-dynamic scenario with $\lambda = 0.15$

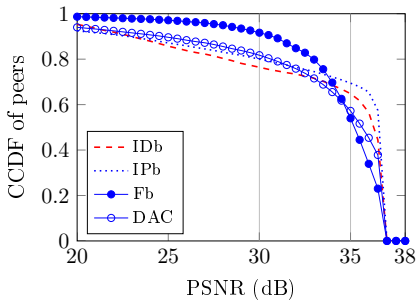
Figure 3.12: Upload bandwidth deficit in the system.

- a *join-only* scenario with $\gamma_{ini} = 0.8$ and $\lambda = 0.15$. The player playing session length is set to the simulation time, thus, peers only join and do not leave the system.
- a *leave-only* scenario in which peers spend their lifetime in the system and then leave. Consequently, we set γ_{ini} to 1, and λ to 0.
- a *fully-dynamic* scenario with $\gamma_{ini} = 0.8$ in which peers join and leave. We extensively studied this scenario using two re-computation time periods ($\Delta t =$ 2 min and $\Delta t = 5$ min) and two peer churn rates ($\lambda = 0.15$ and $\lambda = 0.05$).

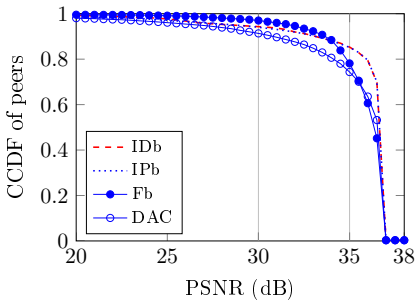
In Figure 3.11, we show how the number of peers evolves in the three scenarios. In join-only, the number of peers steadily increases to the total number of peers. The change of the peer leaving rate for leave-only is due to the *Weibull* playing session distribution. For fully-dynamic, after 20 min, the number of peers is quite stable.

System Provisioning

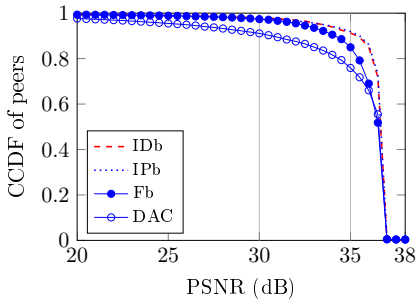
We present the system provisioning for the three scenarios in Figure 3.12. The system provisioning is consistent with the number of peers (Figure 3.11). In the join-only scenario, as the peer number increases, the upload bandwidth deficit increases from



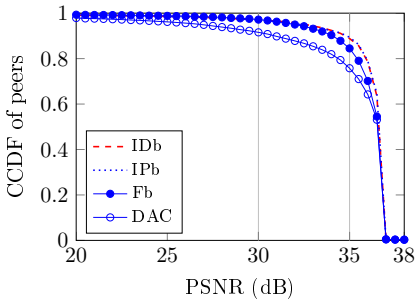
(a) Join-only $\Delta t = 2$



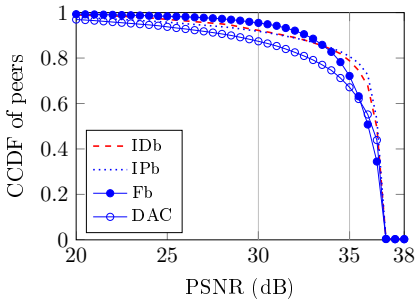
(b) Leave-only $\Delta t = 2$



(c) Fully-dynamic $\Delta t = 2$, $\lambda = 0.15$



(d) Fully-dynamic $\Delta t = 5$, $\lambda = 0.15$



(e) Fully-dynamic $\Delta t = 2$, $\lambda = 0.05$

Figure 3.13: PSNR results.

about 150 Mbps to around 300 Mbps. In the leave-only scenario, the system becomes less underprovisioned from around 300 Mbps to less than 50 Mbps. In the fully dynamic scenario (Figure 3.12(c)), after the sharp decrease at the beginning, the upload bandwidth deficit remains around 50 Mbps. The curves of different algorithms are consistent with those obtained in the static simulation (Figure 3.6).

PSNR Results

Figure 3.13 shows the PSNR results. The results are similar to the static ones. They demonstrate that recomputing the bandwidth allocation on a periodic basis works well. The results in Figure3.13(a) are worse than those in the other figures because the total under-provisioning is greater in the join-only scenario. In all scenarios, the

improved maximum-flow strategies have the best performance; they allow more peers to watch the video at more than 35 dB.

Impact of the re-computation time period Δt. Figure 3.13(d) shows the PSNR results for $\Delta t = 5$ min. Compared to the results obtained with $\Delta t = 2$ min (Figure 3.13(c)), the percentage of peers with a PSNR higher than 35 dB decreases only slightly (2.5% for **IDb** and **IPb**, 0.5% for **Fb** and 0.1% for **DAC**). This indicates that re-computing the allocation after a few minutes does not affect the quality of the allocations significantly.

Impact of the peer churn rate λ. Figure 3.13(e) shows the results for a higher peer churn rate ($\lambda = 0.05$). Compared to the results with lower churn rate and the same re-computation time period Δt (Figure 3.13(c)), the percentage of peers with a PSNR higher than 35 dB decreases (about 11.1% for **IDb** and **IPb**, 12.8% for **Fb** and 8.7% for **DAC**). It shows that, with higher peer churn rate, the system performance degrades but still maintains an acceptable level.

3.8 Conclusion

The problem of underprovisioned multioverlay P2P systems has not received enough attention. This Chapter makes several contributions to this open research topic. The first contribution was to devise strategies that minimize the resource waste when upload bandwidth resources are allocated to overprovisioned overlays although they could be allocated to underprovisioned ones. The second contribution was to design strategies that balance the resource deficit over the overlays. We defined several bandwidth allocation strategies that are optimal in terms of minimizing resource deficit. One of the original contributions of our work was to show that these strategies can be built as solutions of minimum-cost maximum-flow problems. We also designed a fair bandwidth allocation strategy that is aware of the provisioning of each overlay, and a corresponding distributed algorithm was developed. Finally we evaluated our algorithms through extensive simulations. We showed in particular that the introduction of a small tolerable video degradation for the most provisioned overlays and the provisioning-aware fairness based strategy can have a significantly positive impact on the whole system. Furthermore, through a set of dynamic scenario simulations, we showed that the periodic recomputation of an optimal bandwidth allocation can cope with system dynamics.

A part of the work presented in this Chapter have been published in [LAB+12]. We are also considering submitting the whole work as a journal paper. This work is involved in the CNG (Community Network Game) project [CNG], funded by the European Commission's Seventh Framework Programme (FP7, 2007-2013) under the grant agreement no. ICT-248175. The project developed an MMOG user communication tool that enables users to screencast their games either to the members of their guild or to a broader audience. This communication tool has been integrated in The Missing Ink MMOG [mis]. Figure 3.14 shows a snapshot of the screen of an Alpha tester during an online live streaming session. The tester is using the P2P system to

Figure 3.14: CNG user communication tool

watch three live streams of its friends' screens while playing the game. We also made several contribution to the work in the project [ABH+11, BAD+12].

For possible future work, decentralized system and distributed algorithms are directions deserve further investigation. Especially, since the improved strategies, which allow video quality degradation in the prioritized overlays, appear to be attractive, we believe a reasonable further step of the work including designing distributed algorithms that optimizes these strategies in a dynamic environment.

Chapter 4

Discretized Streaming Model for Live Rate Adaptive Streaming

4.1 Introduction

We have illustrated the traditional architecture for the delivery of live video streams in the Internet in Figure 2.5 in Section 2.3. Three main actors are involved in this process: the *content provider*, the *CDN provider* and the *end users*. End users form a heterogeneous population, who consume the videos on a growing range of devices through different types of connections. Content providers are adopting rate adaptive streaming technologies (for example, the DASH protocol) to enhance its ability to serve this heterogenous population. CDNs handle the majority of video traffic on behalf of content providers. However, to the best of our knowledge, currently there is no work that investigates the live rate adaptive streams delivery in the CDN infrastructure. In this chapter, we introduce our first-step theoretical achievement on this domain: the *discretized streaming model* for the general problem of live rate adaptive streaming, and a fast near-optimum algorithm for a specific application scenario.

Live streaming requires the CDN provider to *provision* a delivery infrastructure in advance, *i.e.* to make sure that equipments in the CDN infrastructure are able to transmit streams from source servers to edge servers, and then to the end users. As a result, the sharp growth in the volume of video traffic, and the widely deployed bandwidth consuming rate adaptive streaming technique impose a novel challenge for today's CDN providers: to cope with the *underprovisioned* CDN infrastructures. As a result, the goal of the CDN providers for live streaming is maximizing the throughput of the CDN delivery network, under the constraint of outbound capacity of CDN equipments [ASV11, AMM+11, ZAB+12].

We have introduced the *streaming capacity problem* in Section 2.3.4. Here we briefly recall, and discuss more details about this most relevant work to our study. This problem aims to determine the maximum deliverable bit rate of a P2P network. Some algorithms, mostly based on network coding, obtain near-optimal performances in terms of bandwidth utilization [NL11]. Unfortunately, these solutions are unrealizable in a CDN due to two main reasons. First, they rely on heavy computations which are intractable in the CDN hardware (although the equipment have a very large

bandwidth, their computing capabilities are quite small [net]). Second, the proposed models are elastic video bit-rate based and assume infinitely divisible data streams. However, in the context of rate adaptive streaming, the video bit-rate of representations are well pre-defined. Each stream has to be either delivered in its entirety, or not delivered at all. Consequently, instead of maximizing the bit-rate, the throughput of the network is maximized by maximizing the number of delivered streams.

We propose the *discretized streaming* model which is more suitable for delivering multiple live rate adaptive video channels in modern CDNs. The main challenge is to determine a delivery scheme that maximizes the number of delivered streams in a 3-tier network (as shown in Figure 2.4) that is constrained by the capacity of its inner equipment. We give a formal formulation of the general problem that maximizes the utility of delivered streams by a set of delivery trees. Furthermore, we prove that the problem is NP-complete. This general problem formulation is the first step towards a complete work of delivering live rate adaptive streams over the CDN infrastructure.

As the NP-completeness claim implies, it is currently impossible to implement an optimal solution for the general case. Then, we especially consider a practical scenario, which corresponds to today's CDN implementation of live streams. For this scenario, we present an algorithm, which is fast, easy to implement, and near optimal. We provide formal theoretical approximation bounds, which are shown to be negligible for the regarded configuration. The algorithm represents the second contribution for the problem. To our knowledge, today's CDN providers apply ad-hoc delivery techniques. Our algorithm is thus the first scientific reference for optimal delivery in the discretized streaming model.

The remainder of the Chapter is organized as follows. In Section 4.2, we introduce the discretized streaming model and formally define the problem. Then, in Section 4.3 we provide a formulation of the problem through Integer Linear Programming (ILP) and prove that it is NP-complete. We overview the rationale behind the network configuration in Section 4.4, and describe our near-optimal algorithm for the scenario. The evaluation of the algorithm is provided in Section 4.5. Finally, some conclusion remarks are given in Section 4.6.

4.2 System Model and Problem Definition

In what follows, we present our model of live rate adaptive video streaming in a CDN network, which is followed by a formal optimization problem definition. We first summarize the notations that will be used throughout this Chapter in Table 4.1.

4.2.1 Live Rate Adaptive Streaming in CDN

A CDN is composed of a set of communication devices and a set of directed communication links. The topology of a CDN is modeled by a directed graph $G = (V, E)$, where V represents the communication devices, and E represents the communication links.

There are three types of communication devices, also referred to as nodes, in a CDN: a relatively small number of *sources* (origins), a medium size network of *reflectors*, and a large number of *edge servers*. The sources receive and transcode the

V, E	The set of all nodes, and all links in the CDN
V_S, V_R, V_E	The set of sources, reflectors and edge servers in the CDN
E_{SR}	The set of links connecting sources and reflectors
E_{RR}	The set of links inter-connecting reflectors
E_{RE}	The set of links connecting reflectors and edge servers
d_{ij}, $i \in [k]$, $j \in [l]$	The i-th representation of the j-th channel
$\lambda_i, 1 \leq i \leq k$	The bit-rate of the i-th representation of all channels
T_{ij}^s	The delivery tree of d_{ij} rooted at $s \in V_S$
$D(v), v \in V_S \cup V_R$	The set of representations forwarded by v
$c(v), v \in V_S \cup V_R$	The upload capacity of the nodes
$\alpha_{ij}^e, e \in V_E$	The utility score of edge server e to representation d_{ij}

Table 4.1: Notations used in Chapter 4

raw video channels into a set of live *representations*; the reflectors deliver the representations to the CDN edges, and the edge servers offer the received representations to the clients inside their respective ISPs. Let $V_S, V_R, V_E \subset V$ be the set of sources, reflectors and edge servers, respectively.

Accordingly, there are three types of possible connections in E: E_{SR} connects sources to reflectors, E_{RR} allows communication between reflectors, and E_{RE} delivers the representations to the edge servers. They are formally defined as:

$$\begin{aligned} E_{SR} &= \{(u,v) : u \in V_S, v \in V_R\} \\ E_{RR} &= \{(u,v) : u, v \in V_R\} \\ E_{RE} &= \{(u,v) : u \in V_R, v \in V_E\}. \end{aligned}$$

The live streams consist of l different channels. In rate adaptive streaming, the raw video of each channel is transcoded into k representations, where the bit-rate of the i-th representation, $1 \leq i \leq k$, is λ_i. For simplicity of notation hereafter we denote by $[m]$ the integer interval $\{1, \ldots, m\}$. Also, let d_{ij} be the i-th representation of the j-th channel, $i \in [k]$, $j \in [l]$.

The delivery of a representation d_{ij}, $i \in [k]$, $j \in [l]$, from the source nodes to the edge servers is carried out through a set of trees, $\mathcal{T}_{ij}$. Each tree in $\mathcal{T}_{ij}$, also referred to as the *delivery tree*, has one of the source nodes as its root and edge servers as its leafs. We denote by T_{ij}^s the delivery tree of d_{ij} rooted at $s \in V_S$. For convenience, let $V(T)$ and $E(T)$ denote the node and edge sets of tree T, respectively.

Note that every *forwarding node* v, either source or reflector, can participate in the delivery of multiple representations. However, for any representation d_{ij}, $i \in [k]$, $j \in [l]$, v can be a part of only a single delivery tree in $\mathcal{T}_{ij}$. In addition, every forwarding node $v \in V_S \cup V_R$ is also limited by the total outbound bit-rate (capacity) it can support, $c(v)$. Let $D(v)$ be the set of representations forwarded by v, $v \in V_S \cup V_R$. Then,

$$\sum_{i \in [k]} \lambda_i \cdot |\{j : d_{ij} \in D(v)\}| \leq c(v).$$

Like some related work [ASV11, AMM+11, ZAB+12], we consider that the outbound

capacity of equipment is the only constraint.

4.2.2 Problem Definition

Ultimately we would like every edge server to receive all the representations it requires. This however might not be possible in a underprovisioned CDN. Due to the outbound capacity constraints at the forwarding nodes, the CDN may support the delivery of only a subset of representations for each edge server. In such case, the CDN provider leverages statistics to prioritize the delivery [NSS10].

The preferences of edge servers in respect to the available representations is captured in a *utility score*, such that α_{ij}^e is the utility score that edge server e assigns to representation d_{ij}. To evaluate the performance of a delivery scheme, the idea is thus to evaluate a *utility score function* $\alpha_e(\mathbf{X_e})$ for each edge server $e \in V_E$ as follows:

$$\alpha_e(\mathbf{X_e}) = \sum_{i\in[k]} \sum_{j\in[l]} \alpha_{ij}^e x_{ij}^e$$

where $\mathbf{X_e}$ is an indicator matrix of size $k \times l$ such that x_{ij}^e has a value of 1 if e receives d_{ij} and 0 otherwise.

Our objective is to study the *Maximum Average Utility Score* (MAUS) problem, which essentially is the maximization of the *average* utility score function of the edge servers, as summarized below.

Problem 1 *Given the topology and capacity constraints of a CDN, find delivery tree sets, $\{\mathcal{T}_{ij}\}_{i\in[k],j\in[l]}$, such that $\sum_{e\in V_E} \alpha_e(X_e)$ is maximized.*

4.3 Problem Formulation and Problem Complexity

We now discuss the complexity of Problem 1. We first provide an ILP formulation for MAUS, and then we show that the problem is NP-complete.

4.3.1 Integer Linear Programming formulation

We use the notation introduced in Section 4.2 and extend it by defining two new variables, y and h. Let $T_{ij}^s \in \mathcal{T}_{ij}$, $i \in [k]$, $j \in [l]$, be a delivery tree. Then, for every edge $(uv) \in E$, y_{ijs}^{uv} is an indicator variable such that:

$$y_{ijs}^{uv} = \begin{cases} 1 & \text{if } (u,v) \in E(T_{ij}^s), \\ 0 & \text{otherwise.} \end{cases}$$

For nodes $u, v \in V$ such that $(u,v) \notin E$ we define $y_{ijs}^{uv} = 0$. For every node $v \in V$, h_{ijs}^v is an upper bound on the depth of v in T_{ij}^s, i.e.

$$h_{ijs}^{uv} = \begin{cases} \geq \text{depth of } v \text{ in } T_{ij}^s, & \text{if } (u,v) \in E(T_{ij}^s), \\ = \infty, & \text{otherwise.} \end{cases}$$

To ease the notation, let us define $I^v_{ijs}(U)$ to be the sum of y variables that correspond to incoming edges into $v \in V$ from the nodes in $U \subseteq V$, i.e.

$$I^v_{ijs}(U) = \sum_{u \in U} y^{uv}_{ijs}$$

Similarly, let $O^v_{ijs}(U)$ be the sum of y variables that correspond to outgoing edges from v to nodes in U, i.e.

$$O^v_{ijs}(U) = \sum_{u \in U} y^{vu}_{ijs}$$

For simplicity, in the ILP formulation, we omit the use of set membership indication $\in$ for the main notations. Whenever we write $\forall i$, $\forall j$, $\forall s$, $\forall r$, $\forall e$, and $\forall v$, we imply $\forall i \in [k]$, $\forall j \in [j]$, $\forall s \in V_S$, $\forall r \in V_R$, $\forall e \in V_E$, and $\forall v \in V$, respectively. Moreover, we use i, j, s, r, e, and v to refer to representations, channels, sources, reflectors, edge servers, and general nodes, respectively.

Then, we present the formal problem formulation. The objective of the problem is to maximize the sum of the utility scores of the delivered streams through a set of delivery trees. Thus, we first express the objective as follows:

$$\textbf{max.} \quad \sum_{e \in V_E} \sum_{i=1}^{k} \sum_{j=1}^{l} \alpha^e_{ij} x^e_{ij} \tag{4.1}$$

The objective subjects to the following constraints:

$$x^e_{ij} \leq \sum_{s \in V_S} I^e_{ijs}(V_R) \quad \forall i, j, e \tag{4.2}$$

The constraints in (4.2) ensure that the indicator variables x have non-zero values only if there are incoming edges in the respective trees.

$$I^r_{ijs}(V_S \cup V_R) \leq 1 \quad \forall i, j, s, r \tag{4.3}$$

$$I^e_{ijs}(V_R) \leq 1 \quad \forall i, j, s, e \tag{4.4}$$

Constraints in (4.3) and (4.4) guarantee that every node has only one parent in every delivery tree.

$$\sum_{i=1}^{k} \sum_{j=1}^{l} O^s_{ijs}(V_R) \lambda_i \leq c(s) \quad \forall s \tag{4.5}$$

$$\sum_{i=1}^{k} \sum_{j=1}^{l} \sum_{s \in V_S} O^r_{ijs}(V_R \cup V_E) \lambda_i \leq c(r) \quad \forall r \tag{4.6}$$

These two constraints (4.5) and (4.6) enforce the capacity restrictions on each node.

$$h^s_{ijs} = 0 \quad \forall i, j, s \tag{4.7}$$

$$h^r_{ijs} + 1 - h^v_{ijs} \leq |V|(1 - y^{rv}_{ijs}) \quad \forall i, j, r, v \tag{4.8}$$

These two constraints (4.7) and (4.8) guarantee that there is no cycle in the delivery trees.

$$O^r_{ijs}(V_R \cup V_E) \leq |V|(I^r_{ijs}(\{s\} \cup V_R)) \quad \forall i, j, s, r \tag{4.9}$$

$$I^r_{ijs}(\{s\} \cup V_R) \leq O^r_{ijs}(V_R \cup V_E) \quad \forall i, j, s, r \tag{4.10}$$

Finally, in constraints (4.9) and (4.10), we require that reflector nodes have outgoing edges in delivery trees if and only if there is an incoming edge. These two constraints accomplish the formulation of the problem.

4.3.2 NP-completeness

In this section, we demonstrate that the MAUS problem is NP-complete. Let DMAUS be the decision version of the MAUS problem.

Problem 2 *Given topology and capacity constraints of a CDN, and a real number B, do there exist delivery tree sets, $\{\mathcal{T}_{ij}\}_{i\in[k],j\in[l]}$, such that $\sum_{e\in V_E} \alpha_e(X_e) \geq B$?*

Clearly DMAUS is in NP. We now show that DMAUS is NP-hard by a reduction from 3-SAT. Recall that an instance of the 3-SAT problem consists of n variables, $z_1, \ldots, z_n$, and m clauses, $C_1, \ldots, C_m$, where each clause $C_j = (y^1_j \vee y^2_j \vee y^3_j)$, $j \in [m]$, has exactly three literals.

Given an instance of the 3-SAT problem we construct an instance of the DMAUS problem. Let $G_{3SAT} = (V, E)$ be the topology of a CDN. We define V to be (i) a single source node, $V_S = \{s\}$, (ii) $3n$ reflectors that are partitioned into two sets, $V_R = Z \cup A$, such that $|Z| = n$ and $|A| = 2n$, (iii) and m edge servers $V_E = \{u_1, \ldots, u_m\}$. The node set $Z = \{v_1, \ldots, v_n\}$ represents the variables of the 3-SAT instance, the nodes in $A = \{v^t_1, v^f_1, \ldots, v^t_n, v^f_n\}$ represent the two possible values of these variables, and the edge servers represent the m clauses. Overall $|V| = 1 + 3n + m$.

The edge set E is composed of n links between s and the nodes in Z, $2n$ links that connect Z to A, and $3m$ links between A and the edge server nodes. More specifically,

$$\begin{aligned} E = &\{(s, v) : v \in Z\} \\ &\cup \{(v_i, v^t_i), (v_i, v^f_i) : i \in [n]\} \\ &\cup \{(v^t_i, u_j) : i \in [n], j \in [m], z_i \text{ is a literal in } C_j\} \\ &\cup \{(v^f_i, u_j) : i \in [n], j \in [m], \bar{z}_i \text{ is a literal in } C_j\} \end{aligned}$$

For example, Figure 4.1 shows the CDN graph associated with a 3-SAT instance $(x_1 \vee x_2 \vee \overline{x_3}) \wedge (x_1 \vee \overline{x_2} \vee x_4) \wedge (\overline{x_2} \vee x_3 \vee \overline{x_4})$. The capacities of the nodes are defined as follows: $c(s) = n$ and $\forall i \in [n]$, $c(v_i) = 1$ and $c(v^t_i) = c(v^f_i) = m$. We set the number of channels and representations to 1, and the utility score of receiving the single available representation at every edge server $e \in V_E$ is $\alpha^e_{11s} = 1$. Finally, the value B is chosen to be m.

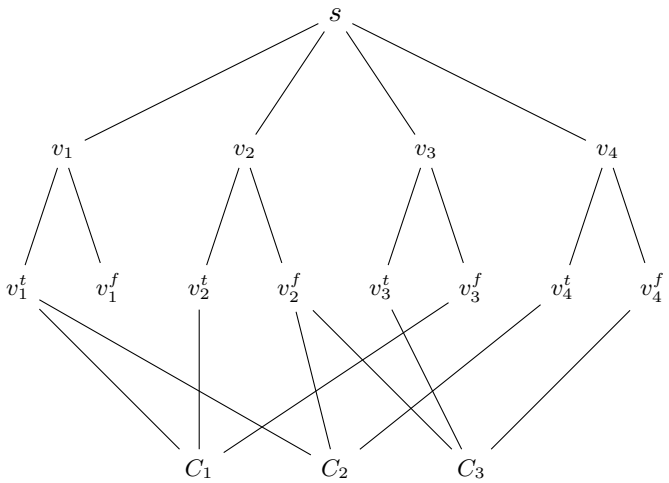


Figure 4.1: The CDN graph associated with 3-SAT

We now show that there is a solution to the 3-SAT problem iff there is a solution to the DMAUS problem.

Let ϕ be a satisfying assignment for the 3-SAT problem instance. We incrementally construct the delivery tree $T_{11}^s = (V_{11}^s, E_{11}^s)$. At first $V_{11}^s = \{s\}$ and $E_{11}^s = \emptyset$. Then, if there exists an edge server $u_j \notin V_s^{11}$, we pick one of the literals in C_j that have a true assignment in ϕ (since ϕ is a satisfying assignment, there must be at least one such literal). W.l.o.g. let z_i be a literal in C_j and $\phi(z_i) = true$. We add to the tree the nodes v_i, v_i^t, u_j and the edges (s, v_i), (v_i, v_i^t), and (v_i^t, u_j). Note that some of the nodes or the edges might already exist in the tree, so we just add the missing ones. These steps never violate the capacity constraints, as for any $v_i \in Z$, at most one outgoing edge, either (v_i, v_i^t) or (v_i, v_i^f), can be added to E_{11}^s, which depends on the assignment ϕ (the first in case $\phi(z_i) = true$, and the latter otherwise). It is easy to conclude that T_{11}^s is a delivery tree rooted at s and has all the nodes V_E as its leafs, and thus the utility score function has a total score of m. The feasibility of each step follows directly from the definition of the CDN topology, G_{3SAT}.

In the opposite direction, let T_{11}^s be a solution to the DMAUS problem (as there is only one representation, one channel and one source node, the solution is a single delivery tree). We construct ϕ by iterating over the nodes in V_E. For every $u_j \in V_E$, if v_i^t is the parent of u_j in T_{11}^s we define $\phi(z_i) = true$, and $\phi(z_i) = false$ otherwise. Note that due to capacity constraints, it is impossible that both v_i^t and v_i^f are in T_{11}^s, and thus the assignment is feasible, i.e. the same variable will never be assigned both $true$ and $false$. After the iteration ends, if there are any undefined variables, we set their values to $true$. What remains to be shown is that ϕ is a satisfying assignment. For every clause C_j, $j \in [m]$, there must exist a literal which corresponds to the father of u_j in T_{11}^s (due to the construction of G_{3SAT}) and has a $true$ assignment in ϕ (due to the iterative definition of ϕ), and thus C_j is true.

4.4 A Practical Scenario and Algorithm

As the above NP-completeness claim implies, it is currently impossible to implement a fast optimal solution for the general case. Thus, we focus on a practical scenario which corresponds to today's CDN implementation. For this practical scenario, we propose a near optimal greedy algorithm which produces delivery trees for every channel.

4.4.1 Practical Bundle Delivery in CDN

In practical CDNs, one can assume that every reflector is connected to any other reflector by a direct link, i.e. for any $u, v \in V_R$ and $u \neq v$, it holds $(u, v) \in E_{RR}$. This can be justified by the fact that the links between reflectors are essentially international connections in the public Internet backbone, where any equipment is virtually connected to any other equipment. In fact, the specifications of the CDN Federation [BSB+12, Le 12] impose full connectivity between the hosts of every member CDN. In what follows we describe a fundamental and popular CDN scenario, named *Homogeneous Bundle Delivery*.

For services which are based on rate adaptive streaming, today's CDNs do not deliver each representation individually. Instead, they gather all representations of a given channel into one *bundle*, and deliver the whole bundle from the source(s) to the edge server(s). Due to the fact that the majority of transcoders are the same, all bundles have roughly the same size, which simplifies the delivery management.

An example of this scenario is as follows: the client of a large-scale CDN is a prominent over-the-top (OTT) service provider, which diffuses a TV package to a large audience. Every channel is bundled, with a total rate of $\lambda = \sum_i \lambda_i, i \in [k]$. All the edge servers are expected to receive all the bundles in the same way, i.e. $\alpha_j^e = 1$ for every $e \in V_E$ and $j \in [l]$ (note the slight change of notation due to the bundling of representations). For a fixed rate data stream, the capacity constraint of every node is essentially an upper bound on the number of simultaneous bundles that the node can support. For simplicity let $b_v = \lfloor c(v)/\lambda \rfloor$ be the number of bundles that can be supported by any $v \in V_S \cup V_R$. As previously, we use the indicator variables x_j^e, $e \in V_E$, $j \in [l]$, which have the value of 1 if the edge server e receives the bundle of the j-th channel. As a result, the objective for this scenario can be summarized as follows:

$$\max \sum_{j \in [l]} \sum_{e \in V_E} x_j^e.$$

4.4.2 The Bundle-Delivery Algorithm

In this section, we describe a fast near-optimum algorithm which is named as Bundle-Delivery for the above practical scenario. The notations used for the description of the algorithm is shown in Table4.2.

First of all, for simplicity of exposition, and following our assumption of full connectivity between the source and reflector nodes (E_{SR}), we can assume there is a single super-source s^* with upload bandwidth $b_{s^*} = \sum_{s \in V_S} b_s$. Note that any solution for the case of having a single super-source can be easily converted into a solution

s^*	The super-source
$\lambda = \sum_i \lambda_i$	The bit-rate of a bundle of all representations
T_j	The delivery tree of the j-th channel
$b_v = \lfloor c(v)/\lambda \rfloor$	The capacity measured on number of bundles for $v \in V_S \cup V_R$
b_v^j	The residual capacity of v after the constructure of T_j
f_v^j	The number of boundle forwarded by v in T_j

Table 4.2: Notations for the Bundle-Delivery algorithm

with multiple sources by distributing the load among the sources according to their upload capacities.

The detailed Bundle-Delivery algorithm is depicted in Algorithm 3. The Bundle-Delivery algorithm is composed of two phases. In the first phase (from line 2 to line 12 in Algorithm 3), we iteratively construct one delivery tree T_j for every channel j, $j \in [l]$. Then, in the second phase (from line 13 to line 15 in Algorithm 3), we provide a local improvement (named as **2-hop connection improvement**) for potentially unused capacities in the first phase. Each iteration of the first phase can be further divided into two parts: first we decide the set of reflectors $V_j \subseteq V_R$ will be be used as forwarding nodes in T_j (from line 4 to line 11 in Algorithm 3); then, based on this set of forwarding reflectors, we generate the delivery tree T_j in line 12 of Algorithm 3.

The general motivation behind the algorithm is that by reducing the number of nodes in every T_j we also reduce the amount of capacity "wasted" on inter-reflector communication (because in the tree structure, the inter-connection capacity equals to the number of nodes in the tree minus 1). Based on our assumption of complete graph, for each delivering tree, the problem is selecting a minimum number of reflectors (V_j) with sufficient aggregate capacity. To achieve that, V_j is composed of reflectors with the maximum residual capacity at that time which is sufficient to deliver the j-channel bundle to as many edge servers as possible. Note that nodes with residual capacity of 1 are not used in the first phase due to the fact that having them as inner nodes in some T_j is not beneficial in any way (as they can forward a bundle to only one node). Instead, we use 2-hop connections in the second phase to utilize their capacity.

There are two main sets of variables in the algorithm. For every node $v \in V_R$ we use b_v^j, $0 \leq j \leq l$, to denote the residual forwarding capacity of node v after the construction of trees $T_1, \ldots, T_j$, i.e. the capacity which remains at v to forward bundles for channels $j+1, \ldots, l$ after it has already forwarded the bundles for channels $1, \ldots, j$. We also define f_v^j, for every $v \in V_R$, $j \in [l]$, to be the number of bundles forwarded by v in T_j, i.e. the number of children v has in T_j. After V_j is determined, these variables are updated according to the forwarding capacity used by each node in V_j and construct the corresponding tree. After the construction of each tree, we update the number of bundles that can still be forwarded by the reflectors and the super-source s^*. This process continues until either all the channels are delivered to all the edge servers or the forwarding capacity of the super-source and/or the reflectors is exhausted.

In the following, we explain the process of the tree construction and the two hop

Algorithm 3: Algorithm: Bundle-Delivery

1: Initialize $\forall v \in V_R : b_v^0 \leftarrow b_v$ and $j \leftarrow 1$
2: **Phase 1:**
3: **while** $\exists v \in V_R : b_v^{j-1} \geq 2$, $b_{s^*} \geq j$, and $j \leq l$ **do**
4: Initialize $V_j \leftarrow \emptyset$ and $U_j = \{u : b_u^{j-1} \geq 2\}$
5: Initialize $\forall v \in V_R : f_v^j \leftarrow 0$
6: **while** $\sum_{v \in V_R} f_v^j < |V_j| + |V_E| - 1$ and $U_j \neq \emptyset$ **do**
7: Extract from U_j a node u^* with maximum forwarding capacity, i.e. $b_{u^*}^{j-1} = \max_{u \in U} b_u^{j-1}$
8: $f_{u^*}^j \leftarrow \min\{b_{u^*}^{j-1}, |V_E| + |V_j| - \sum_{v \in V_j} f_v^j\}$
9: Add u^* to V_j
10: Update $\forall v \in V_R : b_v^j = b_v^{j-1} - f_v^j$
11: Update $j \leftarrow j + 1$
12: **Construct delivery tree** T_j based on the nodes V_j
13: **Phase 2:**
14: **if** not all edge servers receive all channel bundles and $b_{s^*} > l$ **then**
15: Use **2-hop connection improvement** to deliver channel bundles

connection optimization in detail.

Construct Delivery Tree

For every channel j, T_j is a tree which is rooted at s^*, has V_j as its intermediate nodes, and some or all of V_E as its leafs. The out-degree of every node $v \in V_j$ in T_j is exactly f_v^j. The topology of the tree can be arbitrary with a single constraint, that the super-source has exactly one child in T_j.

The topology of T_j has no effect on the number of leafs in T_j, which is $(\sum_{v \in V_j} f_v^j - |V_j| + 1)$ (as we show later). For live streaming, the source-to-end delay should be minimized. Thus, we would ultimately like the tree to have the minimum possible height to minimize the number of hops from the super-source node to the leafs. Minimizing the height of the delivery tree also reduce the latency from the root to the leaves in the tree. For that purpose, we construct T_j in the following way.

- First connect s^* to the node v with maximum value of f_v^j in V_j.
- Then, connect v to f_v^j nodes with the next highest values of f^j in V_j, and repeat this process for every node in $V_j \setminus \{v\}$ according to decreasing value of f^j until all the nodes in V_j have a parent in T_j.
- In the end of the above process, some nodes will have an out-degree less than the corresponding value of f^j. Connect these nodes to a subset of edge servers, yet to be included in T_j, such that the degree of those nodes will match their f^j values (Lemma 1 below shows that it is always possible).

Figure 4.2 shows an example of tree generation. The node-set V_j is composed of two nodes: u and v with $f_u^j = 3$, and $f_v^j = 2$ (Figure 4.2(1)). As u has a higher

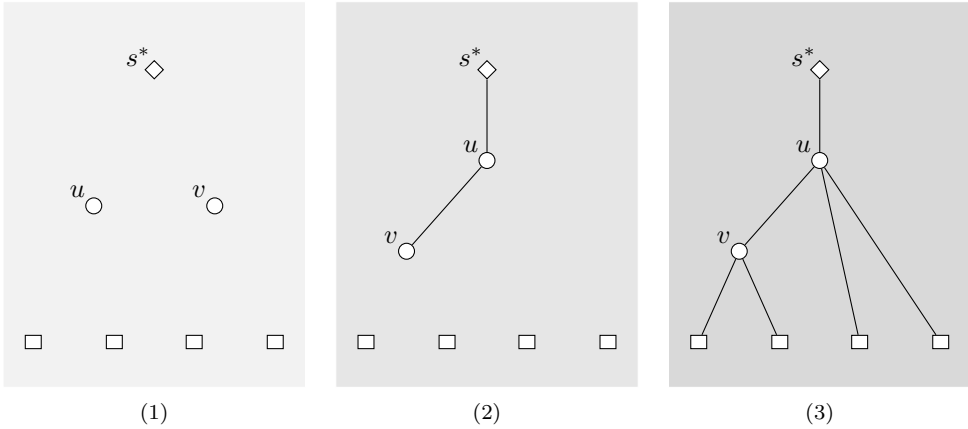


Figure 4.2: T_j generation: $V_j = \{u, v\}$, $f_u^j = 3$, $f_v^j = 2$.

forwarding capacity than v, it is connected to s^* and v is set as the child of u (Figure 4.2(2)). At this point the out-degrees of u and v are 1 and 0, respectively, which are less than their forwarding capacities in T_j. Thus, both u and v are connected to 2 edge servers each (Figure 4.2(3)).

Two-hop Connection Improvement

Let j^* be the last channel for which a delivery tree T_{j^*} was constructed. It is easy to observe that the trees generated are feasible delivery trees rooted at s^*, where T_j delivers the j-th channel bundle, $j \in [j^*]$. In the tree construction phase, for each tree, the super-source consumes 1 unit of upload capacity (because we restrict the number of children of the source to 1). Thus, after the tree construction phase, the source node might have some unused forwarding capacity.

As we show later, if not all edge servers receive every channel bundle, then every reflector $v \in V_R$ has a forwarding capacity of $b_v^{j^*} \leq 1$. The second phase of the algorithm (2-hop connection improvement) aims to improve the performance by using the unused source and reflectors capacity. Thus, we use reflectors with non-zero forwarding capacity to deliver additional channel bundles to edge servers that are yet to receive them in a two-hop fashion $s^* \to v \to u$, where $v \in V_R$ with $b_v^{j^*} = 1$, and $u \in V_E$ such that u is not a leaf in T_i.

In the following Lemma 1, we will show that all the leafs in T_j are edge server nodes.

Lemma 1 *In every constructed tree T_j there are exactly $\sum_{v \in V_j} f_v^j - (|V_j| - 1)$ leafs and $\sum_{v \in V_j} f_v^j - (|V_j| - 1) \leq |V_E|$.*

Proof. Note that for any T_j, $j \in [l]$, the root s^* has exactly one child. Therefore, $|V_j| - 1$ nodes have a parent node which is a reflector. According to the tree construction in Algorithm 3, every reflector $v \in V_j$ has an out-degree of f_v^j in T_j. Thus, we can conclude that in T_j a total of $|V_j| - 1$ forwarding capacity is used to deliver the channel bundle between reflector nodes, which results in T_j having $\sum_{v \in V_j} f_v^j - (|V_j| - 1)$ leaf

nodes. In Algorithm 3, line 7 and line 8 enforce the inequality $\sum_{v \in V_j} f_v^j - (|V_j| - 1) \leq |V_E|$, and therefore we can conclude that tree construction (line 12) is feasible, i.e. it is always possible to connect a reflector v to some edge server, yet to be in T_j, to fill the out-degree of v in T_j.

4.4.3 Performance Analysis

In this section, we provide a detailed performance analysis of the Bundle-Delivery algorithm.

Running time

There are three main steps which contribute to the running time of the Bundle-Delivery algorithm.

- The initialization step (line 1 in Algorithm 3) and takes $O(|V_R|)$ time to execute.
- The second step is the tree construction phase (from line 2 to line 12 in Algorithm 3). We need to maintain the information about the residual forwarding capacity at every reflector. This can be easily implemented by using an ordered list. At first the reflectors are sorted in decreasing order according to their initial forwarding capacity (b_v^0, $v \in V_R$). Then, during the execution of the inner loop (line 6 to line 9 in Algorithm 3), for every $j \in [l]$ at most one node in V_j will not use all of its forwarding capacity in T_j. Removing all the nodes except for, possibly, the last one, and moving the last one in the ordered list according to its updated residual capacity, takes $O(|V_R|)$ time. Clearly the use of the ordered list allows an easy implementation of the collection of nodes U_j, as we are only interested in the information about the residual capacities in decreasing order. Tree generation itself takes linear time in $|V_j| + |V_E|$. Thus, the total running time of the second step is $O(|V_R| \log |V_R| + l \cdot |V_R| + l \cdot |V_E|)$.
- Finally, the third step (the 2-hop connection improvement) takes $O(b_{s^*} + |V_R| + |V_E|)$ time.

To conclude, the running time of Bundle-Delivery is $O(|V_R| \log |V_R| + l \cdot |V_R| + l \cdot |V_E| + b_{s^*})$.

Approximation ratio

Then, we will theoretically analyze how our algorithm approaches to the optimum solution by measuring the approximation ratio. In the analysis we ignore the 2-hop connection improvement phase of the algorithm, as it has no direct effect on the performance bounds, but rather serves as a local improvement, which may or may not occur.

Let S and S^* be the values of the solution obtained by Bundle-Delivery and the optimal one, respectively. The next lemma shows that either the unused capacity of every reflector is at most 1 or $S = S^*$.

Lemma 2 *At the end of the execution of* Bundle-Delivery*, it holds that if* $\exists u \in V_R : b_u^{j^*} > 1$ *then* $S = S^*$.

Proof. Suppose that after the construction of T_{j^*} there exists a node $u \in V_R$ such that $b_u^{j^*} > 1$. Clearly, $b_u^j > 1$ for every $j \in [j^*]$ as the residual forwarding capacity $b_u^{(\cdot)}$ can only decrease in subsequent executions from line 3 to line 12 in the algorithm. There are two possible cases during the construction of T_j:

- **Case 1:** Node u was chosen in step line 7 of Algorithm 3. Then since $b_u^{j^*} > 1$ we can conclude that $f_u^j = |V_E| + |V_j| - \sum_{v \in V_j \setminus \{u\}}$ and u is the last node to be added to V_j.
- **Case 2:** Node u was not chosen in step line 7 of Algorithm 3. Then, the inner loop of the tree construction phase (line 6 of Algorithm 3) ended due to equality $\sum_{v \in V_j} f_v^j = |V_E| + |V_j| - 1$.

Thus, for every $j \in [j^*]$ the equality $\sum_{v \in V_j} f_v^j - (|V_j| - 1) = |V_E|$ holds and due to Lemma 1 the number of leafs in T_j is $|V_E|$. Taking a closer look at the conditions in the main loop (line 3 of Algorithm 3) we can see that $j^* = \min\{l, s^*\}$ (as for any $j \in \{0, 1, \ldots, j^* + 1\}$, $b_u^{j-1} \geq 2$), and as a result $S = |V_E| \cdot \min\{l, s^*\}$. On the other hand, we have $S \leq S^* \leq |V_E| \cdot \min\{l, s^*\}$. Therefore, $S = S^*$.

We are now ready to prove the main theorem of this section. We make a reasonable assumption that it is possible to deliver at least one channel to all the edge servers, i.e. $S^* \geq |V_E|$.

Theorem 1 *Following the execution of* Bundle-Delivery *and under the assumption that* $S^* \geq |V_E|$, $S/S^* \geq 1 - (b_{s^*}/|V_E|)$.

Proof. We start by drawing an upper bound on the optimal solution. The number of edge servers that can potentially be reached by all the reflectors is at most $\sum_{v \in V_R} b_v$. However, some of the capacity needs to be used to maintain the delivery trees (source-reflector, and reflector-reflector connections). In the best case scenario (in terms of "wasted" capacity), every node is used in only one tree, and b_{s^*} reflectors have s^* as their parent in one of the delivery trees. Thus, $S^* \leq \sum_{v \in V_R} b_v - |V_R| + b_{s^*}$.

Next we analyze the solution produced by Bundle-Delivery. If there exists a node $v \in V_R$ such that $b_v^{j^*} > 1$ (recall that T_{j^*} is the last tree to be constructed), then according to Lemma 2, $S = S^*$. Otherwise, for every $v \in V_R$, $b_v^{j^*} \leq 1$. Let $x = |\{v : b_v > 1, b_v^{j^*} = 1\}|$ be the number of nodes that had their residual bundle capacity reduced to 1 during the execution of the algorithm, and $y = |\{u : b_u = 1\}|$ be the number of nodes with bundle delivery capacity of 1 prior to the execution of the algorithm. Hence the total forwarding capacity used in all the delivery trees (before the 2-hop connection improvement), $\sum_{j \in [j^*]} \sum_{v \in V_j} f_v^j = \sum_{v \in V_R} b_v - (x + y)$.

Note that when a node u^* is selected to be added to V_j, $j \in [j^*]$, in line 7 of Algorithm 3, it cannot be used again unless it was the last node to be added to T_j (due to the computation of the forwarding capacity $f_{u^*}^j$). Thus, the total number of reflectors in all the delivery trees is the number of potentially participating nodes

($v \in V_R$, with $b_v > 1$) plus at most $j^* - 1$ times a node might appear in two and more trees (due to the partial assignment of f). Formally, $\sum_{j \in [j^*]} |V_j| \leq |V_R| - y + j^* - 1$.

Based on Lemma 1 and the above we can now derive a lower bound for our solution. As discussed in the beginning of this section we ignore the 2-hope connection improvement in our evaluation (it can only improve the lower bound of S^*).

$$\begin{aligned} S &= \sum_{j \in [j^*]} \left(\sum_{v \in V_j} f_v^j - (|V_j| - 1) \right) \\ &= \sum_{j \in [j^*]} \sum_{v \in V_j} f_v^j - \sum_{j \in [j^*]} (|V_j| - 1) \\ &\geq \sum_{v \in V_R} b_v - (x + y) - (|V_R| - y + j^* - 1) + j^* \\ &= \sum_{v \in V_R} b_v - |V_R| - x + 1 \end{aligned}$$

As we derived in Lemma 2, when a partial forwarding capacity $f_v^j < b_v^{j-1}$ is assigned to some $v \in V_j$, $j \in [j^*]$, the delivery tree T_j has $|V_E|$ leafs. As we already stated in this proof, at most one node can be assigned a partial forwarding capacity in each tree, and thus there is a partial assignment in at least x delivery trees. As a result, $S \geq x|V_E|$. On the other hand, $S \leq S^* \leq l|V_E|$. Summarizing all of the above and under the assumption that $S^* \geq |V_E|$ we obtain the approximation ratio of the Bundle-Delivery algorithm as:

$$\begin{aligned} S/S^* &\geq \frac{S^* - x - b_{s^*} + 1}{S^*} = 1 - \frac{x + b_{s^*} - 1}{S^*} \\ &\geq 1 - \frac{1}{|V_E|} - \frac{b_{s^*} - 1}{S^*} \geq 1 - (b_{s^*}/|V_E|). \end{aligned}$$

4.5 Evaluation

We now evaluate the approximation ratio, S/S^* of the Bundle-Delivery algorithm proposed in Section 4.4.2. For small instances, S^* is obtained by the implementation of the ILP model in IBM ILOG CPLEX software. Due to the complexity of the model, for large instances, we computed S^* according to the upper bound given in the proof of Theorem 1. We simulated a CDN network with 1 source. The number of edge servers ranges from 10 to 100,000. Then, in order to test the performance of the algorithm, the CDN delivery network is underprovisioned, we set the number of reflectors so that the CDN infrastructure can supply 90% of the required bandwidth to deliver 50 channels. Each channel contains 8 representations. The bit-rate of the representations follows recommendations from Apple HTTP Live Streaming [appb]: $\{150, 240, 440, 640, 1240, 1840, 2540, 4540\}$ kbps. Source and reflector capacity is set to 1 Gbps.

The results are shown in Figure 4.3 and Figure 4.4. For small networks (Figure 4.3), edge servers receive 44 channels on average in the optimal solution. The result obtained by the Bundle-Delivery algorithm is slightly below. For large

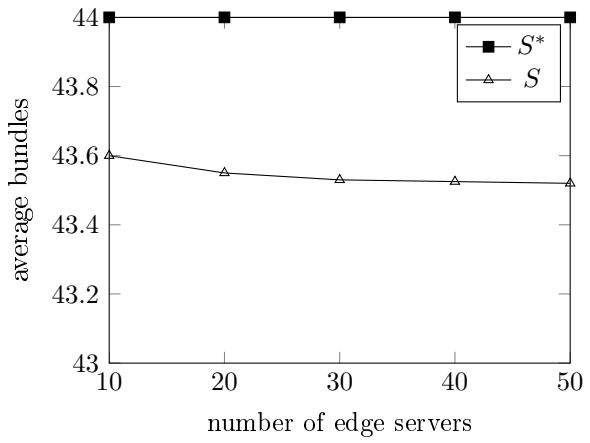


Figure 4.3: Evaluation for small instances

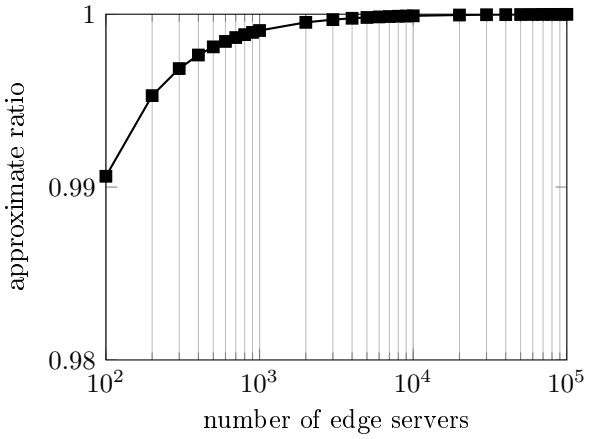


Figure 4.4: Evaluation for large instances

networks (Figure 4.4), the Bundle-Delivery algorithm achieves an approximation ratio of $1 - 10^{-3}$. For networks with 1,000 edge servers, the ratio S/S^* is at least 0.999056, and starting from $x = 10{,}000$ it is above 0.999906. Moreover, it should be noted that the computations were carried out in less than 30 seconds for every instance on a standard desktop PC. The evaluation demonstrates that our algorithm runs fast, and could obtain near optimum solutions.

4.6 Conclusions and future work

In this Chapter we introduced the discretized streaming model, which represents multiple rate adaptive live videos delivery in today's CDNs. We first formulated a general optimization problem for prioritized live video delivery which we showed to be NP-complete. Then, we focused on a realistic scenario: the CDN provider is in charge of delivering the representations of each channel as one bundle. We developed a fast and simple algorithm that guarantees a solution which is at least $1-(b_{s^*}/|V_E|)$

times the optimal solution. To the best of our knowledge, this is the first result for the discretized streaming model [LS13a].

The work presented in this Chapter provide a fundamental theoretical basis toward live rate adaptive streaming in CDN. First of all, the problem formulated is a general problem, with undefined utility function. It is of great importance to explore the computation of the utility score, which is influenced by a large number of parameters, especially in the context of live rate adaptive streaming where the demand from clients may change very quickly. Secondly, the algorithm proposed in this work assumes complete graph CDN topology, consequently, in future work, it would be interesting to design algorithms for general CDN topologies.

In the next Chapter, we further explore the above limitations of this work. We specialize the general problem in to a user-centric problem such that the utility of an edge server on streams represents the QoE perceived by the users of the edge server on receiving the stream. Thus, the objective is to maximize overall user satisfaction on the services. Moreover, we redesign the three CDN functionalities (content placement, content delivery and user assignment) to build a practical CDN system that delivers live rate adaptive streams to a large audience in a dynamic environment. General delivery trees construction algorithm for random graph CDN topology is proposed.

Chapter 5

A User-centric Live Rate Adaptive Streaming CDN System

5.1 Introduction

In the previous Chapter 4, we have introduced a general optimization problem—the *discretized streaming capacity problem* that maximizes the utility of delivered live rate adaptive streams in a underprovisioned CDN infrastructure, under the topology and capacity constraints of the CDN. We formally formulate the problem through ILP and prove that the complexity of the problem is NP-Complete. These work provide a theoretical foundation for live rate adaptive streaming in CDN. In this Chapter, we take a further step towards live DASH video delivery in CDNs by introducing a user-centric live DASH streaming CDN system.

Rate adaptive streaming (also referred to as DASH) is designed to serve heterogenous video consuming devices. The objective is to maximize the Quality of Experience (QoE) of diverse users characterized by their end devices and access links. Rate adaptive streaming accomplishes this objective by providing multiple representations for the users. The users dynamically and adaptively choose the representation best fits to its characteristic. A consequence is that users are capable to play any of the representations (out of the alternative ones), on which the users perceive difference levels of QoE. Out of all alternative representations, a user prefers to receive the one having the highest QoE. As the CDN capacity is dedicated to finally earn user satisfaction, a challenge for CDN providers is to design user-centric live video deliver scheme that maximizes the user satisfaction on the live streams delivered through its infrastructure.

As we have discussed in Chapter 4, the rapidly increasing live video traffic and the multiple representations characteristic of rate adaptive streaming impose a heavy transmission burden on the CDN delivery network. Consistently, we consider (1). the most critical resource to provision is the overall amount of data that CDN equipments can emit per unit of time, and (2). the CDN infrastructure is underprovisioned. That is, the CDN infrastructure is not capable to convey all representations of requested contents to all the edge servers.

For the user-centric delivery, we reuse the discretized streaming model proposed in

Chapter 4, which is especially designed for rate adaptive streaming in modern CDNs, by exploring the *utility* of edge servers on the streams. Typically, the utility connects the stream delivered to the edge server with the value appraised by users on the edge server to the stream. By defining user QoE based utility function, the *user-centric discretized streaming model* maximizes the user satisfaction on the delivered streams. More specifically, in the underprovisioned CDN, instead of serving all representations of a given channel to edge servers requesting this content, the model prioritizes the delivery of representations to the edge servers according to the user satisfaction level on the representations.

5.1.1 Our Contributions

We accomplished two contributions for the user-centric live rate adaptive streaming in CDN network, including a theoretical optimization problem formulation and a practical system. In the following, we summarize the two contributions separately.

Contribution 1. The first contribution relates to a theoretical optimization problem. First of all, we propose a QoE model which measures the user satisfaction level on the received representations. This model permits to define a user QoE based utility function. The user-centric delivery scheme depends on such utility to evaluate streams by how users are satisfied with them. Then, we formulate a joint optimization problem based on the discretized streaming model presented in Chapter 4. The objective is to maximize the overall user satisfaction, as well as guarantee the *max-min fairness* on user satisfaction levels. Further, we introduce new variables in order to make the model jointly decide three problems: (1) the representation that should be sent to the edge servers, (2) a set of delivering trees and (3) the assignment of users to edge servers. These three problems correspond to the three main mechanisms implemented by CDN providers: *content placement*, *content delivery* and *user assignment*. Through a set of simulation on a toy-CDN infrastructure, we demonstrate that the user-centric discretized streaming strategy could achieve higher user satisfaction comparing to the previous approaches.

Contribution 2. The second contribution is a practical implementation which enables a CDN provider to efficiently deliver live rate adaptive streams in a large-scale and dynamic environment. The system contains three components: (1) a user assignment component; (2) a content placement component; and (3) a content delivery component. Each component targets one of the three aforementioned CDN main mechanisms. At last, we conducted a large simulation campaign to evaluate the system performance. We utilized real traces collected from justin.tv [jus] website to set up the behavior of clients. The simulation results approve our expectation: the system could serve a large audience with high satisfaction and limited CDN infrastructure cost.

The rest of this Chapter is organized as follows. We describe in Section 5.2 the model that measures user satisfaction on representations. The optimization problem is formally given in Section 5.3 and evaluated in a toy-CDN infrastructure in Section 5.4. We present in Section 5.5 a practical implementation of a CDN system for delivering

live rate adaptive streams. We provide in Section 5.6 a first evaluation based on a large-scale real-trace simulator. Finally, Section 5.7 concludes this work.

5.2 User-centric Discretized Streaming Model

When the delivery system is underprovisioned, a service provider cannot offer as many video representations as it would with a well-provisioned delivery system. As a consequence, the representation that fits the best a given user's downloading capability might not be offered by the CDN. This situation forces users to get a lower bit-rate video representation, hence with a lower quality.

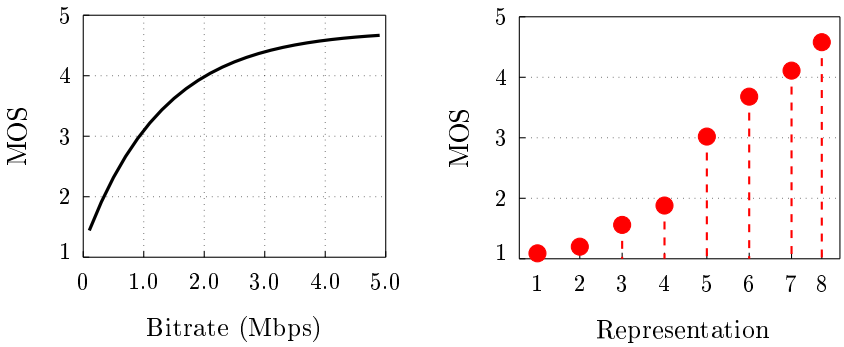
We introduce the concept of *user satisfaction* for every user and every representation. The model we propose simplifies the reality, but it takes into account two major features of rate adaptive streaming: (*i*) representations are encoded with pre-defined settings, and (*ii*) each end-user is characterized by a highest viewable representation. Intuitively, it measures how far is the received representation to the best possible one for a given user. We detail this concept in Section5.2.1.

5.2.1 User satisfaction

The ideal way to measure the QoE perceived by human viewers is to run subjective tests and to combine the obtained scores into a Mean Opinion Score (MOS). An MOS traditionally ranges from 1 to 5, where 1 stands for bad quality and 5 for excellent quality. However, subjective test campaigns are costly to conduct and require time. Researchers have thus worked on QoE models that map objective non-perceptual video quality metrics into MOS values [RCKS10, TKS10, KSJI10]. The idea is that the combination of several paramaters (such as video bit-rate, packet loss, and video genre) can lead to a good estimation of the MOS. In particular, models based solely on video bit-rates represent a nice trade-off between the accuracy of QoE estimation and the simplicity of implementations [MHXW12, YH08, ZWCK13]. In the following, we consider that the CDN provider uses such bit-rate based QoE model to estimate on-the-fly the MOS of a given video representation (Fig. 5.1(a)). One of the advantages of these models is that they do not require any computation on the user devices. Of course, more sophisticated QoE models, based on a wider set of parameters, can be implemented to improve MOS estimation accuracy.

We focus on the bit-rate parameters to measure QoE because it is a central parameter of rate-adaptive streaming. Indeed, (*i*) representations are encoded with pre-defined bit-rates, and (*ii*) each end-user is characterized by her highest viewable representation, which is the representation that the user can download with the highest bit-rate. In Fig. 5.1(b), we depict a typical *MOS profile* for a video. Without loss of generality, we combined the bit-rate QoE model given in [YH08] and the bit-rates recommended by Apple for the set of video representations in HTTP Live Streaming (see Table 5.1 and [appb]). As can be expected, the MOS grows when the encoding bit-rate increases, but the growth is not linear.

However, such MOS-based models of representations do not fully capture the satisfaction of a heterogeneous population of end-users. Indeed, the QoE depends on the context of video consumption [MZÅ11]. For example an end-user watching a



(a) MOS for video bit rates (b) MOS for 8 representations

Figure 5.1: MOS models

Representation	1	2	3	4	5	6	7	8
Bit rate (in kbps)	150	240	440	640	1,240	1,840	2,540	4,540

Table 5.1: Representation bit-rates

320p video on a smartphone in public transportation is satisfied although the same video is perceived as being of very bad quality by the same end-user watching it on her TV screen. Furthermore, various reasons can prevent an end-user to watch high-quality video representations, including too small screen resolution and not enough bandwidth, hence receiving a 720p video may lead to the worst user satisfaction as such video cannot be played at all.

Previous context-aware QoE models such as [MZÅ11] do not consider multi-representation streaming. Thus, we introduce the *relative satisfaction* metric (or satisfaction in short). We normalize the satisfaction of a given user getting a given representation to the satisfaction obtained if she gets her best possible representation. This relative satisfaction is thus between 0 and 1. We show in Fig. 5.2 the relative satisfaction of an end-user who cannot play a video above the fourth representation. We used the same MOS as in Fig. 5.1(b). Please observe in Fig. 5.2 that the third representation has a good relative satisfaction although the same representation has a disastrous MOS in Fig. 5.1(b).

Let us now introduce some notations. The set of end-users is noted N. For each user $n \in N$, we denote by i^n the index of the best possible representation, and by j^n the index of the channel she is watching. Thus, $d_{i^n j^n}$ is the best possible representation requested by user n. We then define a function $MOS(d_{ij}, n)$, which is the QoE rating of user n for representation d_{ij}. In comparison with the plain MOS profile (Fig. 5.1(b)), this MOS score is user-related. There are two cases:

- End-user n can decode and watch d_{ij} ($i \leq i^n$ and $j = j^n$). In this case, the value of $MOS(d_{ij}, n)$ is the plain MOS profile of the representation $MOS(d_{ij})$.
- End-user n cannot decode and watch d_{ij} ($i > i^n$ or $j \neq j^n$). In this case, we "force" $MOS(d_{ij}, n)$ to be equal to one (the least MOS score value). In

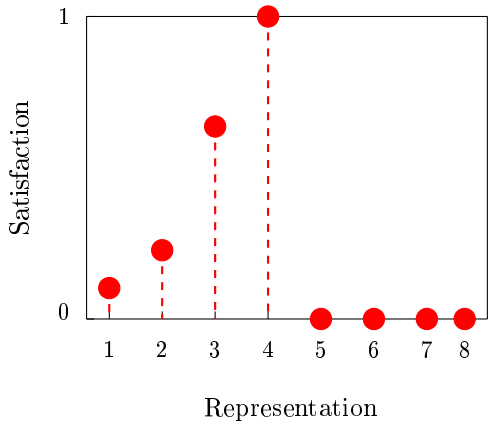


Figure 5.2: Relative satisfaction for users who can play up to the fourth representation

other words, an end-user is totally unsatisfied when getting an unwatchable representation.

$$u_{ij}^{n} = \frac{MOS(d_{ij}, n) - 1}{MOS(d_{i^n j^n}, n) - 1} \tag{5.1}$$

We note u_{ij}^{n} the satisfaction of user $n \in N$ when n receives representation d_{ij}. To compute u_{ij}^{n}, we use the ratio of the MOS of representation d_{ij} to the MOS of its requested representation $d_{i^n j^n}$. Since we would like the user satisfaction to be into the range $[0, 1]$, we use the following computation:[1]

5.2.2 Live video streaming in a CDN

We consider the same 3-tier CDN topology and live rate adaptive streaming mechanism as in Chapter 4. Hence, we reuse the model presented in Section 4.2.1. For fluent reading, we shortly recall the notations used in the model.

A CDN $G = (V, E)$ is composed of sources (V_S), reflectors (V_R), and edge servers (V_E). The sources receive and transcode the raw video channels into a set of live representations; the reflectors deliver the representations to the CDN edges, and the edge servers offer the received representations to the end-users. Consequently, there are three types of connections: E_{SR} from V_S to V_R, E_{RR} between pairs of V_R, and E_{RE} from V_R to V_E. The live streams consist of l different channels. The raw video of each channel is transcoded into k representations, where the bit-rate of the i-th representation, $1 \leq i \leq k$, is λ_i. Let d_{ij} be the i-th representation of the j-th channel, $i \in [k]$, $j \in [l]$.

The delivery of a representation d_{ij}, $i \in [k]$, $j \in [l]$, is carried out through a set, $\mathcal{T}_{ij}$, of trees of G. Each tree in $\mathcal{T}_{ij}$, has one of the source nodes as its root and edge-servers as its leafs. We denote by T_{ij}^{s} the delivery tree of d_{ij} rooted at $s \in V_S$. Lastly, every *forwarding node* v is limited by the total outbound bit-rate (capacity)

[1] We assume that the plain $MOS(d_{ij})$ is larger than 1 for all representations, because logarithmic QoE models are used [YH08].

it can support, $c(v), v \in V_S \cup V_R$. As we have discussed, the outbound capacity of the equipment is the only constraint.

5.3 Formulation of the Capacity Problem

We start the presentation of the optimization problem by introducing several objective functions in Section 5.3.1. We then present the ILP formulation in Section 5.3.2.

5.3.1 Objective Functions

As we consider the CDN is underprovisioned, the network cannot support to transmit all the requested streams to all the requiring edge servers. Under the upload capacity constraints at the forwarding nodes, the CDN supports the delivery of a subset of representations, based on the utility of the streams. To formalize the *user-centric discretized streaming capacity problem*, we define two binary variables:

- $x_{ij}^e = 1$ indicates that edge-server $e \in V_E$ receives representation d_{ij}, 0 otherwise (this notation has been defined in Chapter 4.2.1).
- $z_{ij}^{ne} = 1$ indicates that user $n \in N$ is attached to edge-server $e \in V_E$, and gets representation d_{ij} from e, 0 otherwise.

We suppose that the highest achievable representation for each user n is given. It is determined by user device type and access link. From Section 5.2.1, the user satisfaction u_{ij}^n is thus known for all users and all representations.

The problem can be formulated by using only the variable x_{ij}^e upon giving user requirements on each edge-server. However, CDN implements three main algorithms: user-to-server assignment, content placement and content delivery. As one of the main functionalities of CDN, user assignment determines edge-server requirements, and has an direct impact on stream delivery. In this case, we would like to formulate the problem in a way which combines all the three aforementioned CDN functionalities. Thus, we use the variable z_{ij}^{ne} to include users' behaviors.

Then, the *utility* of representation d_{ij} for edge-server e is the aggregated satisfaction of end users that are attached to e on representation d_{ij}. Formally, the utility of an edge server e in V_E can be written as:

$$\sum_{n \in N} u_{ij}^n \cdot z_{ij}^{ne}, \qquad \forall i \in [k], j \in [l], e \in V_E$$

It is possible to design many different objective functions. We propose one user-centric objective function in the following. An intuitive objective function consists in maximizing the average satisfaction of users, which is expressed as follows:

$$Obj_{global} : \sum_{e \in V_E} \sum_{i \in [k]} \sum_{j \in [l]} \sum_{n \in N} u_{ij}^n \cdot z_{ij}^{ne} \tag{5.2}$$

Such an objective function is simple, but it has some weaknesses. In particular, it can introduce biases, when Obj_{global} can be maximized by focusing on a subset

of the population. It is the case here: users watching low-resolution representations will be served first because satisfying them has a lower impact on the infrastructure in comparison to satisfying users watching high-definition (HD) representations. We thus need another objective function, which avoid such bias. A way to mitigate such weaknesses is to ensure some degree of fairness among the end-users. A simple user-centric objective is formulated as a *max-min*, where the goal is to maximize the satisfaction of the user that is the most poorly served, which is formally given by:

$$Obj_{maxmin} : \min_{n} \left(\sum_{e \in V_E} \sum_{i \in [k]} \sum_{j \in [l]} u_{ij}^{n} \cdot z_{ij}^{ne} | \forall n \in N \right) \tag{5.3}$$

Due to the aforementioned weaknesses, we emphasize on the max-min objective depicted in Equation (5.3). Therefore, the overall objective function is defined as:

$$\textbf{maximize} \quad \varepsilon \cdot Obj_{global} + Obj_{maxmin} \tag{5.4}$$

where ϵ is chosen to be small enough so that Obj_{maxmin} always dominates Obj_{global}. In other words, the user-centric discretized streaming capacity problem is formulated as, firstly ensuring that all end-users have a decent satisfaction, and then maximizing globally the satisfaction of the whole population. In the following, we formulate an ILP model with this objective. It is worth to note that other objective functions can also be considered.

5.3.2 Integer Linear Program Formulation

We formulate an ILP model for the user-centric discretized streaming capacity problem. We reuse the variables defined for the discretized streaming ILP model:

$$y_{ijs}^{uv} = \begin{cases} 1 & \text{if } (u,v) \in E(T_{ij}^{s}), \\ 0 & \text{otherwise.} \end{cases}$$

$$h_{ijs}^{uv} = \begin{cases} \geq \text{depth of } v \text{ in } T_{ij}^{s}, & \text{if } (u,v) \in E(T_{ij}^{s}), \\ = \infty, & \text{otherwise.} \end{cases}$$

$$I_{ijs}^{v}(U) = \sum_{u \in U} y_{ijs}^{uv}$$

$$O_{ijs}^{v}(U) = \sum_{u \in U} y_{ijs}^{vu}$$

In CDN, it is frequent that an edge-server is assigned to a given population of end-users, or a geographic area. For example, an edge-server can be located within the network of an Internet Service Provider (ISP) to serve exclusively clients of this ISP. We define another binary variable:

- $p_n^e = 1$ indicates that user $n \in N$ can be assigned to edge server $e \in V_E$, 0 otherwise.

Constraint (5.5) makes sure that users get at most one representation from one edge-server.

$$\sum_{e \in V_E} \sum_{i} \sum_{j} z_{ij}^{ne} \leq 1 \quad \forall n \tag{5.5}$$

With constraint (5.6) edge-server send only the representations they receive.

$$z_{ij}^{ne} \leq x_{ij}^{e} \quad \forall n, e, i, j \tag{5.6}$$

Constraint (5.7) redirects users to proper edge-servers.

$$z_{ij}^{ne} \leq p_{n}^{e} \quad \forall n, e \tag{5.7}$$

Constraint (5.8) does not allow the delivery of representations that are not associated with a positive satisfaction.

$$z_{ij}^{ne} < u_{ij}^{n} + 1 \quad \forall n, e, i, j \tag{5.8}$$

Constraint (5.9) restricts edge-server capacity.

$$\sum_{n \in N} \sum_{i} \sum_{j} z_{ij}^{ne} \cdot \lambda_i \leq c(e) \quad \forall e \tag{5.9}$$

Then, the remaining constraints construct trees from source to edge-servers, which are identical to the Constraints from 4.2 to 4.10 presented in the ILP formulation of the discretized streaming model (Section 4.3). For fluent reading, we repeat these constraints and shortly explain them:

$$x_{ij}^{e} \leq \sum_{s \in V_S} I_{ijs}^{e}(V_R) \quad \forall i, j, e \tag{5.10}$$

$$I_{ijs}^{r}(V_S \cup V_R) \leq 1 \quad \forall i, j, s, r \tag{5.11}$$

$$I_{ijs}^{e}(V_R) \leq 1 \quad \forall i, j, s, e \tag{5.12}$$

$$\sum_{i=1}^{k} \sum_{j=1}^{l} O_{ijs}^{s}(V_R) \lambda_i \leq c(s) \quad \forall s \tag{5.13}$$

$$\sum_{i=1}^{k} \sum_{j=1}^{l} \sum_{s \in V_S} O_{ijs}^{r}(V_R \cup V_E) \lambda_i \leq c(r) \quad \forall r \tag{5.14}$$

$$h_{ijs}^{s} = 0 \quad \forall i, j, s \tag{5.15}$$

$$h_{ijs}^{r} + 1 - h_{ijs}^{v} \leq |V|(1 - y_{ijs}^{rv}) \quad \forall i, j, r, v \tag{5.16}$$

$$O_{ijs}^{r}(V_R \cup V_E) \leq |V|(I_{ijs}^{r}(\{s\} \cup V_R)) \quad \forall i, j, s, r \tag{5.17}$$

$$I^r_{ijs}(\{s\} \cup V_R) \leq O^r_{ijs}(V_R \cup V_E) \quad \forall i, j, s, r \tag{5.18}$$

Constraint (5.10) indicates that if there is no incoming edge, the stream is not received on the edge-server. Constraints (5.11) and (5.12) enforce a single parent in every tree. Constraints (5.13) and (5.14) enforce capacity restrictions. Constraints (5.15) and (5.16) prevent cycles in every tree. Constraint (5.17) states that if there is no incoming edge in a tree, there cannot be outgoing edges as well. Finally Constraint (5.18)forces reflectors to have at least one output edge if they have an incoming edge in a tree.

This ILP model solves the three main CDN functionalities all at once. The optimal solution jointly indicates : (1) which representations are sent to which edge servers; (2) how to deliver streams from CDN sources to CDN edge servers; and (3) how users are assigned to edge servers.

5.4 Proof-of-Concept for User-centric Discretized Streaming

We now show the relevance of our approach to address the challenge of maintaining a high satisfaction on a population of users in a context of underprovisioned CDN. We considered a very simplified context. Our goal here is not to mimic the reality (we conducted realistic simulations in the following Section 5.6). It is rather a proof-of-concept to illustrate the benefits one can expect from our model. We implemented the ILP model in IBM ILOG CPLEX optimizer and computed the best achievable delivery in the following network configuration.

Due to the complexity of the ILP model, we evaluate the model in a toy CDN network, which consists of one source, two reflectors and eight edge-servers. Figure 5.3 shows the topology. This infrastructure has been reserved and provisioned by the CDN provider to serve one channel, which is encoded in eight different representations. The bit-rates of the representations are set according to the recommended Apple HTTP Live Streaming (see Table 5.1 and [appb]). We considered a population of 360 end-users interested in this channel. The number of users is set to a proper value to make sure that the bottleneck of the video delivery lies within the CDN delivery network, rather than the links between edge servers and end users. Users require one of the 8 representations based on their devices and access networks. The characteristics of devices and network connections (see Table 5.2) are inspired by some recent statistics collected during London Olympic games [nbc].

We assumed a scenario where the CDN provider has severely underprovisioned its infrastructure. The upload capacity of each source is 10 Mbps, which is enough to send all representations. However, the upload capacity of each reflector is only 7.2 Mbps, which is insufficient. The aggregated upload capacities of reflectors represent only 16% of what would be required to send all representations to all edge servers. Finally, the capacity of edge servers is 80 Mbps. There is no constraint on the assignment from end users to edge servers.

We first analyzed the solution that can be directly obtained from the streaming capacity problem. The streaming capacity problem obtains the maximum deliver-

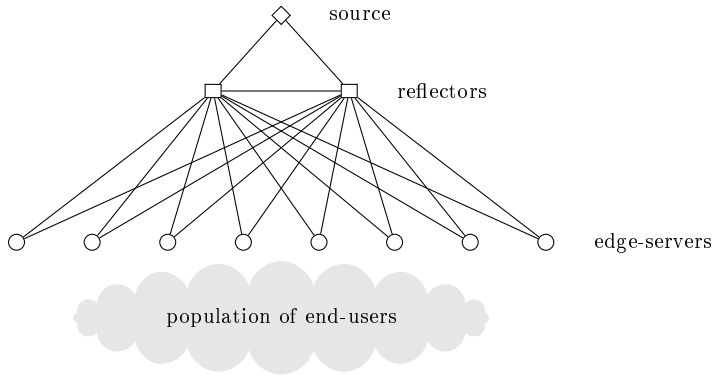


Figure 5.3: Topology of CDN toy-network

able bit rate of a network (discussed in Chapter 2.3.4) Thus, we computed the best achievable streaming bit rate of the toy-network based on the algorithms described in [SLC+11]. Then, we found the best packing of representations for such bit rate. We refer to this approach as the *traditional approach*. To measure performances, we compute the Cumulative Distribution Function (CDF) of the satisfaction of users (see Figure 5.4). A point at $(0.6, 0.2)$ means that 20% of users experience a relative satisfaction greater than 0.6.

We can immediately see in Figure 5.4 the weaknesses of the traditional approach based on the streaming capacity problem. The QoE is far from what can be obtained with a dedicated approach. With the traditional approach, two thirds of users have a satisfaction below 0.75 although there exist solutions where all users have a satisfaction over 0.82.

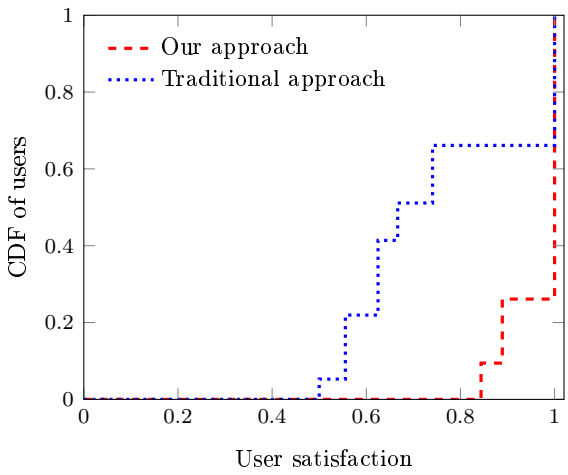
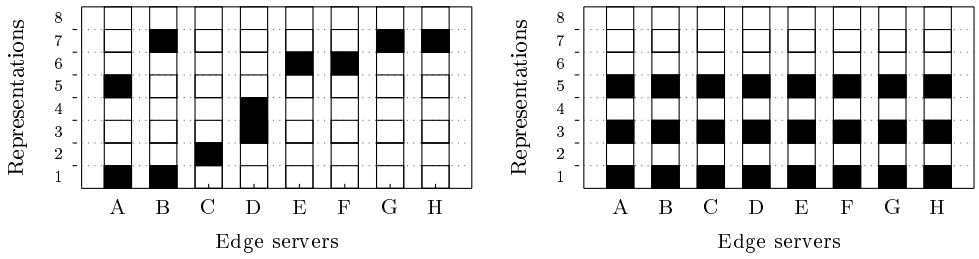


Figure 5.4: CDF of user satisfaction

To better understand the poor performance of the traditional approach, we plot in Figure 5.5 the received representations on each edge server. A black square indicates the representation is received. In the optimal solution, all representations (except the



(a) Our approach (b) Traditional approach

Figure 5.5: Received representations on each edge server

highest one) are hosted in at least one edge server. On the contrary, the traditional approach provides the same set of representations to all edge servers. It is important to notice that the overall number of representations sent to the edge servers is larger in the traditional approach, but the *diversity* of these representations is lower. Moreover, the ILP is able to adjust the number of representations to both the capacity of the edge servers and the demand from end users.

Our second important observation is that our optimization problem is able to keep a high level of user satisfaction despite the very limited CDN infrastructure. Almost three quarters of the population do not experience any degradation although CDN capacity is less than one fifth of what it should be. These results being promising, we developed algorithms that can be implemented in a real system based on the principles of the user-centric discretized streaming model. In the following, we describe the practical live rate adaptive streaming CDN system.

5.5 A Practical System: SCADOOSH

We describe now a practical implementation for CDNs. The overall system is named SCADOOSH, which stands for SCAle Down fOOtprint for live daSH. SCADOOSH aims to efficiently exploit CDN infrastructure to deliver live video streams to a dynamic, large population of users. We use the nomenclature of the DASH standard, but SCADOOSH is independent of the implemented rate-adaptive technology.

In addition to the traditional 3-tier CDN infrastructure, SCADOOSH requires a centralized organizer, which we call a *coordinator*. The coordinator is in charge of orchestrating the delivery into a CDN network. Most CDNs rely on such a global coordinator, which manage the whole network or a restricted area. For simplicity, we consider hereafter one coordinator and one network, but multiple coordinators can co-exist in a giant CDN as long as the boundaries of the network they rule are well defined. For example it makes sense to consider one coordinator by continent for the largest CDNs. A SCADOOSH coordinator executes three main algorithms:

- *User assignment* algorithm aims to assemble users with similar requirements. The result of this algorithm is a user distribution over the edge servers.
- *Content placement* algorithm calculates content utility, based on the reports

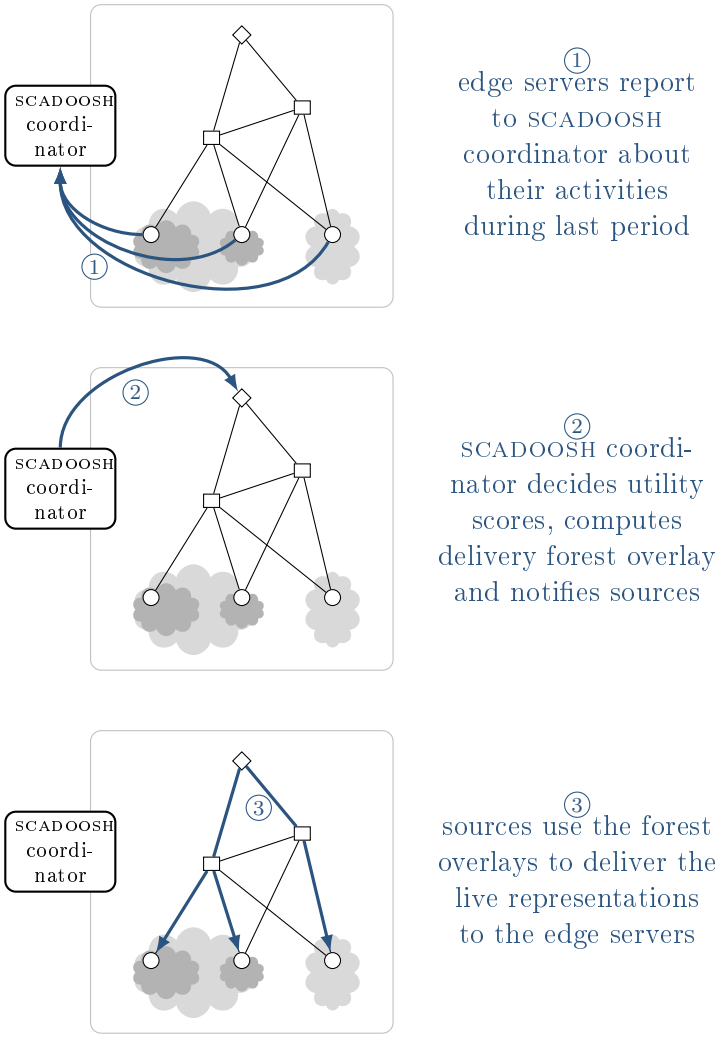


Figure 5.6: The SCADOOSH system

from each edge server about the activity of the served population

- *Delivery trees construction* algorithm determines a utility-driven multi-tree overlay that delivers videos from sources to edge servers.

We represent SCADOOSH in Figure 5.6 on a simple small CDN infrastructure where source, reflectors and edge servers are depicted with a diamond, squares and circles respectively, and the CDN covers two ISP networks (gray clouds). The ISP on the left is even separated into two sub-networks (typically mobile and residential).

SCADOOSH synchronizes on DASH *periods* (defined in DASH Media Presentation Description (MPD) file) to cope with the dynamic feature of rate-adaptive live streaming. For every period, SCADOOSH performs three actions:

① every edge serves periodically reports to the coordinator about the last period. This report may include the number of served clients, the popularity of the

different channels, the popularity of the different representations.

② Upon reception of these reports, the coordinator can estimate the popularity of streams for the next period, and compute the utility score for every edge server and every stream. The utility score indicates the preference of an edge server to receive a given stream. Once utility scores of every edge server have been computed, the coordinator determines a multi-tree delivery where one tree is an overlay for the delivery of one representation stream. The coordinator aims at constructing a *forest* such that the edge servers receive their preferred streams. The information about this forest overlay is sent to origin servers.

③ The edge servers are informed about the stream that they will receive during the next period. They can build their DASH MPD files accordingly. Then, users update the MPD file and use the new file to fetch upcoming content for the next period. During the whole next period, the sources and reflectors use the multi-tree overlay to deliver the incoming live streams.

5.5.1 Type Specified User Assignment

How to assign an user to an edge server when the user starts a streaming session is generally the responsibility of the CDN providers through a DNS redirection scheme. A tool like GeoDNS has been used for years on that purpose [geo]. In the case of proprietary delivery platforms, studies showed that the redirection depends on both network proximity, load-balancing [TFK+11a] and business issues [PB12]. In conformance with [AJZ11], we assign here every end user to only one edge server and *we promote a policy where the end users are assigned to edge servers according to the characteristics of their devices and network connections.* Some network scientists have advocated such policy before us [tel]. We will show later that significant gains in terms of CDN infrastructure can be achieved if the CDN is able to adequately redirect requests with regard to the specialization of edge servers. In the following, we present a practical (*a.k.a.* simple) implementation of such type-based user assignment.

In DASH, users have 2-dimension requirements: the watching channel and the requiring representation determined by the DASH rate adaptation algorithm. User assignment determines user distribution on edge servers, and further drives the user-centric content placement. This new characteristic allows for novel user assignment strategy that is aware of the second dimension of user requirement. Servers serve users who request videos from a wide spectrum of end devices. Proactively preposition all the representations on edge servers provides full DASH adaptation flexibility. However, this is highly demanding on the CDN infrastructure. Assembling users with similar demand can mitigate the transmission burden on CDN infrastructure, while also provide certain DASH adaptation flexibility. User requires representation based on the characteristics of their devices and access link. Consequently, users type can be used as the indicator of their requirement.

Consequently, we roughly distinguish three families of user types: `mobi`, `HD` and `norm`. In short, the `mobi` family is for users with mobile devices and low bandwidth network connections, the `HD` for users with high profile, and last, the `norm` is the non-specialized family, which stands for other types of users and traffic load redirected

from the first two families. Note that finer divisions with a higher number of families can be possible. End-users naturally drive the choice of the session family: an end-user watching the channel from her smartphone or having a poor network connection falls naturally in the `mobi` family while an end-user consuming the video on a High-Definition TV is a `HD` user.

With rate-adaptive streaming technologies, each family can be associated with a subset of representations. Users with the same type are assigned to be served by the same edge servers. Accordingly, edge servers are also specialized into three types: the `mobi` servers serve users from mobile devices, the `HD` servers are for users with high profile, last, the `norm` servers are the non-specialized servers which stand for other types of users and traffic load redirected from the first two types of servers. Indeed, we would like to leverage the fact that HD representations are rarely demanded by edge servers in the `mobi` family while `HD` edge servers should not require low-quality representations.

The technologies to identify users' family is out of the scope of this work. Here we only discuss shortly a possible plan for user identification. For `mobi` users, the mobile device detection technologies have been developed from the earliest days of the mobile web. Typically the HTTP "User-Agent" header field allows any server to distinguish HTTP requests from mobile devices. Other quick testing can be done at session openings to roughly test the network connection. Then, in order to further recognize `HD` users from the other wired users, a concept of user *profile* can be used for the registered users. The profile of a user records the historical percentage of HTTP requests for HD segments when the video requests are NOT released from mobile devices. This profile can work as an indicator of user type, as users with high Internet access connections resided (such as FTTH and high speed ADSL) are more likely to have a high profile. Then, the wired users can be further categorized into `HD` users who are more likely to demand HD representations, and `norm` users who stand for the rest. Hereafter, we assume that the coordinator is able to identify the family of every user.

Once a user connects to the system, a number of live channels are made available to her. When the user clicks on the thumbnail of the video, an HTTP GET request is routed to the user assignment component. This component recognizes the type of the user. Then, the user assignment algorithm works as follows:

1. The algorithm first searches for edge servers (1) from the same family, (2) within the same *area* (*e.g.* ISP), (3) hosting users watching the same channel, and (4) with load lower than a pre-defined threshold. If some edge servers match these requirements, the user is assigned to one of them by a random choice.

2. Otherwise, the algorithm restricts lookup to `norm` edge servers (1) within the same area, (2) hosting users watching the same channel, and (3) with load lower than the threshold. Similarly, the user is assigned to one of such servers.

3. Finally, if no edge server is found in the above steps, the user is assigned to the `norm` edge server within the same area with the minimum traffic load. Otherwise, the system cannot provide live video service for this user.

Then, the user fetches a MPD file from the edge server, which states the base URL of the video, the available video representations, segments and their relative URLs. After parsing the MPD file, the user requests the upcoming segments and plays the video.

There are several benefits of the type specified user assignment: (1) Assembling users watching the same channels can provide DASH adaptation flexibility. When DASH make adaptation decisions, the alternative representations are also required by some other users, thus already made available on the edge server. (2) Assembling users with the same type can maintain high QoE with less load on the CDN infrastructure. For `mobi` edge servers, the HD representations are rarely demanded, on the contrary, `HD` edge servers normally do not require low representations. Consequently, a subset of representations is sufficient to provide high user QoE levels. (3) Further, the redirection mechanism is aware of edge server area and traffic load, thus curbs inter-domain traffic and provides server load balancing.

It is worth to mention that the composition of user types determines that of edge servers. The coordinator is in charge of setting the family for each edge server. Strategies to dynamically change the configuration of edge servers according to users demand have been recently developed [WLC12]. Such work complements our proposal.

5.5.2 Utility-based Content Placement

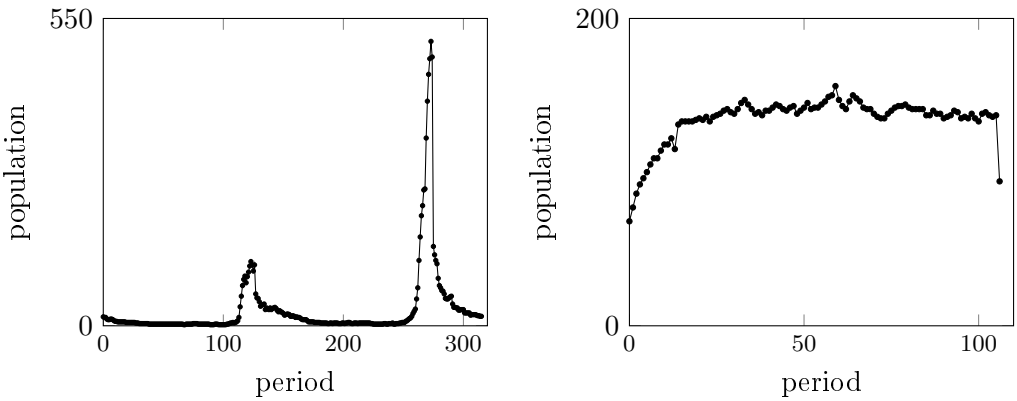
The intuition behind content placement is that we need to prioritize the delivery of certain representations over others for each edge server. In the system, we propose a simple function, where the utility of a representation d_{ij} at the edge-server $e \in V_E$ (defined as α_{ij}^e in Chapter 4), is expressed as the potential aggregated satisfaction of all users that are served by e. Let $N_e \subseteq N$ be the set of end-users served by e. Formally, we have:

$$\alpha_{ij}^e = \sum_{n \in N_e} u_{ij}^n \qquad , \forall e \in V_E, \forall i \in [k], \forall j \in [l]$$

A high utility for a representation d_{ij} for e means that a large number of end-users served by e can be highly satisfied by d_{ij}. Therefore the representation d_{ij} should be delivered to e in priority. However, to compute the utility, the coordinator should know the requests that will be issued *during the next period*. The main challenge is that the number of requests for a given representation at an edge-server can dramatically and unpredictably change between two consecutive periods. To address this problem, we propose in the following a solution based on *time series forecasting*, where the coordinator leverages requests from previous periods to predict the requests for the next period, for every representation on every edge server.

The forecasting model that we utilized is called Autoregressive Integrated Moving Average (ARIMA) [BJ90]. This model has already proven its efficiency for the popularity of channels in IPTV system [WLZ11b]. The algorithm is as follows:

1. Until sufficient records are collected (50 periods are suggested in [BJ90]), the coordinator anticipates that the number of requests for each representation at each edge-server in the next period will be exactly the same as in the previous period.



(a) Long lasting video population (b) Short lasting video population

Figure 5.7: Two types of user-generated video

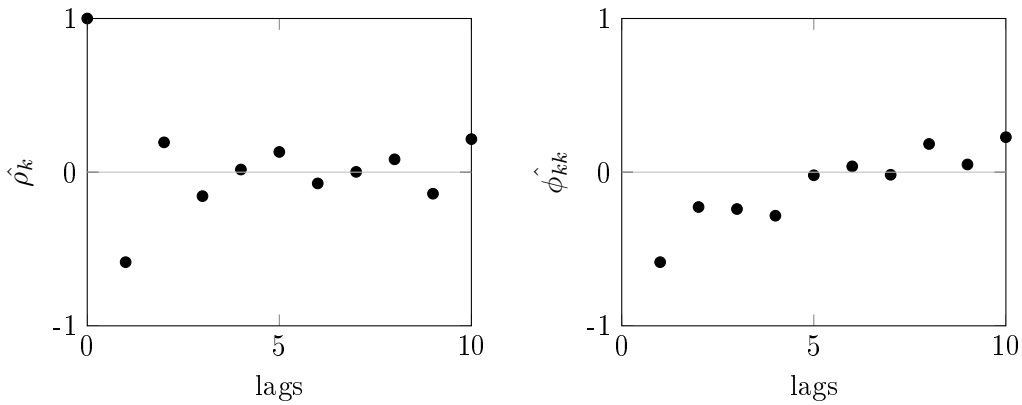
2. When enough records have been collected, the parameters of the identified ARIMA model are estimated ; predictions can be made more accurately.

3. Previous predictions are confronted to real requests. If the performances are below a threshold, the ARIMA parameters should be retrained.

More formally, let T be the current period, so the coordinator should predict the requests for period $T+1$. If reports have been sent from edge servers to coordinator, the latter stores the times series $n^e_{ij}(t), \forall i \in [K], j \in [L], e \in V_E$ for all $t \leq T$. For simplicity, we assume that all these time series are independent, thus we treat them separately. This is not true in DASH. We left for future work the design of more accurate prediction tools, which take into account the correlations between various representations.

Then, we need to identify the ARIMA(p, d, q) model (identify p, d, q parameters). Nowadays, a huge number of live videos are generated and broadcasted by ordinary Internet users (such as videos on justin.tv platform). We collected video popularity information from the justin.tv [jus] platform (a user-generated live video platform), and use these traces to validate our system (in Section 5.6). Comparing to traditional IPTV videos, which has continuous long term channel population historical records, the user broadcast videos can start and end at any time. As a result, we perceived from the traces two different patterns of population time series: long lasting videos and short lasting videos. The long video acts as continuous traditional IPTV channels: peaks are observed during popular hours (Fig 5.7(a)), whereas the short video experiences population increase at the beginning and then remains quiet stable (Fig 5.7(b)).

For these two types of videos, the autocorrelation ρ_k and partial autocorrelation ϕ_{kk} of the first-order difference series have different styles. However, for the second-order difference (∇^2) popularity, both patterns have non-zero $\hat{\rho_1}$ and $\hat{\phi_{kk}}$ tails off, indicating the ARIMA model is ARIMA$(0, 2, 1)$. This model is identical to the one obtained in [WLZ11b], which is derived for traditional IPTV videos. This indicates that our system can be applied for both long-term and short-term types of videos.



(a) Autocorrelation of ∇^2 (b) Partial autocorrelation of ∇^2

Figure 5.8: ARIMA model identification

We plot the estimation of autocorrelation $\hat{\rho}_k$ and partial autocorrelation $\hat{\phi_{kk}}$ in Figure 5.8(a) and Figure 5.8(b). Consequently, the prediction can be expressed as:

$$\hat{n_{ij}^e}(T+1) = 2n_{ij}^e(T) - n_{ij}^e(T-1) + a_{ij}^e(T+1) - \theta_{i,j}^e a_{ij}^e(T)$$

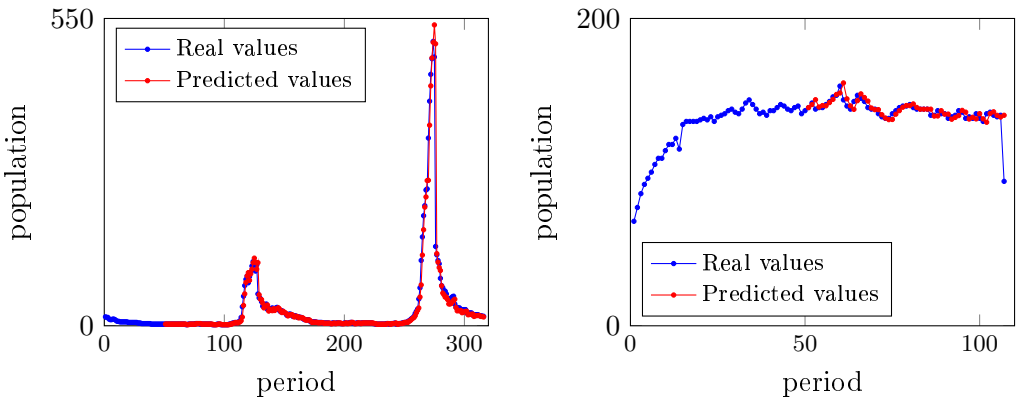
where $n_{ij}^e(T)$ and $n_{ij}^e(T-1)$ are the records in periods T and $T-1$, and a_{ij}^e stands for random errors. In forecasting, the error for the future is treated as zero, thus $a_e^{i,j}(T+1) = 0$. The error for the past is estimated as the difference between the real value and the predicted value, thus $a_{ij}^e(T) = n_{ij}^e(T) - \hat{n_{ij}^e}(T)$. The coefficient θ_{ij}^e can be estimated from the least squares algorithm. To cope with dynamicity, we used the method suggested in [WLZ11b]: For each prediction $\hat{n_{ij}^e}(T)$, we compare the real value $n_{ij}^e(T)$ against its 95% confidence interval. If five continuous real values lie outside the interval, the latest fifty observations are used to retrain the θ_{ij}^e parameter, and the new θ_{ij}^e is used for the following prediction.

We show the quality of the video population prediction for both video types in Figure 5.9(a) and Figure 5.9(b). We utilize the standard way to show the performances of such prediction tools with overlapping curves. As shown in the figures, the ARIMA model is capable to accurately predict population changes for both types of videos.

5.5.3 Utility driven delivery trees construction

After the utility scores are determined, scadoosh coordinator constructs the delivery overlay. It knows (i) the status of the infrastructure, the graph G, and the available upload capacity of every source in V_S and every reflector in V_R, and (ii) the utility score of every edge server in V_E for every representation of every channel.

We describe now the algorithm that builds a multi-tree delivery overlay over the CDN infrastructure (shown in Algorithm 4). The main idea is to create delivery links between an edge server and one of the equipments (source or reflector) that are able to deliver the representation. We process representations iteratively based on the *utility score per rate unit* (`uspru`). The `uspru` is computed for every representation d_{ij} at



(a) On long channels with peaks (b) On short channels without peak

Figure 5.9: Evaluation of the prediction

Algorithm 4: Delivery forest overlay construction

1: $\mathcal{L} \leftarrow$ pile $\{e, d_{ij}\}$ by decreasing `uspru` order
2: transform graph G into graph G'
3: **while** $\mathcal{L} \neq \emptyset$ **and** $V' \cap (V_S \cup V_R) \neq \emptyset$ **do**
4: $e, d \leftarrow$ **pop** $\mathcal{L}$
5: $G'_d \leftarrow$ augment G' with node t, edges (t, s), $s \in V_S$ and edges (t, r), $r \in V_R$ receiving d
6: find max-residual path between e and t in G'_d
7: update G' with new capacities

every edge server u by dividing the utility score α^u_{ij} by the bit-rate λ_i. SCADOOSH coordinator first sorts the set of `usprus` (line 1), then it processes each representation iteratively (lines 3-7).

Let e be the edge server that has to be served. Let d be the representation that has to be delivered, with bit-rate λ. We aim at delivering representation d to edge server e *while minimizing the impact on the infrastructure*. Our algorithm is inspired by the Maximum Residual Energy Routing Path algorithms, which are commonly used in wireless sensor networks with the goal to save energy [CT04]. Instead of delivering through the shortest path, which can quickly drain some equipments and make the network partitioned, the maximum residual capacity path is used. As a result, the bandwidth usage of all equipments are balanced and life time of the system is prolonged. Thus, we look for the delivery path such that the minimum remaining available upload capacity of all equipments in the path is maximum.

To achieve this goal, we transform the original node capacitated infrastructure graph G to an edge capacitated one G'_d. The transformation is illustrated in Figure 5.10. Then, delivering a representation d can be regarded as sending a flow of λ to e, and the delivery link corresponds to the maximum residual capacity path in the transformed graph G'_d. The transformation is in two steps:

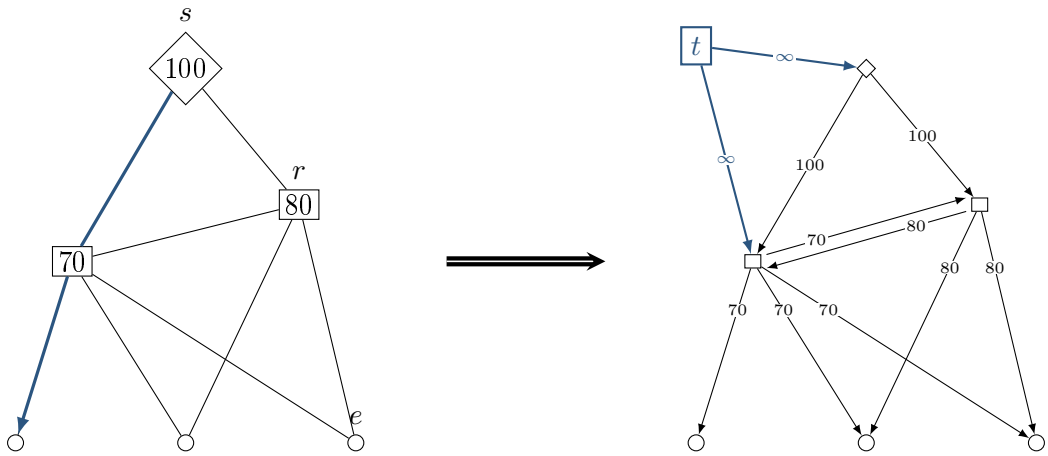


Figure 5.10: Graph transformation

- We first transform the node capacitated graph G to an edge capacitated one G'. We do it by multiplying the original non-directed links between two nodes to bi-direction links. We replace a link between two reflectors (u, v) in G by two links $(u, v, c(u))$ and $(v, u, c(v))$ with each link weighted by the capacity of the head vertex. This bi-directional transformation is not needed for links between sources and reflectors, and between reflectors and edge servers, because streams are never sent in the reverse direction. As a result, in $G'(V', E')$, the set of edges E' is the union of:

$$E_1 = \{(u, v, c(u)), (v, u, c(v)) \forall u, v \in V_R, (u, v) \in E\}$$

$$E_2 = \{(u, v, c(u)), \forall u \in V_R, v \in V_E, (u, v) \in E\}$$

$$E_3 = \{(u, v, c(u)), \forall u \in V_S, v \in V_R, (u, v) \in E\}$$

- The final graph G'_d is obtained by augmenting the obtained graph G' in order to ease the discovery of delivery path for the representation d. We add an abstract node denoted by t. The node t is linked to every node u which can serve the representation with an infinite link. Consequently, E'_d also contains $E_4 = \{(t, u, \infty), \forall u \text{ has } d\}$.

From graph G'_d, it is trivial to find the path from the edge server e to the abstract node t with maximum residual weight. Especially, traditional shortest path algorithms such as the Dijkstra's one can be applied.

5.6 Evaluation

We now evaluate the performances of SCADOOSH. We developed our own simulator with the objective of simulating large-scale systems and a population of users having a behavior inspired from the real traces that obtained from *justin.tv*. A number of live videos are delivered through the CDN network. By measuring the user

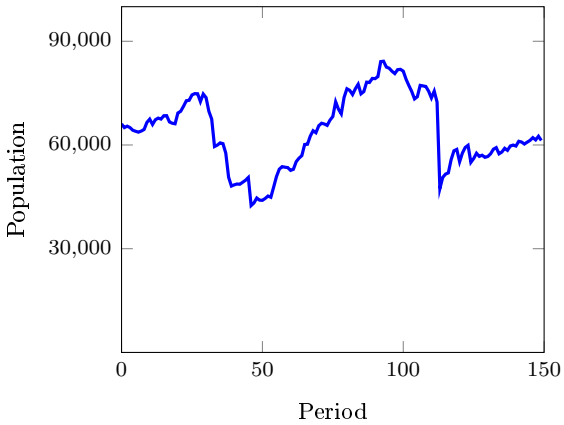


Figure 5.11: Variation of global population over the 150 periods

perceived video quality under various CDN delivery network provisioning conditions. We show that SCADOOSH is capable to maintain high user satisfaction with limited CDN infrastructure cost, and the type-specified user assignment strategy proposed in Section 5.5.1 can achieve higher user satisfaction.

We describe the settings of this simulator in Section5.6.1, and then we analyze the results in Section5.6.2.

5.6.1 Simulation settings

Population of users

We fetched real traces from a popular user-generated live video broadcasting platform, named justin.tv [jus], in August 2012.[2] Every five minutes, we retrieved the popularity of channels of justin.tv. We took 150 of these measures to simulate the variation of popularity of channels in our system. The population variation traces of the 50 most popular channels are used to simulate end user behaviors. Figure 5.11 shows the global population of end users in our simulations.

User setting

We took inspiration from [nbc] to set the population heterogeneity, typically, half of them use mobile devices. This setting is identical to the user setting used in Section 5.4. We defined six types of user network connections: ADSL-slow, ADSL-fast, FTTH, WIFI-slow, WIFI-fast and 3G. Table 5.2 shows the average downlink bandwidth and the average ratio of the population. Along the simulation time, the downlink bandwidth of each user varies around the given average value. At each period starting time, users request the best representation they can get subject to their downlink speed.

[2] Available at *http://enstb.org/~gsimon/Resources/Justintv*

Technologies	Average bandwidth (in Mbps)	Ratio of users
ADSL-slow	2	10%
ADSL-fast	8	30%
FTTH	100	10%
Wifi-slow	1	15%
Wifi-fast	5	15%
3G	0.8	20%

Table 5.2: Technologies and ratio of associated users

CDN setting

We built a large CDN with five geographic *areas*. In each area, equipments are interconnected through a *random graph* in which every link occurs independently with probability 0.8. Among the 50 channels, 30 channels are selected to be *globally hot*, which means that they are accessed by users from any area. The remaining 20 are *locally hot* channels, they are viewed only from users in one area. The capacity of CDN equipments is set to 1 Gbps. In order to serve all users at peak time (no user is rejected due to overloaded edge servers), the CDN contains 320 edge servers.

To value the underprovisioning of the CDN infrastructure, we changed the number of reflectors. We set four underprovisioning configurations with respectively 15, 20, 25, and 30 reflectors. The former one corresponds to a *severely* underprovisioned infrastructure scenario where only 47 Mbps in average have been reserved per edge server for the whole catalog of 50 channels, *i.e.* only 1.38 Mbps per channel that have to be served in every area (*i.e.* the globally hot channels and locally hot channels viewed in the area). Recall that, according to Apple HTTP Live Streaming setting [appb] (see Table 5.1), one channel is a pack of eight representations with an aggregated bit-rate over 11.6 Mbps. The most favorable scenario with 30 reflectors is a *slightly* underprovisioned infrastructure scenario where the CDN reserved around 2.57 Mbps per channel and per edge server.

DASH setting

Each channel consists of 8 representations. Throughout this Chapter, the video bit rate of representations follows the recommendations from Apple HTTP Live Streaming setting [appb], see Table 5.1 for details.

5.6.2 Results

We now analyze the performances of SCADOOSH. Our main comparison is the ideal case where all users are served with their best representation. This comparison is captured by the metric based on relative satisfaction. In Section 5.6.2, we observe whether it is possible to maintain a good user satisfaction despite infrastructure underprovisioning. In Section 5.6.2, we focus on the impact of type-based user assignment on the overall performances.

Overall Performances

We use the same CDF function as in Figure 5.4 to show the performances of SCADOOSH. We represent three curves, corresponding to three different number of reflectors. Results are given in Figure 5.12

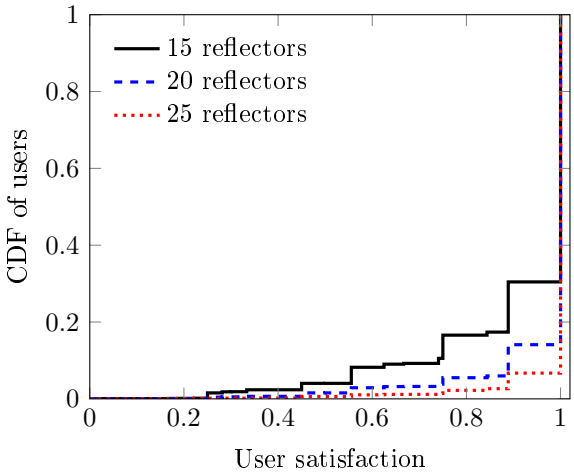


Figure 5.12: CDF of user satisfaction for three numbers of reflectors

Our main observation is that SCADOOSH succeeds in maintaining a good QoE despite the lack of resources in CDN infrastructure. With 25 reflectors, only 6% of users are not served with their best representation. Moreover, most of them have a satisfaction over 0.88. In other words, the QoE is nearly perfect although the CDN slightly underprovisions its infrastructure.

Let us now highlight the severely underprovisioned scenario with only 15 reflectors. Again, SCADOOSH demonstrates its potential: 70% of users experience no degradation at all. Moreover, only 17% (respectively 9%) of users have a satisfaction below 0.88 (respectively 0.75). With respect to the severe underprovisioning of this infrastructure, these results are noteworthy since it shows that the population of users is reasonably well served although the CDN provisions less than half of the required infrastructure.

We now have a closer look at the edge servers where we distinguish into the three families of edge servers. See Figure 5.13. As can be expected, end users from the `mobi` family are almost not impacted by the underprovisioning. Indeed, the `uspru` of these clients is the highest. On the contrary, end-users from `HD` as well as `norm` are the ones that are the most affected by the underprovisioning. It is because a very under-provisioned CDN network cannot afford to deliver the high bandwidth consuming HD representations. In fact, in the most underprovisioned CDN network with 15 reflectors, for `HD` edge servers, 51% of users can receive their required HD representations, and 29% percent of HD users receive the representation just below their required ones. This slight video quality degradation explains why globally users still perceive high satisfaction.

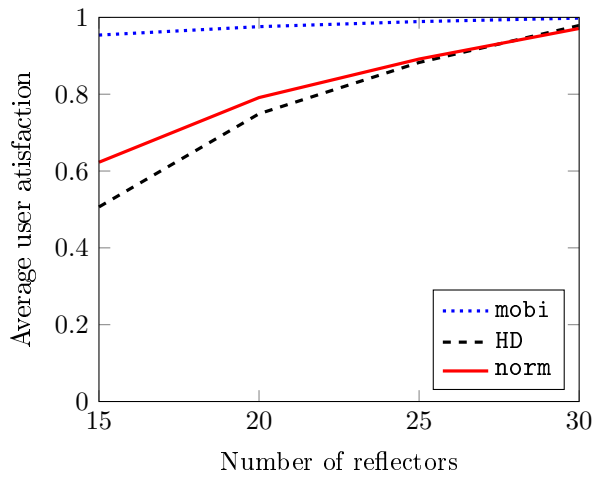


Figure 5.13: Average satisfaction of users regarding to each edge server family

Impact of Type-Specified User Assignment

SCADOOSH includes three components: user assignment, content placement and tree delivery. We would like first to evaluate the importance of the former algorithm, user assignment. Our proposal is to assign users to edge servers based not only on the channels they watch, but also on the type of device they use. As previously said, more accurate algorithms can be developed. Our goal here is to see whether such very simple implementation can already ensue in a gain of performances.

We refer to the type-specified user assignment mechanism proposed in Section 5.5.1 as `SPC`. We compare it to a strategy, which we refer to as `GEN`, that is more commonly used in CDN: it first tries to assign a user to an edge server that hosts the same channel within the same area with traffic load lower than the predefined threshold; otherwise, the user is assigned to the edge server within the same area with the lowest traffic load.

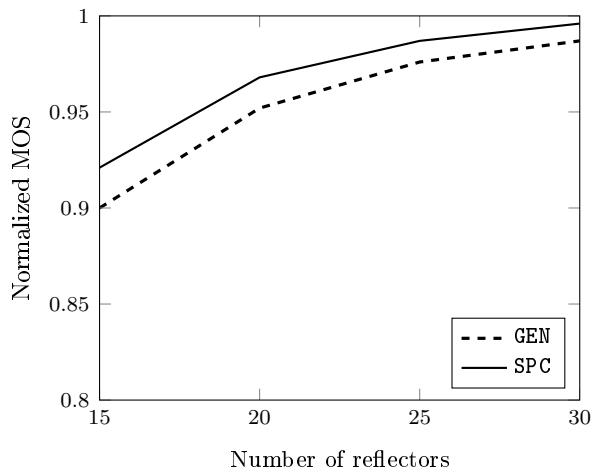
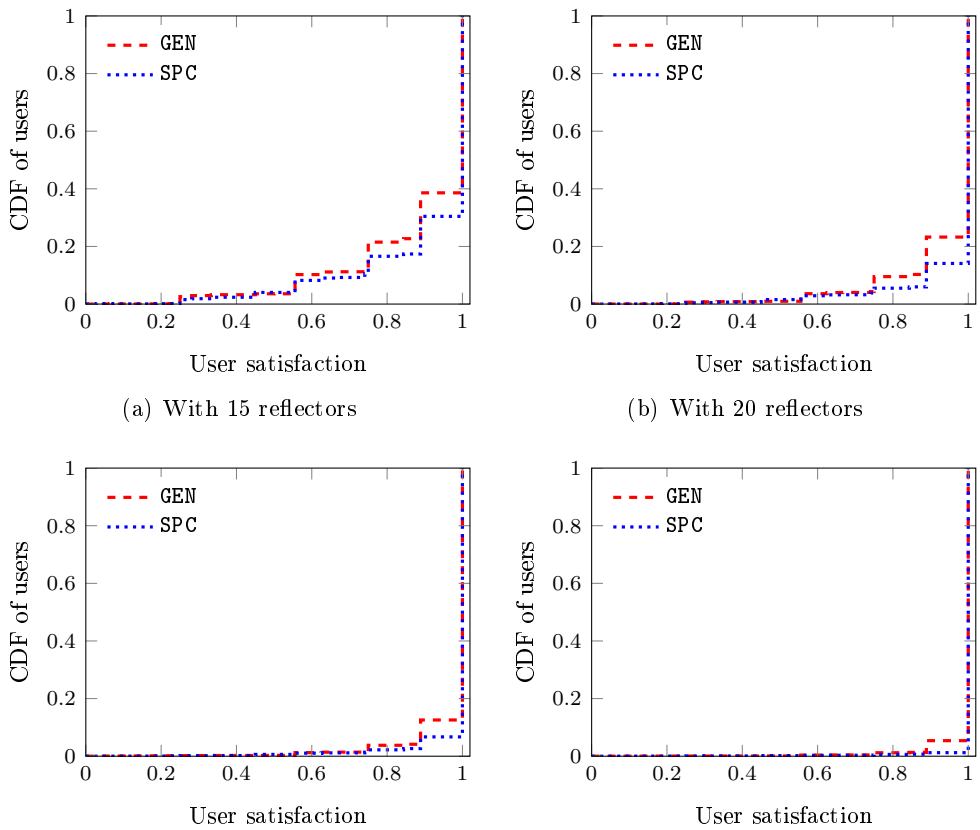


Figure 5.14: Average satisfaction of users: `GEN` vs. `SPC`



(a) With 15 reflectors

(b) With 20 reflectors

(c) With 25 reflectors

(d) With 30 reflectors

Figure 5.15: CDF of user satisfaction: `GEN` vs. `SPC`

We plot in Figure 5.14 the average satisfaction of users for both user assignment algorithms. As expected, with type specified user assignment, end users could obtain higher satisfaction with higher video quality. In order to show more details in user satisfaction levels, we show in Figure 5.15 the CDF of users on satisfaction. The gains of the type specified user assignment are more obvious. We can observe that `SPC` is able to deliver the best representation to a large subset of the population even when the system is very underprovisioned (with 15 reflectors): 38% of the population instead of 30%. With regard to the low complexity of the `SPC` algorithm, such gain can justify its implementation.

5.7 Conclusion

This Chapter further develops the work presented in Chapter 4 toward the user-centric delivery of live rate adaptive videos in CDN. It provides both theoretical and practical contributions.

From a theoretical perspective, our main contribution is the formulation of an optimization problem using the user-centric discretized streaming model. In this

problem, we maximize the satisfaction of users subject to the underprovisioning of CDN infrastructure. We formulate an Integer Linear Program (ILP), which jointly decides the choice of representations that should be sent to the edge servers, the building of a delivery overlay, and the assignment of end users to edge servers. We show that previous work related to streaming capacity do not satisfactorily address the problem met by CDN with rate-adaptive streaming, such as DASH.

Our second contribution is a practical system, named SCADOOSH. This system, which is inspired by our theoretical analysis, consists of three components. We evaluate the performances of SCADOOSH through a dynamic large-scale trace based simulation. We show that SCADOOSH can maintain a good QoE although the CDN infrastructure is severely underprovisioned in dynamic and large-scale CDNs.

The work presented in this chapter is still in progress. We have submitted this work to a journal, however, currently we have not obtained any valid publication. For future work, we envision the following possible directions. On one hand, the optimization problem requires a more comprehensive theoretical analysis. On the other hand, in a more practical perspective, SCADOOSH is a first step, which deserves further studies. We illustrated that the gains of each of the three algorithms are not always remarkable. Thus, one possible motivation is to find better algorithms, with respect to the necessary trade-off between simplicity of implementation and practical benefits. One of the most interesting challenge is to revisit forecasting algorithm so that the multi-representations feature of rate-adaptive streaming is actually taken into account. Another challenge is to determine some network topologies on top of which the QoE can be even better for a given underprovisioning. Finally, it would be extremely interesting to implement SCADOOSH on a real-world CDN infrastructure.

Chapter 6

Conclusion

The popularity of bandwidth-intensive online video services makes the Internet traffic increase faster than the capacity of infrastructures. One of the consequences is that the network is strained to its limits, and can be frequently underprovisioned. For live streaming, which is delay-sensitive, the provisioning of the delivery network has a direct impact on the system performance. Although live streaming has been widely addressed in the literature over the past decade, underprovisioning has not received enough attention in the literature, despite the importance of this issue. It is either ignored, or solved by deploying some additional bandwidth helpers to prevent the system to become underprovisioned.

In this thesis, we study bandwidth efficient real-time video delivery solutions for two popular live video delivery technique: P2P and CDN. We design a multioverlay P2P video sharing system that enables casual Internet users to stream their own live videos. Moreover, we solve the bandwidth allocation problem which is to minimize the waste of user upload bandwidth. Especially, we propose several bandwidth allocation strategies for underprovisioned system. For CDN live streaming, we focus on rate adaptive streaming techniques. For this topic, we propose both theoretical streaming model and practical implementable system. In the following, we first summarize the contributions of this thesis. Then, we will discuss the limitations of this work, and propose for possible future directions.

6.1 Synopsis

The main contributions in this thesis are summarized as follows:

- **Optimum bandwidth allocation in multioverlay P2P system.** We first investigate the bandwidth allocation problem in a multioverlay P2P system. In our opinion, the problem of underprovisioned multioverlay P2P systems has not received enough attention. In order to mitigate resource deficit, the goal is to maximally utilize end user's upload bandwidth. This goal corresponds to minimizing the resource waste when upload bandwidth resources are allocated to overprovisioned overlays although they could be allocated to underprovisioned ones. We show that such optimum bandwidth allocation corresponds to the

maximum flow in a transformed bipartite flow network. Further, we design several bandwidth allocation strategies that balance the resource deficit among the overlays for globally underprovisioned system. These strategies allow service providers to provision the overlays according to their business policies: prioritize some specific overlays, or fairly share the deficit among all the overlays. Finally, the bandwidth allocation algorithms are validated through extensive simulations. We also show the system can cope with dynamics through a set of dynamic scenario simulations.

- **Discretized streaming model for live rate adaptive streaming.** As far as we know, the discretized streaming model is the first work related to live rate adaptive streams delivery in the CDN infrastructure. This part is the basic theoretical contribution for this problem. We first define a general optimization problem which captures the current main concerns of CDN providers: maximizing the throughput of the CDN delivery network by maximizing the utility of delivered streams. Especially, this model allows to prioritize the transmission of streams (measured by the utility) in underprovisioned CDN infrastructure. We formulate this problem through Integer Linear Programming, and prove that the complexity of the problem is NP-Complete. This general problem is further developed into a user-centric one by defining user QoE related utility. The user-centric discretized streaming model maximizes the overall satisfaction of a population, and at the same time guarantees max-min fairness on user satisfaction. The evaluation of the model in a set of toy-CDN infrastructures shows the benefits of the user-centric discretized streaming on achieving higher user satisfaction comparing to the previous approaches.

- **A fast near-optimum algorithm for live rate adaptive transmission in CDN.** As the aforementioned NP-Completeness indicates, it is impossible to quickly find an optimal solution for the general discretized streaming capacity problem. The first practical contribution for live rate adaptive streaming in CDN is a fast near-optimum algorithm for a specific scenario, which corresponds to today's CDN implementation of live streams. Specifically, the CDN provider is in charge of delivering groups of representations as bundles. We provide formal theoretical approximation bound, which is at least $1 - (b_{s^*}/|V_E|)$ times the optimal solution (where b_{s^*} is the capacity of the source, and $|V_E|$ denotes the number of edge servers in the network). This is the first practical result for the discretized streaming model.

- **A practical CDN system for live rate adaptive streaming.** The second practical contribution for live rate adaptive streaming in CDN is an implementation of a system, named SCADOOSH, which enables a CDN provider to efficiently deliver live rate adaptive streams in a large-scale and dynamic environment. SCADOOSH redesigned the three CDN fundamental algorithms: content placement, content delivery and user redirection. At last, the performance of SCADOOSH is validated through a dynamic large-scale trace based simulation. We show that SCADOOSH can maintain a good Quality of Experience although the CDN infrastructure is severely underprovisioned in dynamic and large-scale

CDNs.

Besides the above main contributions, we also made minor contributions in some fellow scientists during the study of the thesis. In [LS13b], another fast near-optimum delivery trees construction algorithm is proposed for one specific application scenario of the generalized discretized streaming capacity problem. In this scenario, the CDN is fully connected and the CDN provider deploys a network with homogenous equipments with a uniform equipment capacity C. We also theoretically prove that the algorithm can obtain an approximation ratio of $1 - \frac{2\lambda^*}{C} - \frac{kl}{|V_R|}$, where λ^* denotes the maximum representation bit-rate, k (l, respectively) represents the number of representations (channels respectively), and $|V_R|$ is the number of reflectors. Numerical experiments demonstrate that for large CDN instances, the algorithm obtains nearly optimum result (*e.g.* approximate ratio of 0.993 for CDN with 5,000 reflectors) within very short time (30 seconds).

In [ZLSB13], we investigated the delivery of live video channels in the so-called Telco-CDN—CDN deployed within the ISP domain. Telco-CDN can be regarded as an intra-domain overlay network with tight resources and critical deployment constraints. This paper addresses two problems in this context: (1) the construction of the overlays used to deliver the video channels from the entrypoints of the Telco-CDN to the appropriate edge servers; and (2) the allocation of the required resources to these overlays. Our ultimate goal is to maximize the number of delivered channels while preserving network resources. To achieve this goal, two approaches are proposed: (1) A joint optimization where both optimization problems are simultaneously addressed; and (2) a two-step optimization where the optimal overlays are firstly computed, then an optimal resource allocation based on these pre-computed overlays is performed. We also devise heuristic algorithms for each of these approaches. The conducted evaluations of these two approaches and algorithms provide useful insights into the management of critical Telco-CDN infrastructures.

6.2 Limitations and Perspectives

In this thesis we contribute to research in improving bandwidth utilization of live video streaming for both P2P-based and CDN-based systems. But these works admit some limitations, which have to be addressed in future works.

- **Decentralize the multioverlay P2P system.** The algorithms that we implemented for the multioverlay P2P system rely on a centra actor (the management server) and centralized bandwidth allocation algorithms. The design of a decentralized multioverlay P2P system would be more appropriate with respect to the distributed nature of P2P systems. A key point in such a design is developing distributed bandwidth allocation algorithms. Currently, only one distributed algorithm has been designed for the fairness-based strategy. From the evaluation of the algorithms, the improved minimum-cost maximum-flow strategies have the highest performance in terms of user perceived video quality. Thus, a reasonable further step of the work includes designing distributed algorithms that optimizes these strategies.

- **Optimum algorithms for the discretized streaming model.** The discretized streaming model provides a theoretical foundation for multiple live rate adaptive videos delivery in today's CDNs. This topic should deserve further study. In this thesis, we proposed a fast near-optimum algorithm for one specific scenario. It would be also interesting to determine families of network infrastructures on top of which fast optimal delivery algorithms can be built. This future work can be integrated in the live rate adaptive streaming CDN system (SCADOOSH) as delivery forest construction algorithms.

- **User QoE-based rate adaptive streaming.** In the current work, we introduced a model to roughly estimate user QoE on perceived representations. This model is a generic one that could utilize any existing work on estimating QoE from various video bit rates. We envisioned two possible future work toward user QoE-based rate adaptive streaming. First, the lack of accuracy calls for more dedicated user QoE estimation models that take more parameters in input: for example, video resolution, video types, etc. Such models can provide more precise estimations on user QoEs to enable content providers and CDN providers to improve their delivery strategy. Second, the bit rates of representations are currently determined by the content provider in a somewhat arbitrary manner. Since users can perceive different QoE on different bit rate of the videos, it would be interesting to determine the set of bit rates for representations based on user QoE values such that a certain optimization goal could be achieved.

- **Improving the live rate adaptive streaming CDN system.** The CDN system (SCADOOSH) that we proposed for live rate adaptive streaming is still at the earliest stages of development, and requires further improvement. For the following works, we envision improvement on each of the three components. The benefits of the type specified user redirection mechanism has been proved. For the next step, more technical details for implementations with balances between practicality and profitability are required. For the forecasting algorithm, the multi-representations feature of rate adaptive streaming should be taken into account. Then, efficient optimum trees construction algorithms should be also explored. At last, it would be very interesting to realize a true implementation of the SCADOOSH system.

More generally speaking, the underprovisioning of delivery network remains an open research topic that has not received enough attention so far. Hence this topic deserves further in-depth study.

First of all, provisioning well the delivery network in advance remains the major cost for service providers. Moreover the provisioning of the network is also influenced by dynamic user requirements: for the same delivery network, it could be overprovisioned under light service requirement, as well as underprovisioned under intensive service requirement. As a result, service providers could benefit from elastic resource provisioning which adaptively determine the size of the delivery network on the fly in response to temporal and spatial dynamics of service demands. In such work, changing the size of the network introduces extra costs, such as data migration, delay, etc.

Consequently, problems should be integrated in problem formulation for optimization purposes.

Finally, bandwidth efficient video delivery is not the only solution for mitigating the impact of system underprovisioning. Efforts could also be made in multiple directions: exploring new types of delivery network; increasing the capacity of the connection links; inventing new coding methods that could provide higher video quality with lower bandwidth consumption, and so on. In one word, scientists in all related domains should pay attention to the underprovisioning of the network.

Bibliography

[ABD11] Saamer Akhshabi, Ali C. Begen, and Constantine Dovrolis. An experimental evaluation of rate-adaptation algorithms in adaptive streaming over http. In *Proceedings of the second annual ACM conference on Multimedia systems*, MMSys '11, pages 157–168, 2011. 24

[ABH+11] Shakeel Ahmad, Christos Bouras, Raouf Hamzaoui, Jiayi Liu, Andreas Papazois, Erez Perelman, Gwendal Simon, and George Tsichritzis. Community tools for massively multiplayer online games. *International Journal on Advances in Networks and Services*, 4(3-4), Nov. 2011. xvii, 4, 18, 55

[ado] Adobe HTTP Dynamic Streaming. `http://www.adobe.com/products/hds-dynamic-streaming.html?promoid=GYMXT`. 25

[AEVW04] Jussara M. Almeida, Derek L. Eager, Mary K. Vernon, and Stephen J. Wright. Minimizing delivery cost in scalable streaming content distribution systems. *IEEE Trans. on Multimedia*, 6(2):356–365, 2004. xx, 7, 20, 27

[AGH+12] Vijay Kumar Adhikari, Yang Guo, Fang Hao, Matteo Varvello, Volker Hilt, Moritz Steiner, and Zhi-Li Zhang. Unreeling netflix: Understanding and improving multi-cdn movie delivery. In *IEEE INFOCOM*, pages 1620–1628, 2012. xviii, 5, 25, 27

[AGM+05] Austin Armbruster, Michael Gosnell, Bruce McMillin, Mariesa Crow, and Senior Member. The maximum flow algorithm applied to the placement and distributed steady-state control of facts devices. In *Control of FACTS Devices, Proceedings of the 2005 North American Power Symposium*, 2005. 34

[AJZ11] Vijay Kumar Adhikari, Sourabh Jain, and Zhi-Li Zhang. Where do you "tube"? uncovering youtube server selection strategy. In *IEEE ICCCN*, pages 1–6, 2011. 21, 85

[aka] Akamai HD Network. `http://www.akamai.com/html/misc/hdnetwork.html`. 25

[AMM+11] Konstantin Andreev, Bruce Maggs, Adam Meyerson, Jevan Saks, and Ramesh Sitaraman. Algorithms for constructing overlay networks for live streaming. *CoRR*, 2011. xiv, xviii, xx, 2, 5, 7, 20, 27, 57, 59

[AMMS03] Konstantin Andreev, Bruce M. Maggs, Adam Meyerson, and Ramesh K. Sitaraman. Designing overlay multicast networks for streaming. In *ACM SPAA*, 2003. xviii, xx, 5, 7, 20, 21, 27

[AMO93] Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network Flows: Theory, Algorithms, and Applications.* Prentice Hall, New-York, 1993. 35

[AOST94] Ravindra K. Ahuja, James B. Orlin, Clifford Stein, and Robert E. Tarjan. Improved algorithms for bipartite network flow. *SIAM J. Comput.*, 23(5):906–933, 1994. 34

[appa] HTTP Live Streaming Overview. `http://goo.gl/XFfa2`. 25

[appb] Apple: Using HTTP Live Streaming, `http://goo.gl/fJIwC`. 70, 75, 81, 93

[ASV11] Micah Adler, Ramesh K. Sitaraman, and Harish Venkataramani. Algorithms for optimizing the bandwidth cost of content delivery. *Computer Networks*, 55(18):4007–4020, 2011. xviii, xx, 5, 7, 20, 21, 27, 57, 59

[BAD+12] Eliya Buyukkaya, Shakeel Ahmad, Muneeb Dawood, Jiayi Liu, Fen Zhou, Raouf Hamzaoui, and Gwendal Simon. Level-based peer-to-peer live streaming with rateless codes. In *Proc. of IEEE Int. Symposium on Multimedia*, 2012. 55

[BB05] Christian Blum and Maria J. Blesa. New metaheuristic approaches for the edge-weighted -cardinality tree problem. *Computers & OR*, 32:1355–1377, 2005. 28

[BBK02] Suman Banerjee, Bobby Bhattacharjee, and Christopher Kommareddy. Scalable application layer multicast. In *Proceedings of the 2002 conference on Applications, technologies, architectures, and protocols for computer communications*, ACM SIGCOMM '02, pages 205–217, 2002. 13

[BJ90] George Edward Pelham Box and Gwilym Jenkins. *Time Series Analysis, Forecasting and Control.* Holden-Day, Incorporated, 1990. 87

[BSB+12] G. Bertrand, E. Stephan, T. Burbridge, P. Eardley, K. Ma, and G. Watson. Use Cases for Content Delivery Network Interconnection. draft at IETF CDN Interconnection, 2012. 64

[BV04] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization.* Cambridge University Press, New York, NY, USA, 2004. 39

[CDK+03] Miguel Castro, Peter Druschel, Anne-Marie Kermarrec, Animesh Nandi, Antony Rowstron, and Atul Singh. Splitstream: high-bandwidth multicast in cooperative environments. In *ACM SOSP*, pages 298–313, 2003. 14

[CHL06] K. Chen, P. Huang, and C. Lei. Game traffic analysis: an mmorpg perspective. *Computer Networks*, 50(16):3002–3023, 2006. 51

[cisa] Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2012-2017. `http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-520862.pdf`. xiv, xv, 2, 3, 23

[cisb] Cisco Visual Networking Index: Forecast and Methodology, 2010-2015. `http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-481360_ns827_Networking_Solutions_White_Paper.html`. xiv, 2

[CM10] Luca De Cicco and Saverio Mascolo. An experimental investigation of the akamai adaptive video streaming. *Lecture Notes in Computer Science*, 6389:447–464, 2010. 23

[CMP11] Luca De Cicco, Saverio Mascolo, and Vittorio Palmisano. Feedback control for adaptive live video streaming. In *ACM MMSys*, 2011. 27

[CNG] The cng project. `http://www.cng-project.eu/`. 6, 54

[Con] Conviva. Viewer experience report. `http://www.conviva.com/vxr/`. xviii, 5

[CRZ00] Yang-Hua Chu, Sanjay G. Rao, and Hui Zhang. A case for end system multicast. In *ACM SIGMETRICS*, pages 1–12, 2000. 13

[CT04] Jae-Hwan Chang and Leandros Tassiulas. Maximum lifetime routing in wireless sensor networks. *IEEE/ACM Trans. Netw.*, 12(4), 2004. 90

[das] Overview of MPEG-DASH Standard, `http://dashif.org/mpeg-dash/`. xv, 3

[DC90] Stephen E. Deering and David R. Cheriton. Multicast routing in datagram internetworks and extended lans. *ACM Trans. Comput. Syst.*, 8(2):85–110, 1990. xiii, 1

[DMP+02] John Dilley, Bruce Maggs, Jay Parikh, Harald Prokop, Ramesh Sitaraman, and Bill Weihl. Globally distributed content delivery. *IEEE Internet Computing*, 6(5):50–58, 2002. 20

[DN08] Sachin Deshpande and Jeonghun Noh. P2tss: Time-shifted and live streaming of video in peer-to-peer systems. In *Proc. of the IEEE International Conference on Multimedia and Expo (ICME)*, 2008. 11

[FBS07] W. Feng, D. Brandt, and D. Saha. A long-term study of a popular mmorpg. In *Proc. of ACM Netgames*, 2007. 51

[FJL+95] Sally Floyd, Van Jacobson, Ching-Gung Liu, Steven Mccanne, and Lixia Zhang. A reliable multicast framework for light-weight sessions and application level framing. In *IEEE/ACM TRANSACTIONS ON NETWORKING*, pages 342–356, 1995. xiii, 1

[geo] GeoDNS BIND patch, `http://www.caraytech.com/geodns/`. 85

[GGSP95] Sukumar Ghosh, Arobinda Gupta, Sriram, and Sriram V. Pemmaraju. A self-stabilizing algorithm for the maximum flow problem. *Distributed Computing*, 10:8–14, 1995. 34

[Goe06] Michel X. Goemans. Minimum bounded degree spanning trees. In *IEEE FOCS*, 2006. 28

[Gol97] A. V. Goldberg. An efficient implementation of a scaling minimum-cost flow algorithm. *Journal of Algorithms*, 22:1–29, 1997. 42

[GT88] Andrew V. Goldberg and Robert E. Tarjan. A new approach to the maximum flow problem. *Journal of the ACM*, 35:921–940, 1988. 34

[GT89] Andrew V. Goldberg and Robert E. Tarjan. Finding minimum-cost circulations by canceling negative cycles. *J. ACM*, 36(4):873–886, 1989. 35

[HA+07] Mojtaba Hosseini, , Dewan T. Ahmed, Shervin Shirmohammadi, and Nicolas D. Georganas. A survey of application-layer multicast protocols. *IEEE Com. Surveys & Tut.*, 9(3):58–74, 2007. 28

[HBC+11] Fabio Victora Hecht, Thomas Bocek, Richard G Clegg, Raul Landa, David Hausheer, and Burkhard Stiller. Liveshift: Mesh-pull live and time-shifted p2p video streaming. In *Proc. of the 36th Annual IEEE Conference on Local Computer Networks, (LCN)*, 2011. 11

[hCRSZ00] Yang hua Chu, Sanjay G. Rao, Srinivasan Seshan, and Hui Zhang. A case for end system multicast. In *in Proceedings of ACM Sigmetrics*, pages 1–12, 2000. xiv, 2

[HLL+07] Xiaojun Hei, Chao Liang, Jian Liang, Yong Liu, and Keith W. Ross. A measurement study of a large-scale p2p iptv system. *IEEE Transactions on Multimedia*, 9, 2007. 17

[IIS] Smooth Streaming. `http://www.iis.net/downloads/microsoft/smooth-streaming`. 25

[Ing12] Mathew Ingram. You think the internet is big now? akamai needs to grow 100-fold. Om Malik, Jun 2012. `http://is.gd/3vTZPC`. xviii, 5

[JDXB03] Xuxian Jiang, Yu Dong, Dongyan Xu, and Bharat Bhargava. Gnustream: A p2p media streaming system prototype. In *In Proceedings of the International Conference on Multimedia and Expo (ICME*, pages 325–328, 2003. 13

[JGJ+00] John Jannotti, David K. Gifford, Kirk L. Johnson, M. Frans Kaashoek, and James W. O'Toole, Jr. Overcast: reliable multicasting with on overlay network. In *Proceedings of the 4th conference on Symposium on Operating System Design & Implementation - Volume 4*, OSDI'00, 2000. 13

[JSZ12] Junchen Jiang, Vyas Sekar, and Hui Zhang. Improving fairness, efficiency, and stability in http-based adaptive video streaming with festive. In *CoNEXT*, pages 97–108, 2012. 21

[jus] justin.tv. `http://www.justin.tv/`. xiii, xv, 1, 2, 74, 88, 92

[KR05] Jochen Könemann and R. Ravi. Primal-dual meets local search: approximating MSTs with nonuniform degree bounds. *SIAM J. Comput.*, 34(3):763–773, 2005. 28

[Kro11] Bill Krogfoss. Analysis: Content peering and the internet economy. Technical report, Alcatel Lucent, April 2011. xx, 7

[KS11] Joohwan Kim and R. Srikant. Achieving the maximum p2p streaming rate using a small number of trees. In *IEEE ICCCN*, 2011. xx, xxi, 7, 8

[KSC+12] Mehdi Kaytoue, Arlei Silva, Loïc Cerf, Wagner Meira, Jr., and Chedy Raïssi. Watch me playing, i am a professional: a first study on video game live streaming. In *Proceedings of the 21st international conference companion on World Wide Web*, 2012. 46

[KSJI10] A. Khan, L. Sun, E. Jammeh, and E. Ifeachor. Quality of experience-driven adaptation scheme for video applications over wireless networks. *Communications, IET*, 4(11):1337–1347, 2010. 75

[KSW+04] L. Kontothanassis, R. Sitaraman, J. Wein, D. Hong, R. Kleinberg, B. Mancuso, D. Shaw, and D. Stodolsky. A transport layer for live streaming in a content delivery network. *Proceedings of the IEEE*, 92(9):1408–1419, 2004. xiv, 2

[LAB+12] Jiayi Liu, Shakeel Ahmad, Eliya Buyukkaya, Raouf Hamzaoui, and Gwendal Simon. Resource allocation in underprovisioned multioverlay live video sharing services. In *Proc. of ACM CoNEXT Workshop on Capacity Sharing*, 2012. 54

[LBHG12] Chenghao Liu, Imed Bouazizi, Miska M. Hannuksela, and Moncef Gabbouj. Rate adaptation for dynamic adaptive streaming over http in content distribution network. *Signal Processing: Image Communication*, 27(4):288 – 311, April 2012. 24, 27

[Le 12] Francois Le Faucheur. CDN Federations: Lessons from the CDN Federation Pilot Phase 2. In *CDN Summit*, 2012. 64

[LEF+11] Thorsten Lohmar, Torbjorn Einarsson, Per Frojdh, Frederic Gabin, and Markus Kampmann. Dynamic adaptive http streaming of live content. In *WOWMOM*, pages 1–8, 2011. 26

[LGL08] Yong Liu, Yang Guo, and Chao Liang. A survey on peer-to-peer video streaming systems. *Peer-to-Peer Networking and Applications*, 1(1):18–28, 2008. xiv, 2

[LS10] Yaning Liu and Gwendal Simon. Distributed delivery system for time-shifted streaming systems. In *Proc. of the 35th Annual IEEE Conference on Local Computer Networks, (LCN)*, 2010. 11

[LS11] Yaning Liu and Gwendal Simon. Peer-assisted time-shifted streaming systems: Design and promises. In *Proc. of IEEE International Conference on Communications (ICC)*, 2011. 11

[LS13a] Jiayi Liu and Gwendal Simon. Fast near-optimal algorithm for delivering multiple live video streams in cdn. In *Proc. of 22nd IEEE ICCCN*, 2013. 72

[LS13b] Jiayi Liu and Gwendal Simon. Fast near-optimal algorithm for delivering multiple live video streams in cdn. In *Proc. of 15th French Conference ALGOTEL*, 2013. 101

[LSR+09] Zhengye Liu, Yanming Shen, Keith W. Ross, Shivendra S. Panwar, and Yao Wang. Layerp2p: Using layered video chunks in p2p live streaming. *IEEE Transactions on Multimedia*, 11(7):1340–1352, 2009. 18

[LSYR11] Nikolaos Laoutaris, Michael Sirivianos, Xiaoyuan Yang, and Pablo Rodriguez. Inter-datacenter bulk transfers with netstitcher. In *ACM SIGCOMM*, 2011. xviii, 5, 12

[Lub02] Michael Luby. Lt codes. In *FOCS*, 2002. 14

[LXQ+08] Bo Li, Susu Xie, Yang Qu, Gabriel Yik Keung, Chuang Lin, Jiangchuan Liu, and Xinyan Zhang. Inside the new coolstreaming: Principles, measurements and performance implications. In *INFOCOM*, pages 1031–1039, 2008. 17

[LZL11] Chao Liang, Miao Zhao, and Yong Liu. Optimal bandwidth sharing in multiswarm multiparty p2p video-conferencing systems. *IEEE/ACM Trans. Netw.*, 19(6):1704–1716, 2011. 17, 18, 19

[MHXW12] Zhan Ma, Hao Hu, Meng Xu, and Yao Wang. Rate model for compressed video. *CoRR*, abs/1206.2625, 2012. 75

[mis] The Missing Ink MMOG, `http://www.missing-ink.com/`. 54

[mKPB06] Al mukaddim Khan Pathan and Rajkumar Buyya. A taxonomy and survey of content delivery networks. Technical report, 2006. xiv, 2

[MLCC12] Ricky K. P. Mok, Xiapu Luo, Edmond W. W. Chan, and Rocky K. C. Chang. Qdash: a qoe-aware dash system. In *Proceedings of the 3rd Multimedia Systems Conference*, MMSys '12, pages 11–22, 2012. 24

[MRG07] Nazanin Magharei, Reza Rejaie, and Yang Guo. Mesh or multiple-tree: A comparative study of live p2p streaming approaches. In *IEEE INFOCOM*, pages 1424–1432, 2007. 16

[MZÅ11] Karan Mitra, Arkady B. Zaslavsky, and Christer Åhlund. A probabilistic context-aware approach for quality of experience measurement in pervasive systems. In *Proc. of ACM SAC*, 2011. 75, 76

[nbc] Robert Andrews. NBC: Nearly half of Olympics streams are from mobile, table. `http://paidcontent.org/2012/08/02/nbc-nearly-half-of-olympics-streams-are-from-mobile-tablet/`. xv, 3, 23, 81, 92

[NBC12] NBC. Super Bowl XLVI Live Stream Sets Traffic Records. Technical report, NBC Sport Press Release, Feb. 2012. xvii, 4

[net] Netflix Open Connect Peering Guidelines, `http://goo.gl/GQ6NU`. xx, 7, 58

[NL11] Di Niu and Baochun Li. Asymptotic optimality of randomized peer-to-peer broadcast with network coding. In *IEEE INFOCOM*, 2011. xx, xxi, 7, 8, 27, 57

[NSS10] Erik Nygren, Ramesh K. Sitaraman, and Jennifer Sun. The Akamai network: a platform for high-performance internet applications. *Op. Sys. Rev.*, 44(3):2–19, 2010. xiv, xx, 2, 7, 20, 27, 60

[NT05] Jian Ni and Danny H. K. Tsang. Large-scale cooperative caching and application-level multicast in multimedia content delivery networks. *IEEE Com. Mag.*, 43(5):98 – 105, 2005. 20

[P2P] P2ptvsim. `http://napa-wine.eu/cgi-bin/twiki/view/Public/P2PTVSim`. 44

[Pas12] Andrea Passarella. A survey on content-centric technologies for the current internet: Cdn and p2p solutions. *Computer Communications*, 35(1):1–32, 2012. xix, 7, 17, 20

[PB12] Louis Plissonneau and Ernst Biersack. A longitudinal view of http video streaming performance. In *ACM MMSys*, pages 203–214, 2012. 21, 85

[PC07] Daniel P. Palomar and Mung Chiang. Alternative distributed algorithms for network utility maximization: Framework and applications. *IEEE Transactions on Automatic Control*, 52:2254–2269, 2007. 38, 39

[ppl] PPLive. `http://www.pptv.com/`. xiii, xiv, 1, 2, 13

[PSLB06] S. Paul, K. K. Sabnani, J. C.-H. Lin, and S. Bhattacharyya. Reliable multicast transport protocol (rmtp). *IEEE J.Sel. A. Commun.*, 15(3):407–421, 2006. xiii, 1

[PSVW01] Dimitrios Pendarakis, Sherlia Shi, Dinesh Verma, and Marcel Waldvogel. Almi: an application level multicast infrastructure. In *Proceedings of the 3rd conference on USENIX Symposium on Internet Technologies and Systems*, ACM USITS'01, pages 5–5, 2001. 13, 14

[RBS12] M. Reza Rahimi, Abdul Bais, and Nima Sarshar. On fair and optimal multi-source ip-multicast. *Computer Networks*, 2012. 27

[RCKS10] A.B. Reis, J. Chakareski, A. Kassler, and S. Sargento. Distortion optimized multi-service scheduling for next-generation wireless mesh networks. In *INFOCOM IEEE Conference on Computer Communications Workshops , 2010*, pages 1–6, 2010. 75

[san] Sandvine. Global Internet Phenomena Report, 2012. `http://www.sandvine.com/downloads/documents/Phenomena_2H_2012/Sandvine_Global_Internet_Phenomena_Report_2H_2012.pdf`. xiv, 2

[Sho06] Amin Shokrollahi. Raptor Codes. *IEEE Transactions on Information Theory*, 52:2551–2567, 2006. 14

[SI11] S. Shen and A. Iosup. The xfire online meta-gaming network: Observation and high-level analysis. In *Workshop MMVE*, 2011. xv, 3, 45, 46

[SIB12] Raymond Sweha, Vatche Ishakian, and Azer Bestavros. AngelCast: Cloud-based Peer-Assisted Live Streaming Using Optimized Multi-Tree Construction. In *ACM MMSys*, 2012. xvii, 5, 19, 27

[SLC+11] Sudipta Sengupta, Shao Liu, Minghua Chen, Mung Chiang, Jin Li, and Philip A. Chou. Peer-to-peer streaming capacity. *IEEE Trans. on Information Theory*, 57(8):5072–5087, 2011. xx, xxi, 7, 8, 27, 82

[Sop] SopCast. `http://www.sopcast.org`. 13

[Sto11] Thomas Stockhammer. Dynamic adaptive streaming over http - standards and design principles. In *ACM MMSys*, pages 133–144, 2011. xv, 3, 23, 24, 26

[Tar85] Éva Tardos. A strongly polynomial minimum cost circulation algorithm. *Combinatorica*, 5(3):247–255, 1985. 35

[tel] Tellabs STL Reports, `http://info.tellabs.com/smartpipes`. 85

[TFK+11a] Ruben Torres, Alessandro Finamore, Jin Ryong Kim, Marco Mellia, Maurizio M. Munafo, and Sanjay Rao. Dissecting Video Server Selection Strategies in the YouTube CDN. In *IEEE ICDCS*, 2011. 85

[TFK+11b] Ruben Torres, Alessandro Finamore, Jin Ryong Kim, Marco Mellia, Maurizio M. Munafò, and Sanjay G. Rao. Dissecting video server selection strategies in the youtube cdn. In *ICDCS*, pages 248–257, 2011. 21

[TKS10] Srisakul Thakolsri, Wolfgang Kellerer, and Eckehard Steinbach. Qoe-based rate adaptation scheme selection for resource-constrained wireless video transmission. In *Proceedings of the international conference on Multimedia*, 2010. 75

[TL12] Guibin Tian and Yong Liu. Towards agile and smooth video adaptation in dynamic http streaming. In *CoNEXT*, pages 109–120, 2012. 21

[TMP11] A. L. Traud, P. J. Mucha, and M. A. Porter. Social structure of facebook networks. *CoRR*, abs/1102.2166, 2011. 46

[ust] ustream. `http://www.ustream.tv/`. xv, 3

[UUS] UUSee. `http://www.uusee.com`. 13

[WBS+09] Christo Wilson, Bryce Boe, Ra Sala, Krishna P. N. Puttaswamy, and Ben Y. Zhao. User interactions in social networks and their implications. 2009. 45, 46

[WL07] Chuan Wu and Baochun Li. Strategies of conflict in coexisting streaming overlays. In *INFOCOM*, pages 481–489, 2007. 17, 18, 19

[WLC12] Feng Wang, Jiangchuan Liu, and Minghua Chen. Calms: Cloud-assisted live media streaming for globalized demands with time/region diversities. In *IEEE INFOCOM*, pages 199–207, 2012. 20, 87

[WLL08] Chuan Wu, Baochun Li, and Zongpeng Li. Dynamic bandwidth auctions in multioverlay p2p streaming with network coding. *IEEE Trans. Parallel Distrib. Syst.*, 19(6):806–820, 2008. xix, 6, 17, 18, 19, 31

[WLLR10] Di Wu, Chao Liang, Yong Liu, and Keith W. Ross. Redesigning multichannel p2p live video systems with view-upload decoupling. *Computer Networks*, 54(12):2007–2018, 2010. 19

[WLZ08] Chuan Wu, Baochun Li, and Shuqiao Zhao. Multi-channel live p2p streaming: Refocusing on servers. In *Proc. of IEEE Infocom*, pages 1355–1363, 2008. 19

[WLZ11a] Chuan Wu, Baochun Li, and Shuqiao Zhao. On Dynamic Server Provisioning in Multichannel P2P Live Streaming. *IEEE/ACM Trans. Netw.*, 19(5):1317–1330, 2011. xvii, 5

[WLZ11b] Chuan Wu, Baochun Li, and Shuqiao Zhao. On Dynamic Server Provisioning in Multichannel P2P Live Streaming. *IEEE/ACM Trans. Netw.*, 19(5), 2011. 20, 31, 87, 88, 89

[WRL05] Yao Wang, A.R. Reibman, and Shunan Lin. Multiple description coding for video delivery. pages 57–70, 2005. 15

[WXR09] Miao Wang, Lisong Xu, and Byrav Ramamurthy. A flexible divide-and-conquer protocol for multi-view peer-to-peer live streaming. In *IEEE P2P*, 2009. xix, 6, 17, 19, 20, 31, 41, 43

[WXR11] Miao Wang, Lisong Xu, and Byrav Ramamurthy. Improving multi-view peer-to-peer live streaming systems with the divide-and-conquer strategy. *Computer Networks*, 55(18):4069–4085, 2011. xvii, xix, 4, 6, 17, 19, 20, 31, 41

[xfi] xfire. `http://www.xfire.com/`. xv, 3

[XLKZ07] Susu Xie, Bo Li, G. Y. Keung, and Xinyan Zhang. Coolstreaming: Design, theory, and practice. *Trans. Multi.*, 9(8):1661–1671, 2007. 17

[YH08] Kazuhisa Yamagishi and Takanori Hayashi. Parametric packet-layer model for monitoring video quality of iptv services. In *IEEE ICC*, pages 110–114, 2008. 75, 77

[YV07] Lu Yan and Sebastien Venot. Peer-to-peer media streaming application survey. In *Proceedings of the International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies*, UBICOMM '07, pages 139–148, 2007. xiv, 2

[ZAB+12] Fen Zhou, Shakeel Ahmad, Eliya Buyukkaya, Gwendal Simon, and Raouf Hamzaoui. Minimizing Server Throughput for Low-Delay Live Streaming in Content Delivery Networks. In *ACM NOSSDAV*, 2012. xviii, 5, 21, 57, 59

[ZLLsPY05] Xinyan Zhang, Jiangchuan Liu, Bo Li, and Tak shing Peter Yum. Coolstreaming/donet: A data-driven overlay network for peer-to-peer live media streaming. In *IEEE INFOCOM*, 2005. 13

[ZLSB13] Fen Zhou, Jiayi Liu, Gwendal Simon, and Raouf Boutaba. Joint optimization for the delivery of multiple video channels in telco-cdn. In *Proc. of CNSM'13: 9th International Conference on Network and Service Management*, 2013. Accepted as short paper. 101

[ZLW11] Can Zhao, Xiaojun Lin, and Chuan Wu. The streaming capacity of sparsely-connected P2P systems with distributed control. In *IEEE INFOCOM*, 2011. xx, xxi, 7, 8, 27

[ZWCK13] W. Zhang, Y. Wen, Z. Chen, and A. Khisti. Qoe-driven cache management for http adaptive bit rate streaming over wireless networks. *IEEE Transactions on Multimedia*, 2013. 75

Publications

International Conference

[1] Jiayi Liu, Gwendal Simon, " Fast Near-Optimal Algorithm for Delivering Multiple Live Video Streams in CDN ", in *ICCCN'2013: IEEE International Conference on Computer Communications Networks*, 30 July - 2 August, Nassau, Bahamas, 2013.

[2] Fen Zhou, Jiayi Liu, Gwendal SIMON and Raouf Boutaba, " Joint Optimization for the Delivery of Multiple Video Channels in Telco-CDN ", in *CNSM'13: 9th International Conference on Network and Service Management.*

[3] Jiayi Liu, Shakeel Ahmad, Eliya Buyukkaya, Raouf Hamzaoui and Gwendal Simon, " Resource Allocation in Underprovisioned Multioverlay Live Video Services ", in *ACM CoNEXT Capacity Sharing Workshop*, 10 - 13 December, Nice, France, 2012.

[4] Eliya Buyukkaya, Shakeel Ahmad, Muneeb Dawood, Jiayi Liu, Fen Zhou, Raouf Hamzaoui and Gwendal Simon, " Level-based peer-to-peer live streaming with rateless codes ", in *ISM'2012: IEEE Int. Symposium on Multimedia*, 9 - 11 December, Irvine, USA, 2012.

National Conference

[1] Jiayi Liu, Gwendal Simon, " Fast Near-Optimal Algorithm for Delivering Multiple Live Video Streams in CDN ", in *ALGOTEL'2013*, 28 - 31 Mai, Pornic, France, 2013.

International Journal

[1] Jiayi Liu, Catherine Rosenberg, Gwendal SIMON and Geraldine Texier, " User-Centric Discretized Delivery of Rate-Adaptive Live Streams in Underprovisioned CDN Networks ", accepted in *IEEE Journal on Selected Areas in Communications (JSAC) Special Issue: Adaptive Media Streaming.*

[2] Shakeel Ahmad, Christos Bouras, Raouf Hamzaoui, Jiayi Liu, Adreas Papazois, Erez Perelman, Alex Shani, Gwendal Simon and George Tsichritzsis, " Commu-

nity Tools for Massively Multiplayer Online Games ", in *International Journal on Advances in Networks and Services*, 2011.

Working Paper

[1] Jiayi Liu, Shakeel Ahmad, Eliya Buyukkaya, Raouf Hamzaoui and Gwendal Simon, " Resource Allocation in Underprovisioned Multioverlay Live Video Services ", submitted to *Springer Journal of Peer-to-Peer Networking and Applications.*

[2] Jiayi Liu and Gwendal Simon, " Optimal Tree Packing Problems for Live Rate-Adaptive Streams Delivery ", submitted to *Networks.*

Glossary

–Number–	
3G	3rd Generation
3GPP	3rd Generation Partnership Project
–A–	
ADSL	Asymmetric Digital Subscriber Line
ARIMA	Autoregressive Integrated Moving Average
–B–	
BDST	Bounded Degree Spanning Tree
–C–	
CDF	Cumulative Distribution Function
CDN	Content Delivery Network
CIF	Common Intermediate Format
CNG	Community Network Game
CPU	Central Processing Unit
C/S	Client/Seriver
–D–	
DASH	Dynamic Adaptive Streaming over HTTP
DMAUS	Decision version of Maximum Average Utility Score
DNS	Domain Name System
–F–	
FTTH	Fiber To The Home
–G–	
GOP	Group of Picture

–H–	
HD	High Definition
HQ	High Quality
HTTP	Hypertext Transfer Protocol
–I–	
ILP	Integer Linear Programming
ISP	Internet Service Provider
–M–	
MAUS	Maximum Average Utility Score
MMOG	Massively Multiplayer Online Game
MOS	Mean Opinion Score
MPD	Media Presentation Description
MPEG	Moving Picture Experts Group
–N–	
NAT	Network Address Translation
–O–	
OSN	Online Social Network
–P–	
P2P	Peer-to-Peer
PSNR	Peak Signal to Noise Ratio
–Q–	
QoE	Quality of Experience
–S–	
SCADOOSH	SCAle Down fOOtprint for live daSH
SPOF	Single Point Of Failure
–T–	
TCP	Transmission Control Protocol
TV	Television
–U–	
UGC	User Generated Content
URL	Uniform Resource Locator
uspru	utility score per rate unit
UTC	Universal Time Clock
–V–	
VoD	Video on Demand

–X–

XML Extensible Markup Language