Adaptive Multicast Live Streaming for A/V Conferencing Systems over Software-Defined Networks

Christelle Al Hasrouty

THÈSE - THESIS

en cotutelle entre - with joint supervision between

Université de Bordeaux
&
University College Dublin

Par - By

CHRISTELLE AL HASROUTY

POUR OBTENIR LE GRADE DE - TO GET THE DEGREE OF

DOCTEUR - DOCTOR

INFORMATIQUE - COMPUTER SCIENCE

Adaptive Multicast Live Streaming for A/V Conferencing Systems over Software-Defined Networks

---

Co-encadrée par - Co-supervised by:

Prof.	Damien	Magoni	Université de Bordeaux
Prof.	John	Murphy	University College Dublin

Date de soutenance - Defense date: 04/12/2018

Devant la commission d'examen composée de - In front of the review board composed of:

Prof.	Colette	Johnen	Université de Bordeaux	Présidente/President
Prof.	Congduc	Pham	Université de Pau et des Pays de l'Adour	Rapporteur/Reviewer
Dr	Ramona	Trestian	Middlesex University London	Rapporteur/Reviewer
Dr	Declan	Delaney	University College Dublin	Examinateur/Examiner

– 2018 –

Declaration of Authorship

"I hereby certify that the submitted work is my own work, was completed while registered as a candidate for the degree of Doctor of Philosophy, and I have not obtained a degree elsewhere on the basis of the research presented in this submitted work."

Acknowledgements

Firstly, I would like to express my sincere gratitude to my supervisor, Prof. Damien Magoni from University of Bordeaux, for the continuous support of my PhD study and related research, for his patience, motivation, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis.

My sincere thanks also goes to my second supervisor Prof. John Murphy who provided me an opportunity to join his team as PhD student at University College Dublin, and who gave access to the laboratory and research facilities. Without his precious support it would not be possible to conduct this research.

Besides my supervisors, I would like to thank Dr. Mohamed Lamine Lamali, for his insightful comments and encouragement, but also for the hard question which allowed me to widen my research from various perspectives. I could not have imagined having a better advisor and mentor for my PhD study. I am also grateful to Dr. Cristian Olariu and Dr. Vincent Autefage for enlightening me the first glance of research.

I thank my fellow lab-mates in University of Bordeaux (Elyas Ben Hadj Yahia, Remi Delassus, Ema Falomir, Vincent Klein, Kendric Ruiz, Matthieu Barjon, Yessin Neggaz, Remi Laplace, Samah Bouzidi, Ema Falomir, Kilian Hett, Remi Giraud, Simon Da Silva, Rohan Fossé, Nicolas Herbaut ...) and University College Dublin (Takfarinas Saber, Bartlomiej Siniarski, Imane Brahimi, John Tobin, Shane Brady ...) for the stimulating discussions, for their support before every deadline, and for all the fun we have had in the last four years. Also I thank my friends (Karmen el Hajj, Georges Nader, Joyce Bou Sleiman, Adib ouayjan ...) for their moral support.

A very special gratitude goes out to all down at Lero, the Irish software research centre, and also IDEX Bordeaux for helping and providing the funding for the work.

I am also grateful to the following university staff: Cathy Roubineau and Isabelle Garcia for their unfailing support and assistance providing all the administrative documents.

Last but not the least, I would like to thank my family: my parents and to my sisters for supporting me spiritually throughout writing this thesis and my life in general.

Résumé en Français

Avec l'arrivée d'Internet comme alternative moins coûteuse aux réseaux de télécommunication privés, les systèmes de téléconférences ont suscité un grand intérêt parmi les individus et les entreprises. Cependant, ces systèmes devraient offrir les mêmes performances que les réseaux téléphoniques dédiés en ce qui concerne la qualité audio/vidéo. Or, les systèmes de téléconférence de haute qualité ont toujours été difficiles à réaliser en raison de leurs contraintes spécifiques sur le réseau et les exigences des utilisateurs, ainsi que leur hétérogénéité. De nombreux architectures et protocoles ont été adoptés par les systèmes de téléconférences afin de fournir une meilleure Qualité d'Expérience aux utilisateurs.

La plupart des entreprises utilisent des systèmes de téléconférences centralisés pour réduire les coûts. Ces systèmes permettent un excellent contrôle des appels, dû à l'utilisation d'un serveur central. Actuellement, l'architecture centralisée la plus utilisée pour les systèmes de téléconférence est basée sur l'unité de contrôle multipoint (*Multipoint Control Unit, MCU*). Le MCU est le serveur central dans ce système de téléconférence. Outre la fonctionnalité de contrôle, le MCU permet l'adaptation du trafic multimédia aux capacités hétérogènes des équipements des utilisateurs . Cependant, ce processus d'adaptation nécessite un temps de calcul élevé et une grande utilisation des ressources du MCU. En outre, le canal d'accès au MCU exige une bande passante importante afin d'éviter les goulots d'étranglement. Ces limitations de bande passante et de ressources empêchent les systèmes de téléconférences centralisés de passer à l'échelle.

Les chercheurs ont donc développé de nouveaux systèmes de téléconférences basés sur des architectures distribuées. Ces systèmes, tels que ceux pair-à-pair (*peer-to-peer, P2P*) et ceux basés sur la diffusion multipoint au niveau de la couche application (*Application Layer Multicast, ALM)*, ont surmonté le problème du passage à l'échelle, car le système ne repose plus sur un seul serveur central. Dans les systèmes de conférences distribués avec des utilisateurs hétérogènes, l'adaptation du flux est effectuée sur le périphérique de l'utilisateur final ou sur un nœud intermédiaire du réseau. Cependant, les systèmes de conférences distribués souffrent du manque de contrôle, de la charge du réseau, des retards et des pertes de paquets dus à la complexité des algorithmes distribués. Les chercheurs ont tenté de réduire la charge du réseau en utilisant la technique de la diffusion multipoint sur IP (*IP multicast*). La diffusion multipoint permet une meilleure distribution du contenu que la mono-diffusion IP (*unicast*), car elle évite la duplication de flux. Cependant, la diffusion multipoint sur IP n'est pas prise en charge par la plupart

des fournisseurs d'applications conférence en raison du manque de contrôle du système distribué et de son problème de déploiement sur le réseau. En outre, la construction d'un arbre de diffusion est difficile et les algorithmes sont complexes.

Ces dernières années, l'introduction des réseaux à définition logicielle (*Software-Defined Networks, SDN*) a ouvert de nouvelles possibilités pour une meilleure gestion des réseaux et des communicztions. SDN permet la séparation entre le plan de contrôle et le plan de transfert des données. Le contrôleur SDN est un serveur ayant une vue abstraite et globale du réseau, il est responsable du plan de contrôle. Il peut collecter des informations sur l'infrastructure complète du réseau et décider de la manière de gérer le trafic afin d'optimiser l'utilisation des ressources. Le plan de transfert, quant a lui, envoie le trafic en se basant sur les instructions du contrôleur SDN. Le système de communication peut tirer parti de la vue globale du contrôleur SDN sur l'état du réseau, afin de fournir un meilleur contrôle et une meilleure gestion des communications. Les observations ci-dessus nous ont incités à tirer profit de l'architecture SDN pour concevoir de nouveaux algorithmes de distribution et d'adaptation de flux pour les systèmes de téléconférences. L'objet de cette thèse est de définir un modèle de système de conférence utilisant SDN et de créer des algorithmes adaptés pour optimiser la qualité de service par rapport à la consommation de ressources. Nos contributions sont les suivantes :

Arbres de diffusion multipoint pour sessions MVoIP avec flux hétérogènes

Dans une audio-conférence entre plusieurs participants, les utilisateurs peuvent être affectés par les limitations du canal o du périphérique nécessitant une adaptation du flux à une qualité inférieure. Dans les architectures de téléconférences centralisées et distribuées mentionnées précédemment, cette adaptation est effectuée soit sur la machine de l'utilisateur final, soit sur un serveur central (par exemple le MCU), en utilisant un transcodeur ou la technique de la superposition des couches vidéo comme, par exemple, le codage vidéo évolutif (*Scalable Video Coding, SVC*). Si l'adaptation est effectuée sur le périphérique de l'utilisateur final, le flux d'origine passera par tout le réseau sans être modifié, jusqu'à ce qu'il atteigne l'utilisateur final. Cette méthode est simple, cependant, elle entraîne un gaspillage de la bande passante du réseau. Si l'adaptation est effectuée sur un serveur central (par exemple le MCU), l'adaptation provoquera une surcharge de calcul sur ce serveur et augmentera la latence.

Pour surmonter le problème de l'adaptation, nous avons défini un modèle de système d'audio-conférence exploitant le paradigme SDN afin de fournir un débit adaptatif à chaque participant. L'utilisation combinée de la diffusion multipoint et du placement d'adaptation au cœur du réseau permet

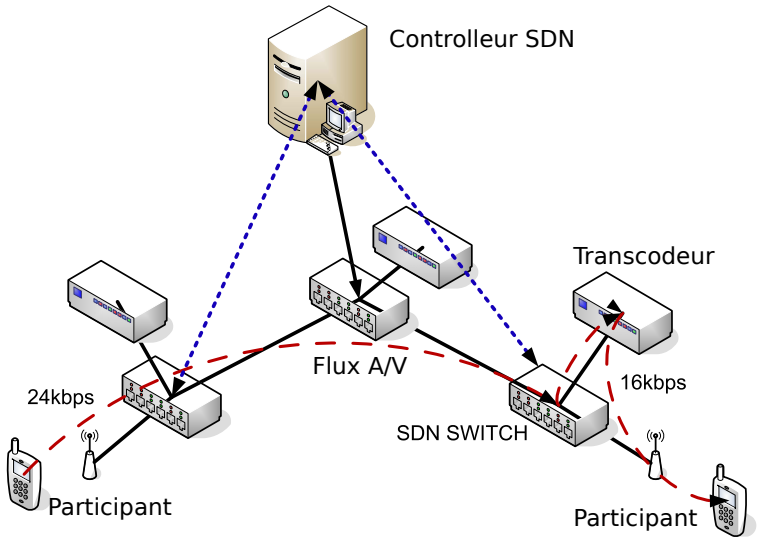


FIGURE 1: Exemple de l'architecture de notre modèle de conférences basé sur SDN

de réaliser d'importantes économies de bande passante. Nous avons conçu deux algorithmes pour créer un arbre de diffusion multipoint : l'arbre minimisant le poids (*Minimizing Spanning Tree, MST*) et l'arbre des plus courts chemins (*Shortest Path Tree, SPT*). L'algorithme MST cosiste à créer un arbre en connectant chaque nouveau nœud au nœud le plus proche existant déjà dans l'arbre. L'algorithme SPT, quant à lui, crée des arborescences en connectant directement le nouveau nœud avec le plus court chemin menat à la racine de l'arbre.

Nos résultats ont montré que l'algorithme MST présente la consommation de bande passante la plus faible par rapport aux algorithmes de téléconférences existants (ALM, MCU). Cependant, la latence moyenne est plus élevée que celle de SPT mais avec des valeurs acceptables. Les résultats montrent également que le nombre de transcodages de flux est réduit de moitié lorsque l'algorithme MST est utilisé par rapport à l'algorithme SPT.

Optimisation de la bande passante des appels en vidéo-conférences

Le modèle discuté dans la première contribution repose sur des hypothèses telles que la capacité infinie du réseau, ainsi que la stabilité des capacités des liaisons d'accès pendant la session. Nous avons adapté notre modèle à des réseaux plus réalistes avec des capacités en bande passante limitées. De même, nous avons étudié l'impact de l'architecture SDN sur les systèmes de vidéo-conférences utilisant des techniques d'adaptation de flux SVC. Les algorithmes présentés précédemment ont été utilisés pour réduire la bande

passante consommée. L'utilisation de SVC et de la diffusion multipoint permettent d'éliminer des couches vidéo à des endroits spécifiques dans le réseau. Les résultats montrent qu'avec la même infrastructure réseau, nous sommes en mesure d'établir simultanément un nombre plus élevé de conférences qu'en utilisant les approches de l'état de l'art telles que MCU et ALM.

Adaptation dynamique des apples vidéo-conférences

De nos jours, la plupart des participants à une audio/vidéo-conférence utilisent des appareils sans fil. Les liens d'accès sont souvent soumis à des interférences qui réduisent la capacité des utilisateurs à envoyer et à recevoir les flux qui correspondent à leur débit. Recalculer l'arbre de diffusion après chaque variation serait coûteux et pourrait entraîner des interruptions. Pour éviter cela, nous avons conçu des algorithmes qui replacent de manière optimale les règles de transcodage de flux sans recalculer les arbres de diffusion multipoint. Ces algorithmes ont une complexité très faible, ce qui augmente la réactivité du système et réduit les risques de coupures durant l'appel. Nous avons comparé le comportement de notre système à ceux basés sur MCU et ALM. Les résultats montrent que nos algorithmes sont plus rapides, plus réactifs, et permettent une meilleure adaptabilité du système par rapport aux approches classiques.

Contents

List of Figures

List of Tables

List of Abbreviations

AT&T	**A**merican **T**elephone and **T**elegraph
CS	**C**ircuit **S**witched
PS	**P**acket **S**witched network
PSTN	**P**ublic **S**witched **T**elephone **N**etwork
POTS	**P**lain **O**ld **T**elephone **S**ystem
VoIP	**V**oice **o**ver **I**nternet **P**rotocol
MOS	**M**ean **O**pinion **S**core
CAC	**C**all **A**dmission **C**ontrol
SDN	**S**oftware **D**efined **N**etworking
ISDN	**I**ntegrated **S**ervices **D**igital **N**etwork
MCU	**M**ulti-point **C**ontrol **U**nits
PBX	**P**rivate **B**ranch **E**xchange
MC	**M**ulti-point **C**ontroller
MP	**M**ulti-point **P**rocessors
MVoIP	**M**ultiparty **V**oIP
PESQ	**P**erceptual **E**valuation of **S**peech **Q**uality
RTP	**R**eal-time **T**ransport **P**rotocol
QoS	**Q**uality **o**f **S**ervice
QoE	**Q**uality **o**f **E**xperience
IETF	**I**nternet **E**ngineering **T**ask **F**orce
OPENSIG	**O**pen **S**ignaling **G**roup
GSMPv3	**G**eneral **S**witch **M**anagement **P**rotocol **v**ersion **3**
DCAN	**D**evolved **C**ontrol of **A**TM **N**etworks
PBNM	**P**olicy-**B**ased **N**etwork **M**anagement
API	**A**pplication **P**rogramming **I**nterface
PSNR	**P**eak **S**ignal-to-**N**oise **R**atio
CSP	**C**onstraint **S**atisfaction **P**roblem
ISP	**I**nternet **S**ervice **P**roviders
NSAL	**N**etwork **S**ervices **A**bstraction **L**ayer
SDM	**S**oftware **D**efined **M**ulticast
SVC	**S**calable **V**ideo **C**oding
ALM	**A**pplication **L**ayer **M**ulticasting
P2P	**P**eer-**to**-**P**eer
CDN	**C**ontent **D**elivery **N**etwork
TCP	**T**ransmission **C**ontrol **P**rotocol
UDP	**U**ser **D**atagram **P**rotocol
ASM	**A**ny **S**ource **M**ulticast
IGMP	**I**nternet **G**roup **M**anagement **P**rotocol
PIM	**P**rotocol **I**ndependent **M**ulticast

SM	**S**parse **M**ode
DM	**D**ense **M**ode
SSM	**S**ource **S**pecific **M**ulticast
Bidir-PIM	**B**idirectional **PIM**
MOSPF	**M**ulticast **O**pen **S**hortest **P**ath **F**irst
RP	**R**endez-vous **P**oint
MLD	**M**ulticast **L**istener **D**iscovery **p**rotocol
SCTP	**S**tream **C**ontrol **T**ransmission **P**rotocol
DCCP	**D**atagram **C**ongestion **C**ontrol **P**rotocol
SIP	**S**ession **I**nitiation **P**rotocol
MMUSIC WG	**M**ultiparty **M**ultimedia **S**ession **C**ontrol **W**orking **G**roup
HTTP	**H**ypertext **T**ransfer **P**rotocol
SMTP	**S**imple **M**ail **T**ransfer **P**rotocol
SCTP	**S**tream **C**ontrol **T**ransmission **P**rotocol
UAC	**U**ser **A**gent **C**lient
UAS	**U**ser **A**gent **S**erver
SDP	**S**ession **D**escription **P**rotocol
ITU	**I**nternational **T**elecommunication **U**nion
RAS	**R**egistration **A**dmission and **S**tatus protocol
RTCP	**R**eal-**T**ime **C**ontrol **P**rotocol
RTSP	**R**eal-**T**ime **S**treaming **P**rotocol
MGCP	**M**edia **G**ateway **C**ontrol **P**rotocol
MeGaCo/H.248	**G**ateway **C**ontrol **P**rotocol
MGC	**M**edia **G**ateway **C**ontroller
ICE	**I**nteractive **C**onnectivity **E**stablishment
ALM	**A**pplication **L**ayer **M**ulticast
WebRTC	**Web** **R**eal-**T**ime **C**ommunication
TURN	**T**raversal **U**sing **R**elay **N**AT
NAT	**N**etwork **A**ddress **T**ranslation
STUN	**S**ession **T**raversal **U**tilities for **N**AT
TLS	**T**ransport **L**ayer **S**ecurity
DTLS	**D**atagram **T**ransport **L**ayer **S**ecurity
MPEG	**M**oving **P**icture **E**xperts **G**roup
iSAC	**i**nternet **S**peech **A**udio **C**odec
AVC	**A**dvanced **V**ideo **C**oding
BL	**B**ase **L**ayer
HEVC	**H**igh **E**fficiency **V**ideo **C**oding
AOMedia	**A**lliance for **O**pen **M**edia
AV1	**A**OMedia **V**ideo **1**
NETVC	Inter**NET** **V**ideo **C**odec
MST	**M**inimizing **S**panning **T**ree
SPT	**S**hortest **P**ath **T**ree
ER	**E**rdős-**R**ényi
MP	**M**agoni-**P**ansiot
WM	**W**ax**m**an

Introduction

Advances in high-speed access networks have pushed the development of many multimedia communication systems including voice, pictures, audio and video. Among the most advanced communication applications, conference calls enable live communication between two or more participants in different locations, each possibly equipped with different devices. The development of free applications and low cost devices has popularized the use of conferencing systems among both individuals and businesses. However, the widespread use of mobile devices implies network constraints, such as low bandwidth availability, high delay, etc. These constraints make it harder to maintain a high quality media delivery and thus a high Quality of Experience (QoE) for users.

Two general methods exist for connecting the participants in a conferencing system. In the first one (typically in standalone conferencing systems), a central control device is used to connect multiple heterogeneous participants in a conference. This centralized control device is usually called a Multipoint Control Unit (MCU). It acts as a bridge for the participants and adapts the media streams to their channel limitations. Inside the MCU, multipoint media controllers and processors are used to re-encode the audio/video stream of a participant, in order to match the other participants' requirements. The MCU approach is easy to manage, however, it requires a significant processing power and may suffer from large delays, a single point of failure and network bottlenecks. In the second method, the participants directly connect to each others in a distributed way without the need of a centralized entity. The most well-know distributed systems used for conferencing are Peer-to-Peer (P2P) systems and Application Layer Multicast (ALM) systems. In P2P systems, all nodes are connected to each other, whereas in order to connect the participants in ALM systems, a non-optimal tree is built on the application level. P2P and ALM systems are more scalable than MCU-based systems since they do not rely on a central server and do not suffer from a single point of failure. However, stream adaptation and management is much more complicated in these systems, due to the use of complex distributed algorithms, especially in the case of heterogeneous participants.

For media transmission, conferencing systems may use IP unicast or multicast delivery. Using unicast in conferencing systems consumes an important amount of bandwidth resources when the number of participants is more than two. This is due to the fact that unicast replicate the same media stream in order to send it to each participant. In the contrary, IP multicast

creates a tree to deliver the same stream to all participants which ensures low delay and high bandwidth savings. Some P2P and MCU systems are able to leverage IP multicast. However, IP multicast is still not widely supported by Internet providers and operators as it faces many control, management, and security problems. In addition, the construction of multicast trees is difficult and the algorithms are complex.

Another component that can ensure high media quality and bandwidth savings is the type of content encoder used. An encoder is used to compress and adapt the stream to the limitations of the channel (i.e., its bandwidth capacity). Scalable Video Coding (SVC) is a layer-based video compression technique that allows a video stream bitrate to be reduced by removing some of its layers. SVC avoids any re-encoding, thus, a stream can be easily degraded to reach bandwidth expectations without the need to being decoded and re-encoded at a lower bitrate. SVC is very useful for video conferencing since it allows a better control and adaptation over the stream bitrate. However, the layer selection is only done at the endpoints (users) or in the MCU, and it remains unchanged inside the core of the network. The limitation imposed on the location of the SVC adaptation may lead to a waste of bandwidth.

In recent years, Software-Defined Networking (SDN) technology was presented as a solution for better network control and management. SDN separates the control plane from the forwarding plane. A centralized controller takes care of managing and controlling the network. A separate protocol, such as OpenFlow, allows communication between the controller and the network infrastructure (i.e., switches). In a conferencing system, SDN can be very useful by leveraging the controller's global view of the network state. Indeed, the controller can re-route the media traffic through a different path in order to avoid congestion, or reduce the stream quality in order to satisfy different receivers' capacities by using SVC inside the network. SDN also makes IP multicast more manageable and easier to deploy in such networks.

The aforementioned observations have prompted us to take advantage of SVC and SDN for designing a new conferencing system model. The purpose of this thesis is to define a new conferencing model and system based on SDN and to create algorithms for flow adaptation placement in order to optimize the QoS and the overall resource usage.

Thesis Contributions

Our contributions are as follows:

1. In order to adapt the streams inside the network, we define an algorithm leveraging SDN for building multicast media distribution trees and reducing the media stream bitrate at certain network locations, leading to an optimized network usage. Our algorithm is able to define where inside the network, and to what bitrate, should we be adapting

the streams for meeting the bandwidth capacities of the receiving participants. We propose two algorithms for computing the trees. The first algorithm, called *Shortest Path Tree* (SPT), minimizes the end-to-end delay; while the second one, called *Minimizing Spanning Tree* (MST), tries to minimize the bandwidth consumption inside the core network. Both algorithms optimize the placement of the streams' adaptation rules in the multicast trees. We compared our solution to MCU- and ALM-based approaches in order to evaluate the bandwidth savings and the latency. We see that our solution offers significant bandwidth savings compared to the two others.

2. Similarly, we have evaluated the usage of SDN on video conferencing systems utilizing SVC adaptation. The model discussed in the first contribution is based on assumptions such as: an infinite capacity of the network, the stability of the access link bandwidth during the call session. Therefore, we have adapted our model to more realistic networks with limited bandwidth capabilities. We use the algorithms discussed before to reduce the bandwidth consumed by video conferences. We took advantage of the SVC layering feature and multicasting to allow video layer dropping at specific locations. With the same network infrastructure, the results show that we are able to fit, simultaneously, a higher number of conference calls based on our model than with the MCU-based conference calls.

3. Since most of audio/video conference participants use wireless devices, access bandwidth variations may occur quite often during a call. In order to avoid a complete re-computation of the multicast trees, which can be costly in processing time, we propose algorithms that optimally relocate the streams' adaptation for each occuring bandwidth variation. The algorithms are based on tree traversal ideas and are very fast, thus offering high reactivity. The results show that our algorithms are faster, more responsive, and allow a better adaptability of the system compared to to MCU- and ALM-based approaches.

Road map of the Thesis

This thesis is organized as follows. Chapter 1 provides an overview of the state of the art on conferencing systems and communication systems' architectures and protocols. Chapter 2 introduces our SDN-based multi-party audio conference model as well as the related algorithms for setting up a call session. Chapter 3 presents our SDN-based video conference model using SVC. In Chapter 4, we propose algorithms supporting a dynamic adaptation of our proposed system to access links' bandwidth variations. We conclude with a summary of our work, and we provide some insights for future work.

The complete list of papers published during this thesis is available in Appendix B.

Chapter 1

State of the art

1.1 Background on communication systems

In 1876, devices enabling communications between two users in different locations started to be advertised for private use. These communications were limited between a pair of users until the invention of phone conferencing in 1945. Conferencing was first dedicated to transmitting audio from a phone to a loudspeaker. In 1956, Bell Labs developed telephone conferencing, followed by American Telephone and Telegraph (AT&T) in the early 1960s. Later on, the device developed by Bell Labs, called Picturephone, was a failure due to its high cost and lack of efficiency (Noll, 1992).

Historically, most telephone connections in the world have been made through Circuit Switched (CS) networks by using the Public Switched Telephone Network (PSTN) also called Plain Old Telephone System (POTS) (Coddington *et al.*, 1995) introduced in 1970's. CS reserves resources before the call. Thus, it guarantees the bandwidth sufficiency since the bandwidth is not shared. However, resource reservation makes CS networks not scalable. To overcome this problem, new types of network, Packet Switched networks (PS), were created. Unlike CS, a PS network requires neither connection establishment nor channel reservation. In PS, data are transferred directly from a participant to another and any new participant is able to use the channel resources since the bandwidth is shared.

Since PS enables information distribution in form of packets instead of a single bit stream, it opened the way for moving traditional communication networks into all-IP networks. Voice Over Internet Protocol (VoIP) is the IP-based version of telephone lines which uses PS telephony, where voice, data or video travels to the destination in network packets over the Internet (Varshney *et al.*, 2002). VoIP offers more flexibility and cost efficiency since the application does not rely on the network infrastructure.

Since technology has allowed both audio and video to be transmitted at the same time, people started to realize the benefits of conferencing systems. Besides the cost and time benefits associated with conferences call, business

companies started replacing meeting and regular calls by audio/video conferences to be able to put faces on names and organize more flexible meetings. From a business point of view, media conferencing reduces 30% of travel cost, creates better communication in order to organize a meeting or finding an employee. It as well increases selling by an average of 80%. 87% of organizations and consumers are using conferencing systems. Conferencing systems' users participate in at least one media conference call a week.

Due to all these benefits, the usage of media conferencing applications has considerably grown, and providing the best quality of video has become a priority for researchers. One of the main criteria when judging the quality of a conference call is the quality of its offered media.

Many components may affect the conference call quality such as its system architecture, the types of equipment used and the protocols and codecs applied to it. This chapter points to the conferencing systems' requirements and discusses the most important developed architectures for communication systems, as well as, their protocols and technologies.

1.1.1 Conferencing systems modes

Many classifications of conferencing systems have been created based on their goal, number of participants, architecture, etc. In our work, we chose to classify conferencing systems based on the number of participants and their roles during the conference session. We note that all conferencing modes with a number of participants higher than three are called Multi-party conferences.

- **Conference calls one-on-one**, which are also called personal conference calls, are call sessions made between only two participants. During this conference mode, both participants are considered to be fully involved in the conference and can both hear and/or see each other, talk to each other and share data like text messages and file or others.
- **Symmetric conferences or audio/video conferencing** are similar to the personal conference calls but with more than two participants. This mode provides the same rights for all its participants. It represents a digital version of a real meeting where all the participants can collaborate with each other without any restriction rules.
- **Voice-activated Switching conferences** are firmer than the previous modes. In this mode, one or more users are defined as speaker. The conference server switches between speakers to enable a specific one to talk. All the other users are considered to be participants who can listen or share data but not talk. This mode can be used in important meetings were the responsible needs to give information to other and does not allow interruption.

- **Role-based Meeting** specifies one participant as a host responsible for managing the conference. All the other participants are listeners unless they ask for the host permission to be a speaker.
- **E-Learning** used for teaching purposes. The instructor in this mode is considered to be a speaker and a listener to all the students. The other participants (students) are speakers but they can listen only to the instructor.
- **Streaming** can be considered as a conference mode where one participant, considered to be a deaf speaker, only sends broadcast media to all the listeners. The listeners are themselves mutes and can only communicate via text messages.

1.1.2 Quality of Service (QoS) requirements for conferencing systems

Circuit switching such as PSTN succeeded in providing a very high quality of audio transmission. This success is due to the fact that in CS a channel is reserved for the communication between the users. However, CS does not share the channel during the call which causes an unnecessary waste of resources. Packet switching systems came to overcome the problem of wasted resources by sharing the bandwidth and transmit information by packets. They hold many advantages comparing to PSTN (mobility, flexibility and cost efficiency), however, they should provide the same quality as PSTN does. Video conferencing systems should provide a high audio quality, as well as, a good video resolution. Video streams are usually very sensitive to the network state and require a certain amount of resource requirements, such as end-to-end delay, packet loss, cost, throughput, etc. (Chen and Nahrstedt, 1998). Video streams management in the network is more complicated than the audio streams due to their larger size. In addition, they are also affected by the users' heterogeneous wireless network which implies more constraints, such as channel capacity, bandwidth availability and latency. On the other hand, Unlike audio streams, video streams are less sensitive to packet loss since we can find many acceptable video quality of the same video. In order to provide an audio quality similar to ear-to-ear voice transmission, as well as a good video resolution, audio/video conferencing systems should meet certain QoS requirements such as ensuring a minimum packet loss, a minimum end-to-end delay and it should well manage the resources (e.g., the bandwidth).

1.1.2.1 Packet loss

The works in (Uhl, 2004) concluded that less than 1% of packet loss is acceptable for audio streams and less than 0.5% video streams during a conference call. More than this percentage will result in degradation of the quality. Packet loss is mostly due to a congestion in the network or to wireless bandwidth variation. In these cases, the packet may not arrive or it may arrive with an anomaly that prevents it from being decoded. There are two different ways to test the amount of packet loss; a subjective and an objective test. The subjective test consists of getting the opinion of the users directly after experiencing it. The parameter used in the subjective test is called Mean Opinion Score (MOS) (Fiedler *et al.*, 2010). This type of test does not rely on network parameter as the objective way does. In objective tests, the results rely on network parameters collected in the network. Some transport protocols, such as Transmission Control Protocol (TCP), ensure lossless data delivery. However, video conferencing tends to use the User Datagram Protocol (UDP) which is considered to be an unreliable transport protocol. UDP ensures minimum latency, however, it does not consider packet loss.

1.1.2.2 Latency or end-to-end delay

A delay in media transmission causes degradation in media quality. Many types of delays can occur during a conference call such as network delay, encoding/decoding delay, jitter delay, etc. An audio/video conference should manage the budget of delays in the network in order to make it acceptable for high voice and video quality. According to lab testing done by Cisco, the quality remains acceptable, with a one-way delay up to 200 ms (Cacheda *et al.*, 2007). The first important type of delays is the network delay. The network delay is considered to be a basic problem existing in all types of telecommunication networks. It is the sum of the transmission delay (the time to transmit a packet onto the link), the propagation delay (time for a packet to reach its destination) and queuing delay (the time the packet spends in routing queues). Another important delay type is the encoding/decoding delay. The encoding/decoding delay is the time used by the codec to encode/reconstruct the transmitted signal. This delay varies with the variation of codec types. A jitter delay is due to a long queuing and putting aside packets caused by a congestion in the network. With a large jitter delay, many packets are put aside. In this case, the codec may not find samples to play out. This sample non-presence leads to audible and visual gaps. During an audio/video conference, the jitter delay allowed is less than 20 ms for audio and less than 10 ms for video (Schuster *et al.*, 2002). This limit may increase by using a more suitable codec. To reduce the impact of jitter delay, a jitter buffer has been created to buffer the packets put aside and play them later in a stable way. However, the size of the jitter buffer affects the delay budget in the network. A Large jitter buffer decreases packet loss on one hand, however, it limits the delay budget of the network. A low jitter delay may allow audible gaps and reduces the quality. Thus, to overcome the Jitter

buffer issues a trade-off should be made according to the system parameters. In addition, an adaptive jitter buffer can be used to dynamically adapt to the delay variation in the network.

1.1.2.3 Network load and bandwidth

Since the network congestion leads to packets loss and delays, it has been proven that network resources play a big role in the degradation of audio/video quality. Conference calls usually share the bandwidth between them. Since the capacity of the link is limited it cannot support many calls at the same time. When the links are loaded the calls start losing quality due to packet loss and delay. To resolve network congestion, A Call Admission Control (CAC) (Perros and Elsayed, 1996) has been set to fit a maximum number of calls in the network. When the link capacity reaches the limits (there is no more bandwidth for a new call), the CAC starts rejecting calls. This causes a problem of scalability of the number of calls in the network. As another solution, the operator started setting priority to each call. The calls are rejected when a higher priority call is connected to the network while the network is saturated. To handle more calls in the network, the researchers headed towards creating routing protocols to distribute the calls among the resources. QoS routing protocols aimed at finding paths that satisfy multiple constraints (delay, weight,...) by computing paths that satisfy all the constraints related to the delay and packet loss. Despite the fact that these routing protocols increased the supported number of call in the networks, it also increased the complexity.

1.1.2.4 Bandwidth variation

Access links face bandwidth variations caused by traffic congestion, delays and buffering, as well as, external heavy activities on the network (video streaming, online gaming, etc.). Interferences suffered by wireless connections (bad weather, obstacles, etc.) also affect the bandwidth and reduce access links capability for receiving high quality streams. Among all real-time applications, audio/video conferences are the most disturbed by bandwidth variations since they rely on providing good quality of streams for the users.

To face bandwidth variation, congestion control algorithm and scalable video codecs are used to adapt the quality of the stream in order to match access bandwidth availability and users' expectations.

There is an extensive literature on conferencing systems under bandwidth variations. For example, Wu *et al.* (2001) provides an architecture and several mechanisms to perform video streaming that adapts to congestion and bandwidth variations. It investigates both unicast and multicast methods. But the congestion control is performed at the application layer, and thus at the endpoints of the communication. The authors De Cicco *et al.* (2008) investigate the effect of congestion and bandwidth variation on

a Skype video call. However, the case study includes only two hosts and, again, the adaptation mechanism is performed within the application layer. Some other works focus on the reactivity to bandwidth variation. For example, Vutukuru *et al.* (2009) proposes a cross-layer approach where the bitrate is directly estimated on the physical layer, allowing a quick adaptation to the new bandwidth. More recently, the new approach of Pseudo analog video transmission, which allows a direct adaptation of the video stream quality according to the channel noise, is investigated in Ding *et al.* (2016) and He *et al.* (2017).

The drawback of these cited works on conference call under bandwidth variation is that the video quality adaptation is at the endpoints. In the case of a multicast approach, this has three consequences:

1. If the video quality adaptation is performed at the sender, it will adapt to the video quality of the receiver with the lowest bandwidth. This means that the receivers with higher bandwidth cannot benefit from a quality corresponding to their access channel.
2. If the adaptation is performed at the receivers, the sender has to emit the highest quality stream through the multicast tree, which implies higher bandwidth consumption in the core network.
3. If the adaptation occurs on a central server, the load on this server will be high since it needs to receive information regarding the participants' capacities and adjust the streams.

1.2 System architectures

The architecture models play a major role in judging the conference quality. In addition, the architecture capability of adapting and adjusting to the participants and network characteristics and variation adds to its efficiency and enhance its quality. Conferencing systems architectures can be categorized based on the topology and media connections. They can be centralized or distributed. Centralized conferences use a central server and are capable of managing the system and the participants in a simpler and clearer way. However, distributed conferences are more scalable and can handle large-scale media (Handley *et al.*, 1997).

1.2.1 Centralized communication systems

Centralized conferencing systems are considered to be simple since they are base on a central server usually responsible for managing the network. The central server, in a centralized conference, is responsible for mixing, transcoding and distributing media streams to all receivers after receiving it from a sender. However, to enable transcoding at the server, the streams

need to be decoded, then, re-encoded to be redistributed. The process of decoding/encoding the streams cost the server a lot of resources. Thus, the number of conferences calls in a centralized model is limited and depends on the number of participants. Therefore, centralized conferencing systems suffer from scalability issues and need a lot of resources on the central server side. In addition, since all the communication has to pass through the central server, the latter can suffer from a traffic load, congestion and create a bottleneck in the system. The central server is also considered to be a single point-of-failure in the centralized conferencing systems.

The most known conference system based on a centralized model are PSTN, Integrated Services Digital Network (ISDN), Multipoint Control Unit (MCU), VoIP and most recently Software Defined Networking (SDN).

1.2.1.1 PSTN and Integrated Services Digital Network (ISDN)

PSTN is a CS network used for real-time telephony calls. It allows the transmission of analog audio data between phones operated by telephony operators (see Figure 1.1). To ensure a good call quality without any delay, PSTN reserves the channel between the participants in the call. To transfer data between participants, PSTN needs first to establish a connection and maintain the reserved channel throughout the call duration. Even if the reservation of the channel guarantees a good QoS, it denies any other participant from accessing the resource capacity. The resource reservation made PSTN costly. In addition to the cost, PSTN had limited data transfer rates which is considered as an important drawback (Kuhn, 1997).

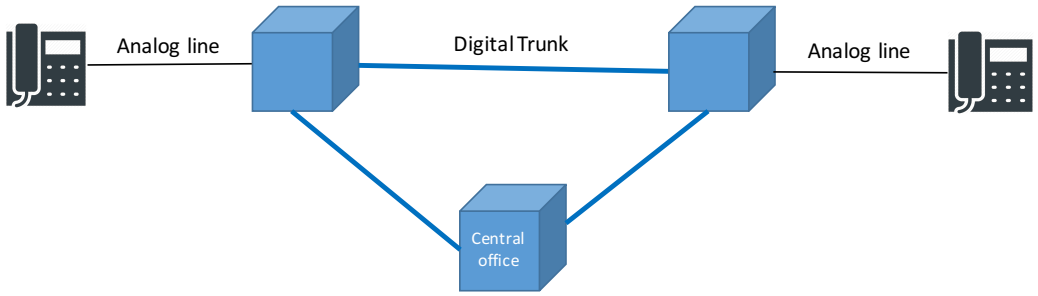


FIGURE 1.1: PSTN architecture

In the 1980s, the demand for high-speed telecommunications support within CS networks has been satisfied by designing ISDN (Stallings, 1989). ISDN has been developed for digital data and audio transmission over ordinary phones. Unlike early PSTN, ISDN uses multiple channels to eliminate the need for separate voice and data telephone lines which allows it to dedicate all its bandwidth to data transmission. Since ISDN is fully digital, lengthy process of analog modems is not required. Thus, it was able to surmount PSTN by providing faster data transfer rates. However, ISDN lines

are that it is very costly compared to PSTN and need special devices, which prevented it from being widely deployed.

With ISDN, digital communications served for developing new systems allowing faster and cheaper communication especially in the business world, such as Private Branch Exchange (PBX) used by business telephone systems to connect users in a local enterprise and allowing them to share few external phone lines, and conferencing systems (Arazi *et al.*, 2002).

1.2.1.2 Multi-point control unit (MCU)

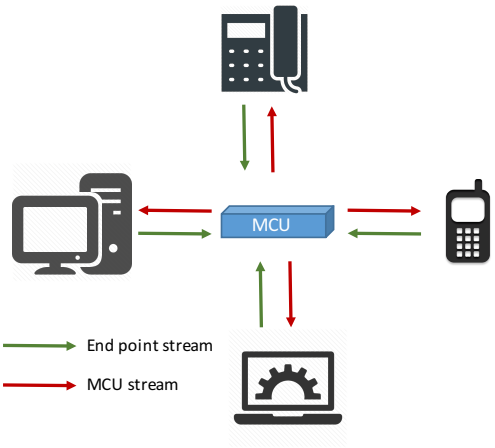


FIGURE 1.2: MCU architecture logic

To create conference calls with more than two participants, conference bridges were used (Shaffer and Fernandes, 1998). A conference bridge can refer to the software, hardware and/or the phone number used to connect participants to a large conference call. However, the hardware/software heterogeneity of the participants' devices and their access links brought a lot of challenges to the bridges. To address these challenges, it was necessary to create a special type of central bridge with more control functionalities. MCU is referred to as a type of central bridge or gateway in a conferencing system that is able to facilitate the call setup and control, redirect and transcode the media streams in order to meet each participant parameters (Willebeek-LeMair *et al.*, 1994). MCU is a software implemented in a central device connected to the network by large bandwidth links for higher Internet connectivity (Ramadass, 2010). It consists of a multi-point controller (MC), responsible for handling signaling and control exchanges between different end-users that operates with different protocols, and optional Multi-point Processors (MP). The communicating participant in a conference call sends media streams to the MCU in order to transmit it to other participants (see Figure 1.2). Before transmitting these streams, the MCU may do some transcoding (coding/decoding) and mixing. The stream transcoding process and the mixing are usually done in the MCU, but it can be done as well in

the participant's device. In the first case, the participant, with low characteristics, asks the MCU to reduce the quality (bitrate, resolution, frame rate) of the transmitted media stream in order to reduce the bandwidth. The MCU transcodes the media streams before sending them to the receivers. The transcoding process is quite expensive and needs high computation power (Rodríguez *et al.*, 2016). Since MCU is centralized, it is vulnerable to one-point-failure. In addition, the connection between the MCU and the network needs to be insured with high bandwidth capacity links which make it very costly. However, Xu *et al.* (2012) showed that the most famous video conferencing services in 2012, Google Hangout, Skype and iChat, uses MCU-based approach.

1.2.1.3 Voice over IP (VoIP)

VoIP also called IP telephony, Internet telephony, voice over broadband, broadband telephony, is the packeting and transport of classic PSTN over an IP network. It uses PS networks instead of CS networks. VoIP works by turning analog audio data into digital data that can be transmitted over the Internet. Unlike traditional phone system (e.g., PSTN), VoIP shares infrastructure of the web in order to decreases the resources cost. It provides as well more flexibility than PSTN by plugin the VoIP phone in any new location and inter-working with other IP protocols. There is no dedicated path required between endpoints.

VoIP architecture

Three main components exist in every VoIP system: a VoIP server, VoIP clients and Gateways (see Figure 1.3).

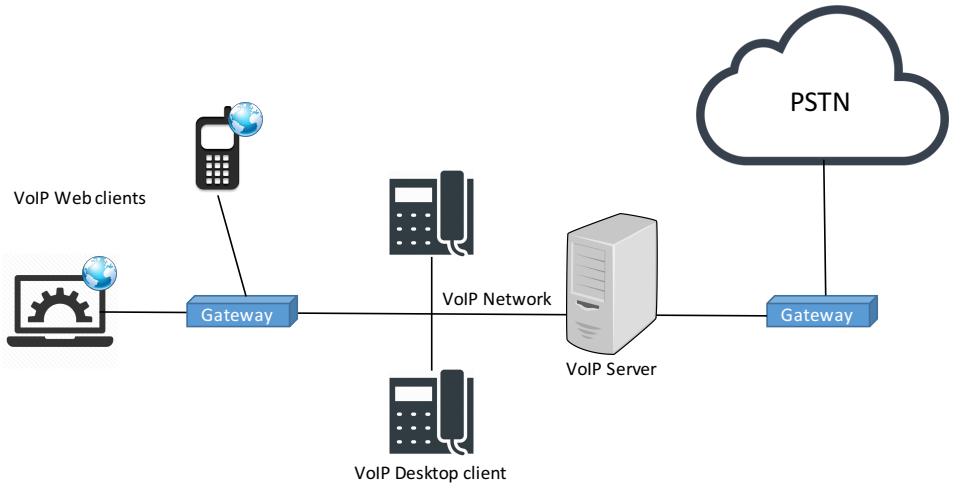


FIGURE 1.3: VoIP architecture

- **VoIP servers** are the main servers that route the call through VoIP phones or devices. These VoIP servers are connected to the Ethernet and to the VoIP clients through the network devices (switches, routers...). The

VoIP servers can also have analog ports connected to it to plug normal analog telephones. They can also be connected to a VoIP trunk from a provider to eliminate the need for telephone lines; The VoIP server can be hosted inside a building or on the Internet like every any other service allocated and paid monthly.

- The VoIP server is connected to devices, called **VoIP clients**, to provide VoIP services. The VoIP clients can be telephones, computers, smartphones, etc. There are two types of VoIP clients, HardPhones and SoftPhones. HardPhones are real normal telephones basically provides real-time audio communications services and configured to the web services, while softphones are software installed on the devices to provide real-time audio communications. Each VoIP client is connected with a given account to the VoIP server.
- The next component is the **VoIP gateway**. Gateways are what connect different types of communication networks; it turns back VoIP communication to normal telephone calls. For example, if a VoIP client wants to establish a call with a normal phone, the call will be routed to the VoIP server, then to the gateway which will connect to a normal telephone line. This can be very important for the enterprises that pay for the Internet connection to reduce the billing charges by using the normal line phones.

VoIP uses signaling protocols, discussed in Section 1.3.3, such as H.323, SIP, MGCP for call establishment and media transfer protocols, discussed in Section 1.3.4.1, such as RTP.

VoIP faced a lot of challenges with the quality of the call since the performance depends mainly on the Internet connection speed and the distance between the points of connection. Multiparty VoIP (MVoIP) conference calls or VoIP calls with more than two participants created more challenges for the standard VoIP. An early work by Prasad *et al.* (2003) tackled the issue of noise floor control in MVoIP calls. As mixing several streams increases the floor noise, they proposed a system to set the maximum number of participants to allow in a conference, in order to retain an acceptable audio quality. In 2006, Xu *et al.* (2006) proposed a peer-aware silence suppression for MVoIP conferences. They limit the number of concurrent speakers by performing silence suppression and speaker selection by a distributed system running in each client. The design of a complete MVoIP conferencing system was proposed by Sat *et al.* (2007). The system leverages user-observable metrics as well as network metrics to adapt its transmission topology, loss concealment schemes and play-out scheduling. Elleuch and Houle (2008) proposed a large-scale peer-based MVoIP system in 2008. Their solution enables multi-host media process support while the conference control and management are kept simplified and centralized around the administrator. They build two different meshed networks to enable both voice audio distribution between

participants and general conference control. Evaluating and comparing audio mixers for MVoIP has been done by Chandra *et al.* (2009). They proposed their own algorithm which outperforms the others in terms of quality and complexity. In 2009, Mani *et al.* (2009) studied MVoIP systems based on the Real-time Transport Protocol (RTP) bandwidth used, the quality degradation due to transcoding and the placement of voice-enhancement modules. They proposed an MVoIP system with adaptive sampling rate scaling up to ten clients. Hoeldtke and Raake (2011) defined in 2011 a new state model for MVoIP calls and applied statistical measures to recordings made during a three-party conferencing quality test, which was designed to evaluate the perceived quality under various speech transmission properties. In 2013, Adel *et al.* (2013) showed that the G.107 E-model is inaccurate in measuring the Quality of Experience (QoE) of MVoIP calls and thus they proposed an improved E-model specifically designed for MVoIP sessions, which produces results very similar to the Perceptual Evaluation of Speech Quality (PESQ) scores.

1.2.1.4 Software Defined Networking (SDN)

The most important network architectures were facing issues due to the complexity of network configuration that requires a lot of management and resources. These requirements were difficult to achieve due to the limitations of the traditional network infrastructure. SDN is considered as a novel and innovative paradigm that provides proper solutions for these issues by supporting intelligent applications and adding policies to the network. SDN can lower operating costs through simplified hardware, software, and management.

In the last years, the SDN has begun to be famous in the networking world due to the work of the Internet Engineering Task Force (IETF) Forwarding and Control Element Separation working group in 2000. OpenFlow (Heller, 2009) has brought the implementation of SDN closer to reality. The concept of SDN is to separate the intelligence of the network from the network device itself and to locate all the control in a plane different from the data plane. This concept took interest in solving many network limitations.

1.2.1.4.1 From policy-based network management (PBNM) to SDN

The advantage of SDN is based on the principle of control plane and data plane separation and the benefits of centralization in terms of control. This approach has not been invented in SDN for the first time but it has been discussed in other networking research groups such as in the Open Signaling

Group (OPENSIG) (Campbell *et al.*, 1999), and the General Switch Management Protocol (GSMPv3) (Doria *et al.*, 2002) where the first believed in the necessity of having a programmable network interface to facilitate access to network resources to allow the development of network environment, and the second used controllers to control a label switch. Likewise, Devolved Control of ATM Networks (DCAN) (Merwe and Leslie, 1997) and 4D Project (Greenberg *et al.*, 2005) proposed the decoupling of control plane and forward plane. In addition, (NETCONF) protocol (Caesar *et al.*, 2005) allowed network devices to expose an API through which extensible configuration data could be sent and retrieved. Boucadair and Jacquenet (2014) considers that SDN techniques as a whole are an instantiation of the PBNM (Bertó-Monleón *et al.*, 2011) framework which is created for managing QoS and security on distributed networks. PBNM includes policy-based network management, the use of delineated policies to control access to and priorities for the use of resources. Policies are operating rules that can be referred to as a way to maintain order, security, consistency, etc., and are represented by a cycle of (event/condition/action).

1.2.1.4.2 Architecture

SDN is implemented on network resources such as switches, routers, virtual machines, etc. The decoupling between the control plane and forwarding plane allows having a direct programming of the network control, a simple enforcement of policies, a global view of the network due to the centralized controller, and more agility in the forwarding plane which opens the doors in front of the innovation of new protocols and applications. SDN architecture is shown in Figure 1.4. This architecture (Brief, 2013) is composed

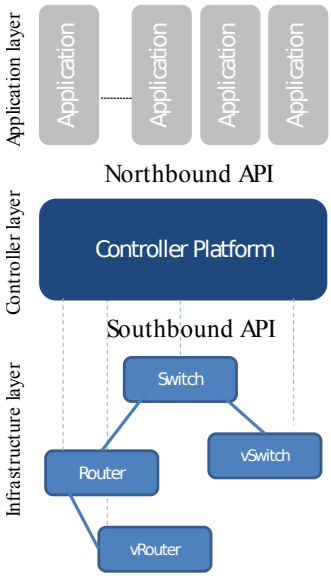


FIGURE 1.4: SDN architecture

of three layers:

1. An infrastructure layer that regroups all the network machine ensures the data forwarding between each other.
2. A control layer responsible for all control requirements of the network environment.
3. An application layer where all the business applications are located.

Furthermore, SDN architecture supports a set of Application programming interfaces (APIs) that makes it possible to implement common network services, including routing, multicast, security, access control, bandwidth management, traffic engineering, quality of service, processor and storage optimization, energy usage, and all forms of policy management, custom tailored to meet business objectives (Sezer *et al.*, 2013). There are two types of API:

1. "Southbound API" for networking devices to integrate into the ecosystem (e.g., OpenFlow). Its main function is to enable communication between the SDN controller and the network nodes (both physical and virtual switches and routers). OpenFlow protocol is the most known protocol used by SDN southbound API as a communication interface between the control and the forwarding layers.
2. "Northbound API" for applications to attach to the ecosystem (no standard), it describes the area of protocol-supported communication between the controller and applications or higher layer control programs.

The hardware devices located in the infrastructure layer of SDN can be either specific to work only with a controller to have the control information or they can support at the same time the traditional protocol which does not use the controller. To reduce the overhead of new requests on the centralized controller, there is a proposal of hybrid controller usage which uses a local and a global controller. Multiple controllers may be used to reduce latency and provide more performance. The placement of the controller and the quantity of new packet request may affect the latency of the network in order to some works such as Palette (Kanizo *et al.*, 2013) and One Big Switch (Kang *et al.*, 2013). A solution of these problems is presented by Ethane (Casado *et al.*, 2007) and DIFANE (Yu *et al.*, 2010) which demonstrate the large controller's capacity to manage a big number of new packet request and by regrouping packets with some similar characteristics in a flow. Due to the controller capacity discussed as a solution of the preceding problem, SDN supports the reactive controlling approach which can react on each new packet request without having the problem of latency, while the proactive controlling approach is always considered as a static approach reacting only with the information pre-implemented in the controller. The advantage of the proactive control approach is that any controller problem will not affect the network performance. A mixed reactive and proactive approach is proposed in DevoFlow (Curtis *et al.*, 2011), by letting the switch holds the responsibility of the normal packet while the controller looks after the flow with the high throughput.

As a summary, SDN can be presented in several architectural forms, however, all forms are cut down to the following main bases:

- Decoupling control plane from the data plane,
- Forwarding data follows a certain flow path and sustain same policies and rules decisions as all packets of the same flow.
- Controlling the network is global and logically centralized in the SDN Controller.
- Programming the network through different APIs.

Researchers rely on the programmability of SDN and its centralized global view to create less complex operation adjusting the infrastructure to new technologies requirements. SDN enables this adjustment by simply deploying new low level rules to the devices or creating new APIs. Even if SDN offers more scalability and simplicity to get over many traditional network issues, it creates different challenges regarding controller scalability and its resiliency to failures and requires special attention to northbound standardization (Yeganeh *et al.*, 2013).

Many applications can benefit from the emergence of SDN to improve the QoS. Since SDN enables dynamic network reconfiguration, applications such as video streaming and real-time video transfer can address the problem of bandwidth usage. On the other hand, voice applications can reduce packet loss and gaming application can benefit from the low latency.

1.2.1.4.3 SDN for Video Streaming

Nowadays, content providers such as Netflix and YouTube are responsible for most of the Internet video traffic. Thus, providing the best QoE has become the main challenge for the providers. Video quality degradation is mainly caused by bandwidth overload that increases playback buffering and startup delays for video streaming and exceeds latency bound for live/interactive video applications.

Researchers today are applying SDN to implement less sophisticated techniques for bandwidth estimation, better resources utilization and rate adaptation used by real-time and streaming video providers.

To enhance video delivery, Yu and Ke (2018), Civanlar *et al.* (2010) and Egilmez *et al.* (2012) proposed new routing methods over SDN that reduce packet loss rate and delay. Yu and Ke (2018) beats Dijkstra and Bellman-Ford algorithms with a genetic-algorithm-based for finding the optimum path. It reduces the packet loss rate while improving the throughput and Peak Signal-to-Noise Ratio (PSNR). Civanlar *et al.* (2010) proposed a routing algorithm that provides a tradeoff between the delay and packet loss while selecting the path (a non-shortest path). It makes sure of introducing more packet losses to a path that maintains a tolerable delay. Egilmez *et al.* (2012)

minimizes packet loss and latency by proposing a dynamic QoS routing algorithm that routes some flows based on their Constraint Satisfaction Problem (CSP).

Egilmez *et al.* (2013) and Cofano *et al.* (2016) used SDN to focus on providing the best quality bitrate to the end user. Egilmez *et al.* (2013) improves the quality of scalable video streaming by proposing dynamic rerouting of QoS streams with a minimum disturbance on best-effort traffic. Cofano *et al.* (2016) built a video control plane which enforces video quality fairness among concurrent video flows generated by heterogeneous end users. The bitrate adaptation assistance showed the best results in terms of video quality fairness among clients.

On another hand, some researchers work on improving the bandwidth utilization in an SDN environment (e.g., Kim *et al.* (2010) and Jan Willem *et al.* (2016)). Kim *et al.* (2010) proposed new APIs as an extension of OpenFlow. These APIs redirect some flows to rate-limiters for a better use of the aggregated bandwidth. Jan Willem *et al.* (2016) proposed a DASH-aware networking architecture based on SDN. This proposal enables the better configuration of the network and defines how bandwidth should be shared between video traffic and other traffic types, and among video players.

Several works focus on the problem of network resource management to achieve a better resource utilization in SDNs. Van Adrichem *et al.* (2014) monitors QoS metrics for each flow to determine adequate parameters. By examining flows at the source and destination switches, accurate results can be obtained while minimizing the network and switch CPU overhead. Jarschel *et al.* (2011) presents a model based on queuing theory to evaluate the impact of different processing strategies. Georgopoulos *et al.* (2013) takes into account, device and network requirements to optimize the QoE for all video streaming devices in a network. In the context of home networks, Kumar *et al.* (2013) proposed to leverage SDN to enable Internet Service Providers (ISPs) to expose some controls to the users to manage service quality for specific devices and applications in their household.

1.2.1.4.4 SDN for VoIP

Unlike Video streaming, works involving SDN applied to VoIP applications are very limited since the importance of video traffic is much bigger than the one for VoIP. Jivorasetkul *et al.* (2013) proposed an end-to-end header compression mechanism for reducing latency in SDN networks thus improving time-sensitive applications. Saldana *et al.* (2014) used SDN to identify flows of small packets (such as VoIP) for removing common header fields and multiplexing packets in the same frame in order to reduce the overhead. A Network Services Abstraction Layer (NSAL) and a unified data model were introduced by Sieber *et al.* (2015) for both SDN and legacy devices in order to achieve QoS for time-critical VoIP applications. QoS level guarantee and resource prioritizing enforced by SDN have been proposed by Karaman *et*

al. (2015) via limiting bandwidth, assigning flows to different queues and adapting routing decisions based on network conditions. They were able to reduce the the loss rate by 5% and latency and jitter by more than 50% in VoIP scenarios.

1.2.1.4.5 SDN for video conference

To improve the participants' QoE and simplify the network management of video conferences Zhao *et al.* (2014) and Yang *et al.* (2016) proposed that the SDN controller directly manages the multicast tree construction and the video layering inside the network. However, while some of them minimize the bandwidth consumption but degrade the video quality stream of the participants, the others provide high-quality video stream but at the price of higher bandwidth consumption. For example, the latter constructs a different multicast tree for each video layer. This is not optimal in terms of bandwidth saving.

Some works show the advantage of SDN regarding video delivery. For example, the authors Laga *et al.* (2014) and Yang *et al.* (2016) use SDN with SVC to reduce the number of video freezes and the bandwidth consumption, respectively. However, the first one only applies to one-to-one video delivery, while the second focuses on the participants' screen size capacity and does not take into account the network access link properties. Both works keep SVC functionnalities at the end users or in the MCU.

1.2.1.4.6 SDN for multicast

Another use case of SDN that is also involved in improving multicasting. This involvement is called Software Defined Multicast (SDM). Combining SDN features with multicasting enables better deployment of new routing algorithms. SDM can overcome the limitations of traditional IP multicast. Thanks to the centralized view, the recalculation of multicast routing tables has become more flexible and reusable.

Applications like live video streaming, video and audio conferencing, use multicasting to transmit data to multiple users simultaneously. The existing multicast forwarding algorithms are either not efficient or not scalable. SDM is able to provide more efficient algorithms using its centralized controller. Noghani and Sunay (2014) allowed the SDN controller to deploy IP multicast not only between source and destination but also via a northbound interface. The results showed an increase in the PSNR of the received video comparing to the one in a non-SDN network. Fan *et al.* (2016) used SDM to create a multicast solution for fat-tree data center networks to ease the connection/disconnection of a user in a multicast group. This solution increased the performance of delay/throughput comparing to the existing centralized multicast scheduling algorithm. Humernbrum *et al.* (2016) empowered users

in a multicast group by giving them membership control which does not exist in IP multicast. In addition, they have developed a new approach for the calculation of multicast trees. Their results show a decrease in the number of flow table entries.

1.2.1.4.7 SDN for video layering

Multicast methods, together with video layering techniques such as SVC, can provide mechanisms to manage video calls while ensuring different quality streams to the participants. For example, De Amorim *et al.* (1999) and Chandrasekar and Baskaran (2011a) use these mechanisms to improve the participants' QoE. However, the video layering adaptation in this work can only be performed at the application layer, and thus the network bandwidth consumption cannot benefit from it.

As mentioned before, the emergence of SDN improved the participants' QoE and simplify the network management. Tang *et al.* (2014) have introduced a video streaming multicast application in SDNs to adapt to network conditions. The proposed solution uses SVC to deliver coded video with as high fidelity as possible. Zhao *et al.* (2014) and Yang *et al.* (2016) gives the management of the video layering on multicast trees to the SDN controller. Yang *et al.* (2015) proposed SDM2Cast, an SDN-based, scalable multimedia multicast streaming scheme, to deliver each SVC video layer has its own multicast tree. The proposed scheme optimizes multimedia flow management and provides scalability and flexibility. Xue *et al.* (2015) focus on how to solve the SVC video manycast problem of efficiency coming from multiple sources to multiple destinations. They designed two heuristics to achieve the optimal solutions for small-scale problems. More recently, Yang *et al.* (2017) and Yang *et al.* (2018) use SVC and SDN to propose a method that provides admission control, in-network adaptation, and supports heterogeneous devices having different display capabilities. Since SVC is capable of adjusting the bitrate of a video stream and of reducing its load, it would be interesting to use it in the core of the network. Egilmez and Tekalp (2014) create a distributed architecture for SDN network control in order to comply with QoS constraints. However, the authors use SVC in unicast mode. Oliveira *et al.* (2018) considered delay, throughput and PSNR in order to prove that using SDN with SVC in a video conferencing systems delivers better video quality and reduce delay.

1.2.1.4.8 Summary

Many researchers took interest in SDN to implement new solutions for different topics such as VoIP, video conferencing systems, video streaming and IP-multicast communications and video layering. Table 1.1 depicts a comparison of each related work included in this section, taking into account SDN involvement in these mentioned topics. Some work treated each topic

alone, others leveraged SDN features in order to combine many topics together. We can see that there exists only a few SDN-based works involving conferencing systems comparing to other topics (e.g., video streaming). Furthermore, Zhao *et al.* (2014) and Yang *et al.* (2016) pointed out the importance of SDN involvement with video layering and IP-multicast for better conferencing performance. However, there are no other works in the literature which combines these three topics together.

TABLE 1.1: Comparison table of SDN related work

Work Done	Video streaming	VoIP	Video conferences	SDM	Video layering
(De Amorim *et al.*, 1999)					X
(Civanlar *et al.*, 2010)	X				
(Kim *et al.*, 2010)	X				
(Jarschel *et al.*, 2011)	X				
(Chandrasekar and Baskaran, 2011a)					X
(Egilmez *et al.*, 2012)	X				
(Egilmez *et al.*, 2013)	X				
(Georgopoulos *et al.*, 2013)	X				
(Kumar *et al.*, 2013)	X				
(Jivorasetkul *et al.*, 2013)		X			
(Van Adrichem *et al.*, 2014)	X				
(Saldana *et al.*, 2014)		X			
(Laga *et al.*, 2014)			X		X
(Egilmez and Tekalp, 2014)	X				X
(Zhao *et al.*, 2014)			X	X	X
(Tang *et al.*, 2014)	X			X	X
(Noghani and Sunay, 2014)	X			X	
(Sieber *et al.*, 2015)		X			
(Karaman *et al.*, 2015)		X			
(Yang *et al.*, 2015)					X
(Xue *et al.*, 2015)				X	X
(Cofano *et al.*, 2016)	X				
(Jan Willem *et al.*, 2016)	X				
(Yang *et al.*, 2016)			X	X	X
(Fan *et al.*, 2016)				X	
(Humernbrum *et al.*, 2016)				X	
(Yang *et al.*, 2017)	X				X
(Yang *et al.*, 2018)	X				X
(Yu and Ke, 2018)	X				
(Oliveira *et al.*, 2018)			X		X

1.2.2 Decentralized communication systems

The main aim of a decentralized system is to get rid of the fundamental problem of the centralized system i.e., having a single point of failure. In a decentralized system, there is no need for a central server. Participants can communicate directly with each other by sending streams via unicast or multicast. We note as decentralized architectures: Peer-to-Peer (P2P) systems and Application Layer Multicasting (ALM).

1.2.2.1 P2P communication systems

In a P2P system, nodes are considered as senders and receivers at the same time (Alessandria *et al.*, 2009). The nodes transmit media streams to each other through a spanning tree that connects the network nodes together. P2P systems do not have a point of failure since the connection between two nodes is arbitrary and varies at each connection. Thus, P2P systems are considered to be robust and scalable since they always ensure end-to-end communication, unlike centralized systems where all communication must pass through a given server.

Several VoIP providers have implemented their own VoIP systems based on P2P communication. Skype (Guha and Daswani, 2005) is one of the most known VoIP P2P system, created in 2003. Skype allows its users to exchange voice, video and text with each other through Internet infrastructure as well as PSTN network. Media exchange in Skype is distributed, however, login is centralized and requires passing through a Skype login server.

Other researchers have also tried to decentralize existing VoIP and video conference systems, in order to benefit from P2P features. Klauck and Kirsche (2009) replaced the central SIP servers for localization and invitation, in a video conference system, with a decentralized server-free P2P SIP-based system. Amad *et al.* (2009) also propose a new P2P SIP architecture for Internet telephony to optimize the global end-to-end delay. Since then, a lot of work tried to enhance P2P SIP systems; Yu (2012) worked on improving SIP query in P2P VoIP systems, Maenpaa (2013) worked on reducing the session setup delays. Other works WIFI P2P for VoIP Mobile Telephony, such as Kbar *et al.* (2010) and Mhatarmare and Raut (2013) who used P2P to allow users to find each other within WIFI range in order to establish a P2P connection and communication with no cost. Chaubey and Trivedi (2014) designed a completely decentralized system for Mobile conferencing based on the p2p architecture, using WIFI, without the need for any centralized services. Jabbar *et al.* (2013) proposed a middle-ware between mobile nodes and higher application layers for more efficient direct P2P communication among users in the mobile environment. Yu and Yu (2012) proposed a multicast architecture for video conferences, able to support a high number of video call sessions at the same time.

P2P gained popularity in video streaming and VoIP, however, for video conferencing it still faces bandwidth problems. Li *et al.* (2004) proposed a hybrid system solution that mixes centralized servers and P2P in order improve the bandwidth efficiency in P2P video conferencing systems. Ponec *et al.* (2009) used the same solution for multiparty conferencing applications. Another hybrid system was created by Munther *et al.* (2012). This system uses Hybrid Content Distribution Model to distribute parts of the video stream fairly among participants, taking into account heterogeneous networks. Liang *et al.* (2011) proposed a bandwidth sharing algorithm that optimizes system utility for multi-party P2P conferencing systems. Currently, P2P is applied between users, in WebRTC, to allow browser-to-browser communication, such in Nurminen *et al.* (2013), Apu *et al.* (2017) and Wang and Mei (2017), mainly in order to save bandwidth.

Most of the proposed P2P solutions offer cost-efficiency, easy deployment and enhance the quality for the end users. However, some problems and limitations face P2P audio/video communication systems in terms of security and control. Since P2P systems allow all nodes to communicate with each other, an interference of an untrusted node threaten the system security and makes it more vulnerable than client-server systems. Many researches discussed security solutions for many P2P communication systems components, such in Garfinkel (2005), Klauck and Kirsche (2009), Watzlaf *et al.* (2010), Jabbar *et al.* (2013), etc. On the other hand, node-to-node and network-to-network communication within a P2P system may cause a degradation in media quality depending on the node and the network capacity. Decreasing the quality form a higher capable node to a lower capable node can be beneficial in some case, however, its effect will reach all the successor nodes as they will all receive the lower media quality. In addition to quality degradation, node-to-node and network-to-network communication within a P2P increase the control complexity of the algorithms for finding the next node and may affect the real-time performance.

1.2.2.2 Application Layer Multicasting (ALM)

Applications such as video streaming and multiparty video conferencing are usually based on one-to-many communication and uses IP-multicast in order to distribute data to all destinations. IP-multicast, implemented usually in the network infrastructure, allows data duplication and transmission in the core of the network. However, since IP-multicast relies on the network infrastructure, Application Layer Multicasting (ALM) (Hosseini *et al.*, 2007a) was introduced in order to overcome the hurdle of router dependency and to create efficient data delivery without modifying the network. To do so, ALM uses multicasting as an application benefit rather than a network benefit (Banerjee *et al.*, 2002a). The multicasting in ALM is implemented by the peers (end-points) instead of the routers. In place of creating an optimal tree, as in IP-multicast, ALM consists of transmitting packets between peers by

building non-optimal trees. The transmission is done using unicast protocol with multicast functionalities (see Figure 1.5).

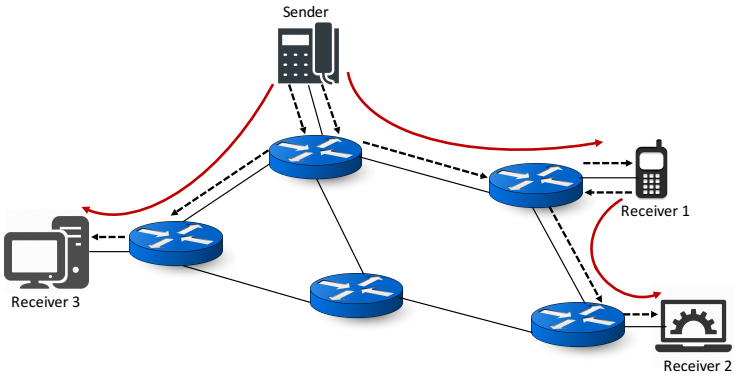


FIGURE 1.5: ALM architecture

There have been many research regarding the construction of different ALM protocols, such as ALMI (Pendarakis *et al.*, 2001), ZIGZAG (Tran *et al.*, 2004), NICE (Banerjee *et al.*, 2002b), and OMNI (Banerjee *et al.*, 2003). While NICE reduces the network resource usage, it does not put much effort in optimizing the end-to-end delay performance, which is very important for real-time systems. Unlike NICE, OMNI and ZIGZAG minimize the average delay, however, ZIGZAG faces bandwidth load on the nodes closer to the source which might not be acceptable. On the other hand, ALMI suffers from one-point-failure due to its centralized nature. These are just a few examples of the many ALM protocols. While the ALM solutions overcome the deployment and maintenance limitations of the IP multicast, it has practically zero information about the fundamental network topology, compared to IP multicast.

ALM was deployed in conferencing systems (Hosseini *et al.*, 2007b), since it can distribute data signals in a fast way. As an example, Luo *et al.* (2007) and Ogirima *et al.* (2014) proposed video conference systems based on P2P architecture. Luo *et al.* (2007) built multiparty video conferencing system taking into account the heterogeneity. Ogirima *et al.* (2014) offered a smooth video conferencing with low delay and short freezes.

Nowadays, ALM is deployed by Content Delivery Networks (CDNs) where multicasting is achieved on the application layer level without involving the network layer, in order to deliver data streams. Although, Since CDN nodes deliver data in unicast mode to the clients, it represents a weakness for the bandwidth usage comparing to IP Multicast (Rückert *et al.*, 2015).

1.2.2.3 Web Real-Time Communication (WebRTC)

WebRTC is a communication standard resulting from a joined project between the World Wide Web Consortium and IETF (Jennings *et al.*, 2013).

This standard permits the P2P exchange of real-time media through a web browser (Zeidan *et al.*, 2014). These media can be accessed through a JavaScript API, enabling developers to easily implement their own real-time web applications. Before WebRTC, to establish any audio or video communication, users needed to have an account on the application site or have to install plugins. WebRTC allowed browser-to-browser communications that do not require any software installation or registration for any type of calls, media sharing or gaming.

WebRTC is used in various applications. It started to gain popularity with Gmail video in 2008 then Hangouts and Ericsson in 2011. Nowadays, WebRTC is supported by many applications, such as Whatsapp, Facebook messenger, etc. WebRTC can be run on smartphone operating systems, such as iOS and Android, or on browsers, such as Googles Chrome, Mozilla Firefox, Opera, Safari, Vivaldi. Many mobile operators also support WebRTC, such as KDDI, AT&T. Telefonica, Deutsche Telekom, TokBox.

In WebRTC, developers are not limited to a specific signaling protocol since it maps directly to the PeerConnection that enables audio and video communication between peers. The signaling abstraction in WebRTC allows more compatibility with different technologies. To ensure signaling it needs to exchange "session description" objects. These objects represent the necessary transport and media configuration information necessary to establish the media plane, such as, what formats the participant supports and what does he want to send and the network information for the peer-to-peer connection. WebRTC uses SDP to communicate media metadata and ICE framework for finding network interfaces and ports in order to connect peers. If the connection fails using peers addresses due to Network Address Translation (NAT) devices and firewalls, ICE uses Session Traversal Utilities for NAT (STUN) server to obtain public IP addresses and the type of NAT. If that fails too, traffic is routed via a Traversal Using Relay NAT (TURN) relay server (Mahy *et al.*, 2010). Figure 1.6 shows the logic architecture of webRTC system.

However, to exchange media, a direct path is enabled between browsers without any need for a server. The transport protocols used in WebRTC are TCP, UDP and SCTP (Rahaman, 2015). Users' devices are responsible for the quality of the flows by using codecs able to adapt to the network condition. WebRTC supports various voice codecs such as G.711, G.722, Opus, iLBC and iSAC, and video codec such as VP8.

Even if WebRTC shows successful capabilities, it showed more success for one-to-one communication between a pair of participants, but it faced some issues with applications that rely on an intermediary (e.g., central server), as in conferencing systems, and VoIP systems when it is interacting with SIP. Some works, as in Amirante *et al.* (2013) and Zeidan *et al.* (2014) created a solution for more interoperability between webRTC and SIP for video and audio conferencing. Segeč *et al.* (2014) also worked on the integration of SIP and WebRTC to extend a new type of integrated communication environment. It

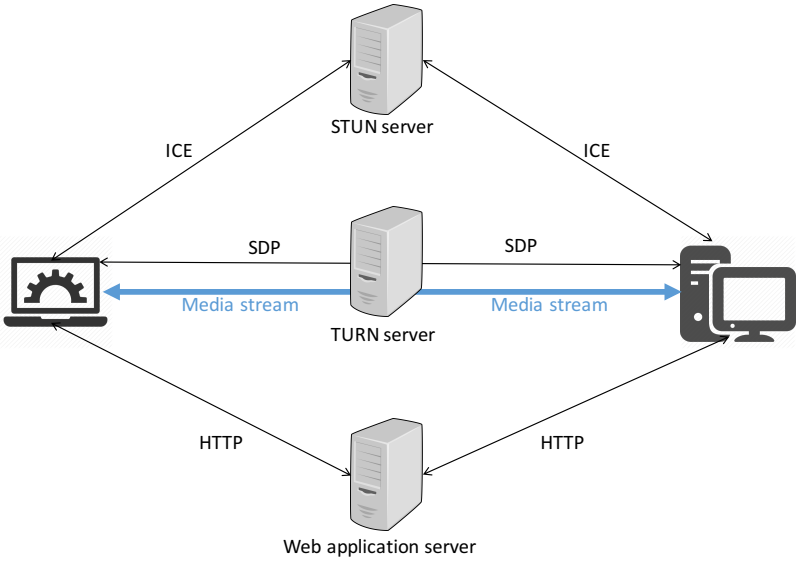


FIGURE 1.6: WebRTC architecture

provided a technological overview describing the aspect of the integration of these two protocols. Some academical works also focused on WebRTC integration with SIP, such as Universidad Politecnica de Madrid who developed a WebRTC video conference system, and Illinois Institute of Technology who developed Voice and Video on the Web (VVoW) (Deshpande and Mohani, 2015). Elleuch (2013), Edan *et al.* (2017) and Wang and Mei (2017), created a new models of a cross-platform multimedia conferencing system based on WebRTC technology. Currently, WebRTC technology is still trying to remove any plugins necessary for conferencing systems, however, this idea is still in its early stage. Apu *et al.* (2017) used P2P connection with the WebRTC architecture to create a video conferencing system that does not need any plugins or third party software. Their system is more scalable and incurs less infrastructure cost.

1.3 Protocols

1.3.1 Network Layer

1.3.1.1 IP multicast

Since unicast consists of sending replicated packets from a source host to each destination, the traffic load in the network becomes very high. Thus, unicast is ineffective in terms of network resources and it presents scalability issues. This led to the creation of multicasting as a network layer protocol and service (Deering and Cheriton, 1990). Multicasting is a technique that relies on transmitting information from one or various sources, at the same time, to multiple destinations through a multicast tree. Unlike unicast, it

saves a significant amount of bandwidth since there is no redundancy while transmitting data between pairs of users. The packets are only duplicated when a stream needs to be split in order to reach receivers in different leaf nodes of the network. To ensure efficiency and flexibility, many applications had to implement multicasting in their architecture, such as VoIP, Video-on-demand and video conferencing.

The multicasting implementation started on the network level with IP Multicast (Sahasrabuddhe and Mukherjee, 2000; Diot *et al.*, 2000). The senders, first, transmit media to a multicast group address, then the receivers join certain multicast groups. Network devices (Multicast routers) take responsibility for transmitting data through multicast trees. The data is transmitted once from the source, then, it is duplicated on its way to reach each and every destination (Gu *et al.*, 2015). In IP multicast, neither the sender nor any single point is aware of the receivers and their characteristics, which makes IP multicast scalable.

IP multicast and unicast can be used together in order to benefit from the central server and the multicast advantages. This is done by applying a unicast stream between the sender and the central server, then, applying multicast to distribute the traffic to the receivers.

The first multicast deployed model is Any Source Multicast (ASM) (Deering, 1988) used for dynamic multi-source sessions like conferencing and financial trading. The ASM model consists in allowing a receiver to receive from any sources belonging to its multicast group after joining via Internet Group Management Protocol (IGMP) specifically IGMPv2 (Fenner, 1997) or IGMPv3 (Cain *et al.*, 2002). To set up multicast distribution trees from the senders to the receivers, ASM uses Protocol Independent Multicast (PIM) (Farinacci *et al.*, 1998) and its variations. Another multicast model is Source Specific Multicast (SSM) (Cain, 2006). SSM consists in allowing a receiver to receive from a specific source. SSM helps in the case the source is known beforehand.

Chandrasekar and Baskaran (2011b) studied the performance of video conferencing in multicast communication using PIM. In their work, they compared the data transmission performance in unicast and multicast communication. The multicast communication showed the best performance with a high number of nodes. In their other work, Chandrasekar and Baskaran (2012) studied the effect of core failure in the performance of multicast systems. The results showed a significant increase in end-to-end delay when a core node fails. Researchers have involved IP-multicast in most of the conferencing architectures. However, IP-multicast still has difficulties to be supported by the conferencing applications vendors due to its deployment complexity.

1.3.1.1.1 Protocol Independent Multicast (PIM)

PIM is a routing protocol used for multicast. It does not rely on any routing protocol for unicast traffic. It assembles a multicast tree, using existing routing protocols, to allow data distribution from senders to all receivers. To build the routing table, PIM uses reverse path forwarding to create a unicast routing table. PIM contains five variations:

- **PIM Dense Mode (PIM-DM):** With PIM-DM (Nicholas *et al.*, 2005), it is assumed that every single segment is going to have someone that wants to listen to the multicast feed. PIM-DM is a push type of approach to distributing a multicast traffic. It goes over every single segment, even the ones with no interest with the multicast traffic. Then, it will prune back the none interested segments (the branches connected to receivers with no interest in multicast traffic). This result an inefficient sequence of flooding than pruning off. The first multicast routing protocol, Distance Vector Multicast Routing Protocol (DVMRP), used dense-mode multicast routing.
- **PIM Sparse Mode (PIM-SM):** Unlike PIM-DM, PIM-SM (Estrin *et al.*, 1998) is a pull technology. All the servers will direct their multicast to a Rendez-vous Point (RP) and the clients will pull their request of the multicast traffic from the shared tree approach. It is the preferred technology for disseminating the multicast traffic since it is scalable in large networks.
- **PIM Source-Specific Multicast (PIM-SSM):** The major drawback of any source multicast is that the receiver does not know about the source initially unless the RP provides him with this information. Knowing this information the shortest path can be built between the source and the receiver. RP in PIM-SM represents a point of failure that can be avoided using PIM-SSM. PIM-SSM (Bhattacharyya, 2003) allows the receiver to know about a specific source and send join messages directly to this source without going through the RP. IGMPv3 helps the router to learn about the specific source.
- **PIM Sparse-Dense Mode (PIM-SDM):** The knowledge about the RP is introduced manually to each router in PIM-SM. To be able to introduce it automatically, a method for dynamic RP address management is used. However, this method relies also upon the multicast group for its operation and need also a RP. To resolve this problem, Cisco invented PIM-SDM (Adams *et al.*, 2004) which uses the PIM-DM mode for disseminating RP information and the PIM-SM mode to run everything else.
- **Bidirectional PIM (Bidir-PIM):** Bidir-PIM (Handley *et al.*, 2007) supports many-to-many multicast applications by building bidirectional trees. The trees are not necessarily built on shortest paths which can increase the delay. However, it is very scalable since it is not based on a specific source. Bidir-PIM is very suitable for video conferencing.

1.3.1.1.2 Internet Group Management Protocol (IGMP)

IGMP is an elemental protocol used by IP multicast that runs on the network layer. It is responsible for the communications between the router and the end-device. IGMP is basically used on IPv4 networks, however, in IPv6, IGMP changes its name to Multicast Listener Discovery protocol (MLD) with almost identical characteristics to IGMP. IGMP can be used for conferencing, online streaming video and gaming, and allows the more efficient use of resources when supporting these types of applications. The two major goals for IGMP are: To indicate to the leaf router that a host wants to access the multicast traffic of a specific multicast group and to inform the local multicast router that the host wants to leave a multicast group. There are 3 different versions for IGMP:

- **IGMPv1** uses two specific message structures; the Report messages used by the client to join the multicast group and the Query messages used by the multicast router to check if the host still exists. The major drawback with IGMPv1 that it takes time to be aware if a host decided to stop receiving multicast and keep sending to him traffic.
- **IGMPv2** is the common and default version used nowadays. It brought a number of improvements to IGMPv1. The major improvement is the Leave Group message. It permits the host to notify the router of leaving the multicast group. The IGMP query interval in IGMPv1 was fixed to 60s, however, IGMPv2 has a tunable timer. In addition, IGMPv2 authorizes the selection of a specific router in case of having many routers capable of receiving the query messages. It also permits a router to separate two multicast groups and choose a specific group for sending queries.
- **IGMPv3** allows us to do Source Specific Multicast, which allows selection of the multicast server from wherever we want to receive the multicast traffic. This is applicable in the case of having multiple servers sending multicast traffic to the same multicast group.

1.3.1.2 OpenFlow

OpenFlow (McKeown *et al.*, 2008) is a communication interface between the control and forwarding layers of the SDN architecture. It allows direct access and manipulation of the forwarding plane, allows the network to be programmed on a per-flow basis, and it is easily supported by multi-vendors due to its standardization by the Open Networking Foundation (ONF). Figure 1.7 represents an OpenFlow-based architecture.

An OpenFlow device contains several flow tables to communicate with the controller. The number of flow tables can be as low as one table. Each flow table holds a number of entries. When a packet is received on an OpenFlow device, it passes along the first Flow table (number 0). If it matches an

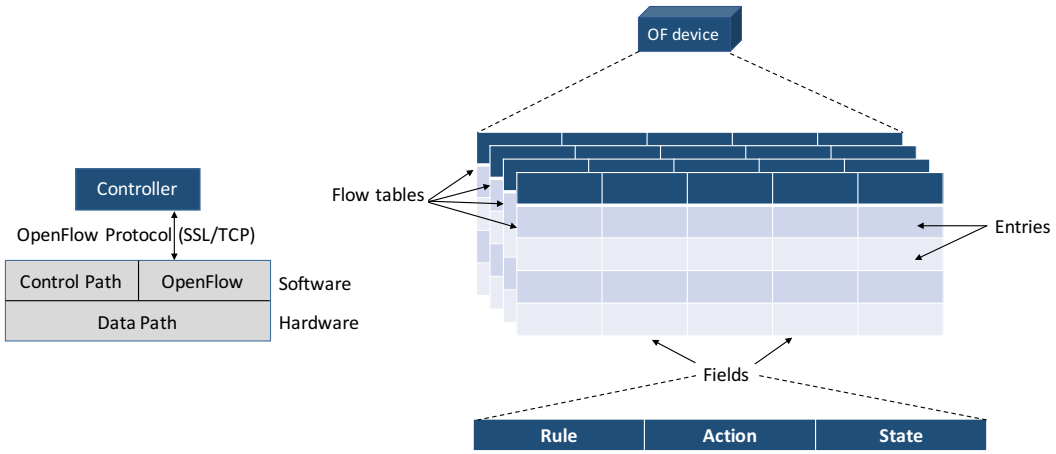


FIGURE 1.7: OpenFlow flow tables

entry in this table, it will execute the instructions in the match entry fields. These instructions may describe packet forwarding, packet modification and group table processing. In addition, the instructions may lead to another Flow table, the packet passes along the new table until it reaches the last flow table. If the packet didn't match any entry it will be sent to the controller. In this case, the controller creates a new flow with 0 or more new entries, sends it to the device to update its flow tables and sends back the packet to the device which can now recognize it. OpenFlow has a predefined message structure. These messages are called OpenFlow messages which are passed between the controller and the OpenFlow device. Every device will have its own Flow Table as opposite to the MAC address table and routing table in the traditional switches. Every new incoming frame/packet is matched against the existing Flow table of the device and takes the necessary action specified in the "Action" field of the Flow table. The protocol consists of three types of messages:

1. controller-to-switch messages sent by the controller,
2. asynchronous messages sent by the switch,
3. symmetric messages sent by either switch or the controller.

This fine-grained control of packets is very important to enable advanced functionality which could not be implemented by traditional devices.

Most conferencing systems have two separate channels for signaling and media and use a central server to control everything. The traffic load becomes a burden for the central server when the number of participants grows, and the server becomes a bottleneck due to the bandwidth limitation. Using SDN for conferences enables the separation between the control plane, which will be handled by the conferencing server, and the media plane, which will be

handled by OpenFlow switches. Using OpenFlow rules, media traffic does not require passing through the server every time which reduces the load on the server and makes the system more efficient, scalable and controllable.

1.3.2 Transport Layer

Multimedia applications are mainly using TCP (Postel, 1981), Stream Control Transmission Protocol (SCTP)(Fraczek *et al.*, 2010), Datagram Congestion Control Protocol (DCCP) (Kohler *et al.*, 2006) and UDP (Postel, 1980) as transport protocols for establishing a connection and ensuring media transfer.

TCP is an acknowledgment protocol that requires a connection setup between both sender and receiver parties. When the connection is established, the sender will start transferring data as segments. On the other hand, the receiver knowledge the capability of accepting the segments than arrange them according to their sequence number provided by the TCP connection. TCP allows the acknowledgment of the correct data reception and provides the capability of dropping duplicated segments or re-transmitting the missing ones. Even if TCP seems to be a robust protocol that ensures the control of data transfer and error correction, it can flood the network with large overhead due to retransmission that may consume a lot of bandwidth. In addition, the TCP mechanism can create buffering on the receiver side as well as losing a lot of data. Since buffering and recovering missing data cause latency, TCP is not considered to be very suitable for sensitive real-time communications.

Real-time communications need quicker and more efficient transmission protocol without the potential latency disadvantage of TCP. Thus, most of the real-time applications prioritize UDP over TCP for media transport. To avoid overhead, UDP does not use any of the TCP control features; In other words, UDP does not require an acknowledgment for connection establishment and does not guarantee any ordering or re-transmitting in case of missing data. Without any protocol overhead, acknowledgment and re-transmitting, UDP represents an efficient protocol for fast real-time communications and less demanding bandwidth, however, it does not ensure a safe data transfer.

To satisfy new applications' requirements, two transport layer protocols, namely SCTP and DCCP, have been designed to fill some TCP and UDP gaps. SCTP is a better-structured protocol than TCP and uses a four-way handshake procedure for connection establishment. SCTP provides multi-streaming features. Multi-streaming allows a user to define/receive video streams delivered in different resolutions, frame rates, and compression types to meet quality, storage, demands, codecs, and CPU requirements. On the other hand, DCCP is designed for applications that prefer combining the simple of UDP concept and TCP strict delivery order (Takeuchi *et al.*, 2005). DCCP has been mostly used for time-sensitive applications such as online gaming, VoIP, conferencing systems and video streaming. Many works have

tested the performance of DCCP for those applications. Chowdhury *et al.* (2009) showed that DCCP provides better performance for applications that suffer from the trade-off between delay and in-order delivery. Nor *et al.* (2017) compared the performance of DCCP to UDP, TCP, and SCTP over 4G network for video streaming. DCCP shows the best throughput improvement with the minimization of delay and jitter compared to UDP, TCP, and SCTP. The drawback of DCCP is that it is degraded when delivering data over long delay links (Nor *et al.*, 2012). Studies such as Lien and Ding (2011), Schier and Welzl (2012) and Rahman *et al.* (2012), showed that DCCP bandwidth performance is lower than the one delivered by TCP connections and has a hard time coexisting with TCP traffics in VoIP systems. Dunigan and Fowler (2004) and Tripathi *et al.* (2013) also tried to provide TCP functionalities to UDP in order to meet the needs in various networking scenarios and provide faster communication.

1.3.3 Session Layer

Signaling protocols are responsible for finding the user location, establishing the call session, negotiating about the session properties and managing the mobility of the participants in the session. In VoIP, a call can be defined as the multimedia session between two or more participants, while the signaling associated with a call is referred to as the connection. Signaling data is also exchanged between the two types of networks (e.g., a VoIP network and PSTN). Research found a successful way for inter-working VoIP systems with PSTN by using gateways. A VoIP Gateway is a network device responsible for transferring data packet between VoIP networks and circuit-switched networks. When the gateway is acting as a signaling gateway it transmits the packets to circuit-switching network in order to establish a call setup up, management and tearing-down. WebRTC also uses signaling protocols to set the connection between web servers. The most common standards used for signaling are SIP and H.323. SIP and H.323 are referred to as signaling protocols that work with IP networks.

1.3.3.1 Session Initiation Protocol (SIP)

SIP is a standard specified by the IETF Multiparty Multimedia Session Control Working Group (MMUSIC WG) in 1999 and was published as RFC 3261 in 2002 (Rosenberg *et al.*, 2002). It is used for conference call setup, modification and termination between two or more endpoints. It is used to handle sessions between two points and invites users to unicast or multicast sessions. Similar to Hypertext Transfer Protocol (HTTP) or Simple Mail Transfer Protocol(SMTP), SIP works on the top of transport protocols such as UDP, SCTP or TCP, in the application layers. Like HTTP, SIP adopts a client/server (request/response) architecture. The main SIP components are the User Agents (UA) a.k.a call terminals, and the five types of SIP network servers: proxy servers, redirect servers, location servers and registrar servers.

SIP network servers are responsible for the users' location, availability, capabilities as well as Session setup and management. To setup a call a User Agent Client (UAC) initiates SIP requests, a proxy server receives the requests and decides to which server a request should be forwarded. A redirect server notifies the UAC of the location of User Agent Server (UAS). It contacts a location server that keeps information about UAS location. The registrar servers, accept REGISTER requests from UAC and forward it to agent UAS. Since SIP relies on the user agent (client/server), it does not consider centralization control and it is only distributed.

SIP is used in most of the communication system architectures since it adopts a wide range of protocols such as transport protocols (UDP or TCP), real-time protocols (RTP and RTCP), etc. As mentioned in 1.2.2.1, many researchers integrated SIP with P2P communication systems to benefit from P2P features. With the creation of WebRTC that enables P2P communications, the integration and interaction between SIP communications and WebRTC became a hot topic. However, SIP integrating applications like audio and video conferencing still face programming and protocol challenges as well as it remained mostly based on standalone software and applications (Segeč *et al.*, 2014).

1.3.3.1.1 Session Description Protocol (SDP)

Session Description Protocol (SDP) (Jacobson and Handley, 1998) is a well-established format used by entities in order to agree on compatible media types and parameters for interaction like audio and video conferences. This is the reason why SDP is used by SIP and other protocols for defining session parameters. SDP provides the following information before starting a session: Session name and purpose, the time period the session is active for, the media types and formats that the clients would like to use (video format, ...) and where the other end should send the media to (e.g., user agents, IP addresses, transport protocols and ports). Having information about the clients will help SIP to better function. For example, if two clients do not support common codec they can not communicate with each other. Clients should support at least one common codec in order to communicate. If the clients support more than one common codec, the user agent in SIP can switch between one to another during the session.

1.3.3.2 H.323

In 1996, H.323 standard was approved by the International Telecommunication Union (ITU) as an umbrella of protocols that provide communication transmissions over IP networks (Recommendation, 1998). Unlike SIP, H.323 is not only considered as a signaling protocol for setting up the call, it also includes mechanisms for transmitting data (Glasmann *et al.*, 2003). H.323 is responsible for managing the administrative operations of a user,

call setup and tear-down using the H.225 protocol, and controlling a terminal's capabilities negotiation using the H.245 protocol. The main network architecture components that depend on the H.323 protocol are terminals, gateways, gatekeepers and MCUs.

Starting with the first component, terminals are the endpoints or clients or participants in the call session. Terminals must support the H.245 protocol that transmits endpoints capabilities information, as well as the Registration Admission and Status protocol (RAS) that allows the participants to register and find/disconnect from the gateway. In addition, they should also support RTP. Gateways are network devices that provide two-way communications, in real-time, between an IP network and PSTN. Gatekeepers are responsible for the call control and policy direction for terminals. Gatekeepers manage bandwidth by keeping information about the status of ongoing calls such as the bandwidth used and controlling the number of existing terminals. MCU mixes, switches and processes media streams as well as it controls the stream destination. For additional control, H.323 separates signaling control from the centralized network control in a distributed model or it relies on a centralized approach. H.323 can use application-layer feature control to enable the usage of new services on the top of H.323 protocol and endpoints.

To establish a call session, first, the terminals register with the gatekeeper. Gateways exchange terminals capabilities then a connection is setup between the terminals as well as a logical channel allowing data exchange. H.323 uses RTP encapsulation to transmit media stream as well as RTCP for control information. Finally, the terminals or gateways ask the gatekeepers for call disconnection using RAS.

H.323, is one of the best standards for audio, video and data transmissions as well as VoIP. It can control both point-to-point and multipoint conference calls. In addition, it allows the management of media traffic, bandwidth and user participation by the gateway.

1.3.3.3 Media Gateway Control Protocol (MGCP) and Megaco/H.248

Media Gateway Control Protocol (MGCP) is a VoIP signaling and call control protocol that was defined by the IETF in 1999, developed by Cisco (Arango *et al.*, 1999). MGCP is a centralized protocol used in VoIP systems. The three main components of MGCP are: the Media Gateway Controller, the Media Gateways and the Signaling Gateway. The Media Gateway Controller (MGC), also called the call agent, is responsible for the call control and the management of the MGs. The Media Gateways (MGs) provide a conversion between telephone circuits and network packets for audio signals and data packets. The Signaling Gateway (SGs) translate signaling messages from PSTN to IP. The call agent sends commands to media gateways in order to be executed. Thus, MGCP is considered to work a master/slave basis where the agent is the master and the MGs are the slaves. In order to setup a connection between endpoints, the call agent first sends a request to

the MGs asking them to create a connection with the endpoints and to send their information back to the call agent. Once the MG-endpoint connection is established, the call agent sends MGs information to other MGs in order to create a MG-MG connections. In the case where multiple call agents are needed, MGCP uses signaling protocols, such as SIP and H.323, to synchronize different call agents.

Megaco is another call control and signaling protocol, developed by Cisco to control the PSTN access by IP terminals (Cuervo *et al.*, 2000). It is an enhancement version of MGCP. It handles both signaling and session management for multimedia conferencing (Kaur and Kaur, 2015).

1.3.4 Application Layer

1.3.4.1 Real-time Transport Protocol (RTP)

The Real-time Transport Protocol (RTP) (Group, 2003) is a protocol usually used by real-time applications for delivering end-to-end audio and video streams. RTP was created by IETF's Audio-Video Transport Working Group to be used in VoIP applications with the association of signaling protocols (Jacobson *et al.*, 2003). RTP allows unicast or multicast data transmission over IP networks. The RTP standard is composed of two protocols; a protocol for real-time data transfer, used to exchange data, and another protocol for QoS control called RTP Control Protocol (RTCP) (Friedman *et al.*, 2003). RTCP allows the detection of packet loss, jitter delay, timing restriction and other common transmission problems of large multicast networks. Since RTP works on the top of UDP as well as other transport protocols, and it can be associated with SIP or H.323. Audio/video codecs encode the media to be transported in RTP packets which will be further encapsulated in UDP/TCP and finally IP packets. RTP supports a wide type of real-time applications. RTP profile for audio and video conferences with minimal control was proposed in Schulzrinne and Casner (2003) to make RTP specification more suitable for audio and video conferences.

In real-time applications, such as conferencing, some streams require conversion in order to reach lower bandwidth required by the receiver. In this case, RTP uses a translator, situated in between the sender and the receiver, to convert the streams. Audio and video streams are sent separately, via a unicast or multicast connection, through two pairs of audio and video ports. The pair of audio ports (same for video) contains one port for sending RTP streams and another for sending RTCP streams.

1.3.4.2 Real Time Streaming Protocol (RTSP)

The Real-Time Streaming Protocol (RTSP) (Schulzrinne *et al.*, 1998) is an application layer protocol used for audio and video streaming control in real-time applications. RTSP is used to establish and control media sessions between client and server. The transmission and the streaming itself is not a task of the RTSP protocol. RTSP uses TCP connection to maintain an end-to-end connection. The RTSP messages in RTSP are exchanged between the client and the server. RTSP works similarly to HTTP. To retrieve a type of information from the server, the client sends a request option to the server. The server returns an acceptance for the request option. Then, the client can ask for media description, the server in this case return information about multicast addresses and ports, in case of multicast communication, or it sends only the destination in case of unicast communication. To establish a connection, a setup request specifies how the media stream must be transported. After connection establishment, the client can tell the server to start sending bitstreams via the transport mechanism specified in the setup request.

1.4 Codecs

Since the start of digital media, audio and video formats are developed every year in a attempt to provide improvement in quality and file size. The popularity of these media, especially videos, continues to grow rapidly. Usually videos are received in containers. A container for a video file is a part that contains all the other files needed to play a video. Theses files includes a video stream, an audio stream and the meta data. The video stream will tell the player what needs to appear on the screen, while the audio stream which sound should be played along side the video. The metadata includes all the other information about the video such as, bitrate, resolution and most importantly the codec used. Thus, the heaviness of video streams affects the network bandwidth and need to be compressed in order to send through the network. The compression mechanism is done using codecs.

The word codec in a combination of the words "**Co**der" and "**Dec**oder" used to create an encoded video or audio stream by compressing it to create a smaller and easy to manage stream. When the player device or the target software receives the compressed version of the stream, it decodes it based on the rules set by the codec and plays back the media with a similar quality to the original.

There are hundreds of different codecs used on audio and video files. The codecs differ by a number of parameters, including the supported bitrates, the encoding/decoding algorithm complexity and the capacity of handling less data losses and errors as the most important parameters that identify a codec.

Multi-party applications such as conferencing systems, represent a challenge for the codecs due to the heterogeneity of the participants in terms of their device and bandwidth capacity and power constraints. Codecs are used, in this case, to adapt the audio or video streams and ensure a quality that satisfy the users. In this section, we list some of the most important audio and video codecs.

1.4.1 Audio codecs

1.4.1.1 MP3

MPEG-1 Layer III, aka MP3 (Brandenburg, 1999) is one of the most famous audio codecs developed by the Moving Picture Experts Group (MPEG) in 1993. It is a lossy audio codec that takes advantage of the limitations of human hearing, also referred to as auditory masking, in order to save space without noticeable quality loss. Thus, MP3 is often reduced to 128 Kb/s which sounds close to the original audio, while it only is 9% of the file size. Until now, MP3 continues to be a popular format for sharing and playing back audio content.

1.4.1.2 internet Speech Audio Codec (iSAC)

iSAC is an audio codec developed by Global IP Solutions in 2011. It can adapt to the bandwidth variation with a bitrate encoding range that varies between 10 kbit/s to 32 kbit/s (wideband) or 10 kbit/s to 52 kbit/s (super-wideband). And it covers audio sampling rates of 16 kHz (wideband) and 32 kHz (super-wideband). It is used in many VoIP and streaming audio applications, such as AIM Triton, the Gizmo5, QQ, and Google Talk. It is now included as a part of the WebRTC project.

1.4.1.3 Opus

Opus (Valin *et al.*, 2012) is a royalty-free codec for audio compression, created by IETF in 2012. It is a combination of Skype's SILK (Vos *et al.*, 2010) codec, based on linear prediction, and Xiph.Org's CELT codec (Valin *et al.*, 2010). The main characteristic of this codec is that it is not affected by Internet connection variation during the call since it adapts the stream bitrate to this variation on the fly. Opus covers a wide range of real-time Internet which gives it privilege comparing to the existing audio codecs applications (Valin *et al.*, 2016). Opus has a wide bitrate encoding range that varies from 6 kbit/s to 510 kbit/s, frame sizes that vary from 2.5 ms to 60 ms, and various sampling rates from a Narrowband audio quality starting from 8 kHz to a Fullband audio quality of 48 kHz (Rämö and Toukomaa, 2011). Opus is supported by operating systems (e.g., Google, macOS High Sierra, iOS 11, Windows 10 and multiple Unix operating systems), media players, Browsers

for WebRTC applications (e.g., Mozilla Firefox, Chromium, Google Chrome, Blink-based Opera and Safari), hardware (e.g., Apple iPods and IP Phones).

1.4.2 Video codecs

1.4.2.1 H.264 Advanced Video Coding (AVC)

H.264 AVC (Wiegand *et al.*, 2003) is the most commonly used video codec because it provides a significantly better bitrate over its predecessors (H.263) for the same file size. For this reason, it is very widely supported for video conferencing, mobile video and high definition broadcast.It was standardized in 2003 as a non-scalable encoding that presents different levels. Each level is defined by a certain resolution, frame rate and bitrate. H.264 AVC also specifies profiles for each application. These profiles indicate the algorithms and the coding structure to code and decode the video streams.

1.4.2.2 H.264 Scalable Video Coding (SVC)

H.264 SVC (Schwarz *et al.*, 2007) is an extension of H.264 AVC approved in 2007. SVC is used for video streaming, conferencing, surveillance, broadcast and storage. The idea of SVC is to transfer the video through several streams. Each stream has different characteristics. Thus, SVC allows participants with different capacities to participate in one video conference. Each participant will accept the corresponding SVC streams and drop the ones higher to its capacity. The video stream is split into different layers of quality. A layer of higher quality is set on the top of the one of lower quality. All the layers are built on the top of a Base Layer (BL). BL must contain the stream with the lowest quality acceptable. All the layers travel together from the sender to the receiver in a conference. If a receiver has lower capacity than the sender due to network low network reception or limited device capacities, it will choose to receive all the layers starting from the BL to the layer equal or lower to its capacities. And it will drop all the layers higher to its capacity. This procedure is simple and it only needs to drop unnecessary layers without encoding them. Layers are built using different criteria: the frame rate (Temporal scalability), the picture resolution (Spatial scalability) and coding quality or bitrate per pixel (Quality scalability). Thus, H.264 SVC is scalable and more flexible across networks than its predecessors. The drawback of SVC is the fact that it adds up to 20% more overhead (Avramova *et al.*, 2007) to the streams and it still not fully standardized and implemented only by few vendors (e.g., Polycom, Radvision and Vidyo). However, Google Hangout, Skype, and iChat adopt SVC for video encoding and adaptation according to a study done by Xu *et al.* (2012). The results of this study show that using SVC can improve user experience of Google Hangout application. Some works, such as Castellanos *et al.* (2017), Klaue *et al.* (2003) and Detti *et al.* (2009), created tools in order to evaluate SVC impact on the video quality in video streaming and video conferencing applications.

1.4.2.3 H.265 High Efficiency Video Coding (HEVC)

H.265 HEVC (Sullivan *et al.*, 2012) a successor of H.264 AVC with a double compression rate. This means that the file encoded with H.265 HEVC is at least 50% smaller than the one encoded with H.264 AVC. This is extremely beneficial for resolution above 2K as well as live streaming. The flip side of HEVC is that it is much more complicated than to encode, requiring triple the resources for the video to be prepared for playback. Just like H.264, H.265 is a presbytery codec and has a royalty associated with its usage. Although, H.265 is still not widely supported like its predecessors.

1.4.2.4 VP9

VP9 is developed by Google to be a royalty-free and open-source codec. Originally, it was used for YouTube, because it reduces the bitrate by 50% more than it predecessor **VP8** (Bankoski *et al.*, 2011) which is also developed by Google and published by IETF. VP8 resists to frame loss and deliver a rapid decoding, however, it only supports temporal scalability. Just like H.265 HEVC, VP9 is good for resolution and live streaming (Mukherjee *et al.*, 2013). However, it is harder to encode and less supported than H.264. The technology behind VP9 generally makes streams more consistent and reliable, while H.265 HEVC usually provides better image quality. VP9 commonly uses the WebM and IVF containers.

1.4.2.5 AOMedia Video 1 (AV1)

After VP9, Google created a new codec called VP10. On the other hand, the Alliance for Open Media (AOMedia) collaborated with Google to create a new codec called AOMedia Video 1 (AV1) which is a combination between VP9 and HEVC (Topiwala *et al.*, 2017). AV1 overtook V10 (successor of V9) reputation and aims to be a royalty free video format for the web. It provides higher resolution than H.264 which makes it interesting for real-time application such as WebRTC. AV1 is supported by web browsers such as Safari, alongside the Opus audio format, as well as Firefox, and it will certainly be adopted by others.

1.5 Summary of the state of the art

The demand for high-performance conferencing solutions is growing for both business and individual use. Developing and deploying these solutions require either using current conference architectures and protocols or creating new ones. In order to improve audio/video conferencing services, designers and researchers mainly worked on enhancing:

- the call establishment and control,

- the data distribution, and
- the streams' adaptation

For call establishment, most teleconference architectures use signaling protocols such as SIP. In a MCU-based conference, Multi-point Controller Units (MCUs) are responsible for handling signaling and control exchanges. MCUs are centralized and operate from a single location. Likewise, VoIP also relies on a central server for call control. The problem with these systems is that the call control is based on a centralized unit which can suffer from a single point-of-failure. Contrarily, distributed architectures, such as P2P and ALM, use distributed protocols for call setup and control. However, SIP for P2P-based conferencing is still mostly based on a standalone software and it faces a lot challenges. Currently, new conferencing systems have been developed based of WebRTC. WebRTC uses SDP (based on SIP) to communicate media meta-data and control the call. However, WebRTC also has problems when interacting with SIP. The development of SDN opened the way to new ideas for improving conferencing systems. Even if the call establishment still has to pass through a centralized controller, the control complexity will be reduced since the controller has a global view of the network.

For data distribution, conferencing architectures mainly rely on IP unicast or multicast distribution. Most MCU-based conferences use unicast connections to transmit streams from the sender to the MCU, and from the MCU to the participants. Using unicast loads the network especially in the case of multiparty conferences with a large number of participants. Thus, real-time conferencing systems have tried to use IP-multicast in order to save bandwidth. ALM has proposed to use transport layer connections to build a non-optimal tree that connects all the participants. The data distribution between the participants is done via unicast connections through relaying participants. IP-multicast is appropriate for bandwidth savings, however, it lacks control and suffers deployment problems which prevent it from being widely supported by conferencing systems vendors. SDN can solve IP-multicast control and deployment problems by enforcing a separation between the control plane and the data plane. The controller in this case is responsible for creating multicast trees and controlling the network, however, the data distribution is done using OpenFlow directly inside the switches.

In a conferencing application, participants use different devices with different capabilities through different access links. The heterogeneity of the participants makes it harder for the conferencing system to provide good media quality to all the participants. For example, in the case of an access bandwidth limitation or a low device resolution capacity in any of the participants, a high quality stream must be adapted to meet this participant's limitations. Stream adaptation can be done at the endpoints or on an intermediate node inside the network. MCU-based conferencing systems adapt the stream using transcoders implemented in the MCU. After the adaptation, the same adapted stream is distributed to all the participants regardless of the

receivers' limitations. That causes problems in case of heterogeneous participants with different access characteristics or device capabilities. In VoIP and ALM systems, the stream will be adapted when reaching a participant's device or on a intermediate box just before the participants. In both cases, the stream sent in the network will remain the same through the communication process until reaching the intermediate box or the participant device, where it will be adapted using the appropriate codec. To avoid this situation, P2P systems have allowed the adaptation of the stream on any node of the P2P network by using SVC. However, this will happen if the stream goes through a node with limited capacity on its way to the destination. In this case, the adaptation will affect all the consecutive nodes.

After this comprehensive state of the art, we have noticed that SDN can be beneficial for stream adaptation inside the network while taking into consideration all of the participants' limitations. The global view of the network allows the controller to detect the participants' limitations, create near-optimal multicast trees and choose the best locations inside the network to adapt the streams' bitrates. By doing so, we are able to ensure the best quality for each participant.

Chapter 2

SDN-Driven Multicast Streams with Adaptive Bitrates for VoIP Conferences

With the advent of the Internet as a cheaper alternative to the private telecommunication networks, VoIP technologies have gained a wide acceptance among individual users and organizations. VoIP conference calls, also called Multiparty VoIP (MVoIP) calls, were soon developed along the way. However, as the Internet was not initially designed to enforce strict QoS requirements, VoIP and MVoIP applications needed specific mechanisms to achieve a proper QoE for the users. For example, VoIP streams can be heavily degraded in the presence of heavy background traffic or harsh network conditions if no provisioning or priority is given to them. In addition, the connection characteristics of the client may vary widely among the participants when a conference call is set up. New audio/voice codecs, such as Opus (Valin *et al.*, 2012), are capable of adjusting the bitrate of the audio stream on the fly, allowing a dynamic optimization of the stream to the characteristics of the connection. However, this adaptation is currently occurring at the users' devices or at a centralized unit, and there is no possibility of changing the stream bitrate in the core of the network. In order to leverage multicast, we need change the bitrates of the streams at the points of duplication inside the network. To solve this problem, we propose a new centralized algorithm that builds MVoIP sessions according to the desired reception bitrate of each participant, while minimizing the bandwidth usage in the network. The implementation of our algorithm is intended to be deployed in an SDN controller.

In this chapter, we propose a solution for reducing bandwidth consumption in multiparty VoIP (MVoIP) systems. This solution uses SDN in order to facilitate the management of audio streams to enable the bitrate adjustment in the core network.

In this chapter we present:

- a presentation of our MVoIP model in an SDN environment (Section 2.1).

- the description of our algorithm for building the multicast trees of the voice streams from each participant to all the others (Section 2.2).
- an evaluation of our proposal by running simulations, and a discussion of the results showing the gains in bandwidth usage and the impact on path latency (Sections 2.4 and 2.5).

2.1 Our MVoIP model

In an MVoIP call, the users can be affected by the channel limitations. A simple solution to this is to limit everybody's voice codec bitrate to match the conditions of the worst receiver. However, there will be an unnecessary loss of quality affecting the users that have good channel conditions. Another solution is to establish unicast streams between each pair of participant, where each session matches every peer's receiving capacity. However, this solution will lead to a wasteful increase of bandwidth usage. To avoid the issues from the unicast solution, we have designed a new MVoIP model in an SDN environment. Our model uses SDN features in order to propose a new algorithm leveraging multicast trees and transcoding functionalities in the core of the network.

2.1.1 Our model design

An MVoIP session or call is a multiparty VoIP conference between three or more clients called participants. Each participant is both a sender to, and a receiver from all others. We assume that each participant is using a mobile device connected through a wireless access network, hence the access bandwidth is supposed to be scarce. The core network is SDN-enabled. Figure 2.1 depicts an example of SDN deployment that transports one MVoIP call. The streams are transcoded along the way (i.e., the bitrate of the stream is reduced). The deployment is composed of OpenFlow-enabled switches that are connected in-band to an SDN controller. End users are connected via base stations to the SDN network, and can place VoIP calls between them. The end users' devices use voice codecs that can pick from a range of streaming rates. Typically these rates range from a few kbps up to a few tens of kbps. One of the common lowest rate is 8 kbps (e.g., for Opus (Valin *et al.*, 2012), AMR (Varga *et al.*, 2006), G279 (Salami *et al.*, 1997)), while the upper rate can go up to 64 kbps (e.g., for G711 (Rec, 2008)). As changing the bitrate a little has no real effect on the audio quality, the possible bitrates usable by the devices are usually quantified into a few meaningful separate bitrate levels. As depicted in Figure 2.1, access links are wireless. They are sensitive to signal interference and have variable communication channel conditions. Generally, the access links have a fixed limited capacity, thus sometimes hindering users from either sending or receiving at peak codec bitrate. Higher bitrates are

preferred when the network conditions allow it, as typically call quality is correlated with codec bitrate.

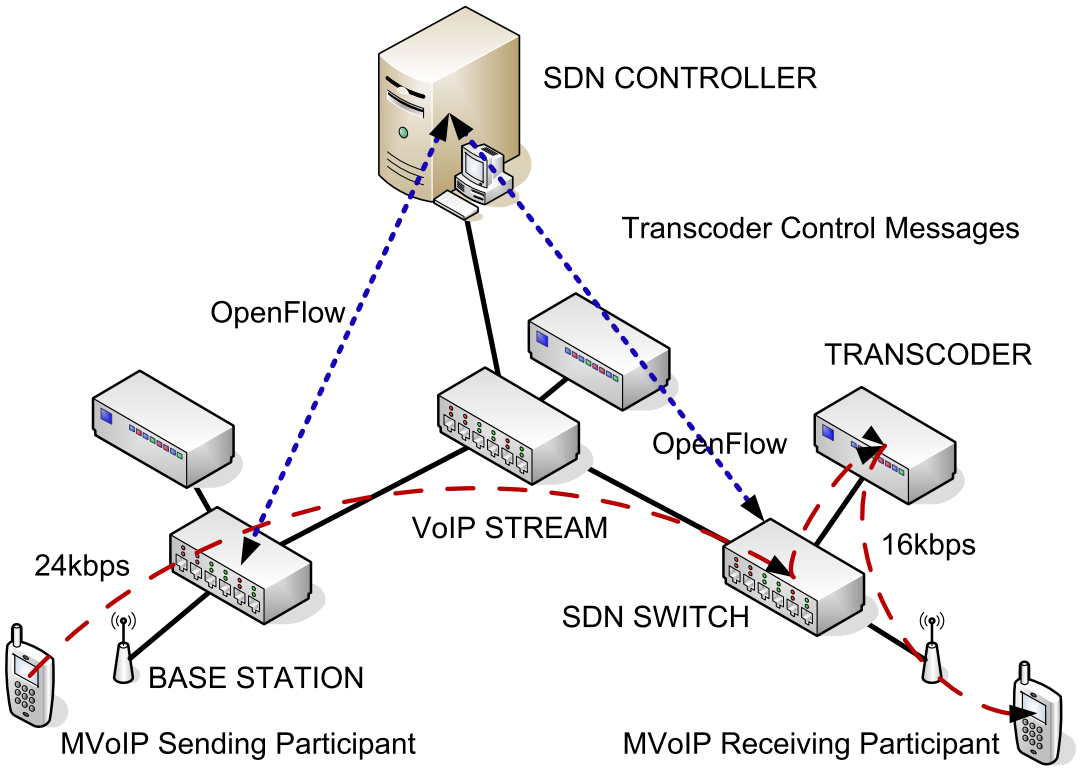


FIGURE 2.1: An example of SDN MVoIP model architecture

2.1.2 Our proposed solution

Each participant will send at a maximized codec rate, in order to satisfy all participants with good receiving conditions, whereas the participants affected by channel conditions will be served by a downgraded version of the same stream. The downgrading should happen at appropriate locations in the network. Those locations are the topic of the next section where our algorithm is explained. Thanks to SDN, the networks can be observed now globally from a central unit (the Controller). Once the controller collects the information of network topology and channel conditions, the MVoIP scenario can be tackled more efficiently. For example, a sender's codec may operate at 24 kbps, even if the receiver can only receive 16 kbps. As the controller is aware of the channel conditions of the receiver, it activates a transcoder attached to an OpenFlow switch that will downgrade the stream to 16 kbps. The transcoder is defined as a device which is able to reduce the bitrate of audio/voice streams on the fly. We envision the transcoder as a computing machine that is connected to a nearby OpenFlow switch. The transcoder needs to be instructed about which rate to transcode to, and that can be done in at least two different ways. One option is for the transcoder to be managed by

the SDN controller using extended OpenFlow messages, as depicted in Figure 2.1. This could be done using the Experimenter Field in the OpenFlow specification, while the OpenFlow process inside the transcoder can be run by OpenVSwitch. The coding rates at which streams have to be transcoded are given by the SDN controller directly to the transcoder. Another option is for the transcoder to be a simple machine that will listen to incoming voice packets and transcode them. The desired transcoding rate can be signaled by using different port ranges for different desired codec rates. For example, any packets arriving at port x to port $x + 1000$ should be transcoded to 8 kbps, and so on. In both solutions, the SDN controller creates a flow entry in the OpenFlow table of the switch. When the flow attends the switch, the packets execute the OpenFlow "action" corresponding to the entry. In our case, the "action" consists of forward the packets of the stream that needs transcoding to the outgoing interface towards the transcoder.

2.1.3 Example

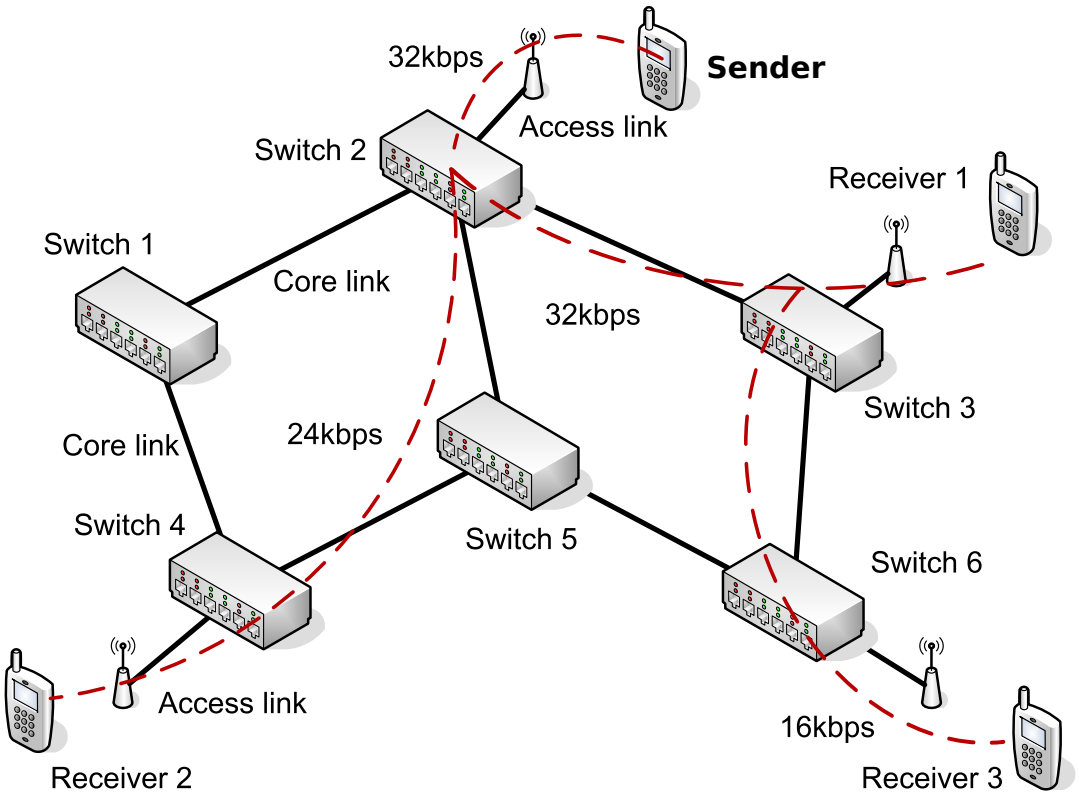


FIGURE 2.2: MVoIP multicast tree for a sender

We aim to illustrate the idea of our algorithm through an example. The scenario is depicted in Figure 2.2. Each participant needs to build a multicast tree to reach its peers, and for simplicity, we depicted one such tree for the participant marked as a sender. In our scenario, the sender tries to reach

all its three receivers. As we use SDN, the controller collects information of all participants, their channel limitations, and the network topology. In our example, receiver **#1** can receive at 32 kbps, receiver **#2** can receive at 24 kbps, while receiver **#3** can only receive at 16 kbps.

The algorithm starts by looking for receivers that do not need transcoding. As receiver **#1** can receive at the original codec bitrate, the stream is forwarded there without having to be transcoded. Then the algorithm proceeds by looking for participants that can receive at the next bitrate downwards. In our example, it will find receiver **#2**. Thus, in the path between switch **#2** and **#4**, a transcoding transformation is necessary. We depicted switch **#2** as being such location. The algorithm has one more receiver to treat, i.e. receiver **#3**. Since there are already two streams in the network that could be transcoded to receiver **#3**'s requirements, the algorithm has to make a decision. In our case, the algorithm chooses to transcode the stream serving receiver **#3** at switch **#3** and forward it to switch **#6** which finally connects to receiver **#3**. As mentioned, the algorithm will iteratively solve this topology for each participant while considering the other participants as receivers.

2.2 Our proposed algorithms

2.2.1 Technical assumptions

The algorithm that builds the MVoIP session containing the multicast trees of all participants (senders) to the others (receivers) relies on several assumptions:

- The network capacity is considered very high, much above the requirements for a single session, thus the bandwidth of core links is considered to be infinite.
- The SDN controller knows the complete topology of the network and the position of the participants.
- There is no mixing of voice streams inside the network, only transcoding is possible. A mixing of speech samples from different participants is required if more than one participant are talking at same time.
- There is one transcoder per SDN switch and it can transcode simultaneously any number of streams.
- Access link characteristics for all participants do not vary over the duration of the session.

TABLE 2.1: Model notations

Notation	Definition
n	Number of nodes in the network
m	Number of links in the network
$\mathcal{P}$	Set of participants
p	Number of participants
s_i	Sender i
T_i	Multicast tree associated to the sender s_i
R_i	Set of receivers in the tree T_i
$r_{i,j}$	Receiver j in the tree T_i
$B(x,y)$	Bandwidth of the link (x,y)
$b_i(x,y)$	Bitrate of the link (x,y) in the tree T_i
$B(r_{i,j})$	Downlink bandwidth of the receiver $r_{i,j}$
$b_i(r_{i,j})$	Downlink bitrate of the receiver $r_{i,j}$ in the tree T_i
$B(s_i)$	Uplink bandwidth of the sender s_i
$b(s_i)$	Uplink bitrate of the sender s_i
CB	Set of the bitrates level recognized by the codec used
$path$	Shortest path connecting a receiver to a sender or to the closest node of the tree
k	Node where transcoding is needed
$P_i(r)$	Parent of a node r in the multicast tree T_i
$C_i(r)$	Set of children of a node r in the multicast tree T_i

2.2.2 Formal model

We model the network as a graph $G = (V, E)$, where $|V| = n$ is the number of nodes and $|E| = m$ is the number of links. The set of participants is denoted by $\mathcal{P} = \{1, \ldots, p\}$. There is p multicast trees, one for each participant as a sender. Thus, a multicast tree T_i is associated to a sender s_i and a set of receivers $R_i = \{r_{i,j} \mid i \in \mathcal{P} \wedge j \neq i\}$. For simplicity, we consider that the participants are also nodes of the multicast trees (the receivers as leaves and the sender as the root). For any link specified by a pair of nodes (x,y), the bandwidth[1] (or capacity) is denoted by $B(x,y)$ and the bitrate (the current used capacity in a specified tree T_i) is denoted by $b_i(x,y)$. For simplification, $B(r)$ (resp $b_i(r)$) denotes the download bandwidth (resp. the bitrate in T_i) of the receiver r and $B(s)$ (resp $b_i(s)$) denotes the uplink bandwidth (resp. the bitrate in T_i) of the sender s. We note as CB the set of the bitrates level recognized by the codec used in the system. Table 2.1 summarizes the different notations.

[1]Note that it is possible that $B(x,y) \neq B(y,x)$ if the uplink and downlink bandwidths are not the same.

2.3 Setting a MVoIP call

In this section, we describe the algorithm used for building the MVoIP call session. In the next chapters, this algorithm will be adapted to video conferencing system and it will run periodically to adapt the session to the variations of the access links' characteristics. The algorithm consists of the following general steps:

1. Collecting the uplink/downlink bandwidth for each participant.
2. Computing the optimal receiving and sending bitrates for each participant.
3. Building a multicast tree from each participant to all the others.

In step 1, the controller backs up the network state. Then it retrieves the uplink/downlink bandwidth values of each participant by polling its access switch (the one through which the participant is connected to).

In step 2, the receiving bitrate of each participant $b_i(r_{i,j})$ is calculated by first dividing its downlink bandwidth $B(r_{i,j})$ by the number of participants excluding itself $(p-1)$, and then by picking the highest bitrate level lower or equal to this computed value. The receiving bitrate represents the highest bitrate which a receiver is able to receive from any sender. The sending bitrate $b(s_i)$ of the sender s_i is then defined as the maximum received bitrate level $b_i(r_{i,j})$ found among all the receivers. The bitrate $b(s_i)$ should be lower or equal to its uplink bandwidth $B(s_i)$. As we assume that the uplink bandwidth is always higher than the highest bitrate level, the sending bitrate is thus the same for each participant. However, the sending bitrate may be lower than the highest quality bitrate if no receiver can receive it. Eq.(2.1) summarizes the computation of the receiving/sending bitrates.

$$\begin{aligned} b_i(r_{i,j}) &\leftarrow \max_{k \in CB} \left\{ k, k \leq \frac{B(r_{i,j})}{p-1} \right\} \\ b(s_i) &\leftarrow \min \left\{ B(s_i), \max_{r_{i,j} \in R_i} b_i(r_{i,j}) \right\} \end{aligned} \tag{2.1}$$

In step 3, the controller builds a multicast tree from each sender to all the receivers.

2.3.1 The algorithm

Algorithm 1 builds multicast distribution trees and places transcoders at optimal locations, after computing the best receiving/sending bitrate of each participant.

First, Algorithm 1 takes as input a participant s_i and builds its diffusion tree T_i (line 1-3). The function `SortReceivers`(R_i, s_i)(line 4) sorts the receivers R_i first, form the highest receiving bitrate to the lowest, then from the closest (to the sender) to the farthest. For each receiver $r_{i,j}$, a shortest path is built depending on the mode (line 7-8). Two modes are defined :

- **Minimizing Spanning Tree (MST)**: in this mode, *BuildShortestPathToTree($r_{i,j}$, T_i)* builds a shortest path from a receiver $r_{i,j}$ up to the closest node already belonging to the tree T_i. Thus, MST builds a tree that minimizes bandwidth usage in the network (line 7).
- **Shortest Path Tree (SPT)**: in this mode, *BuildShortestPathToSender($r_{i,j}$, s_i)* builds a shortest path from the receiver $r_{i,j}$ to the sender s_i, potentially stopping at the first node of the tree (root). This mode builds a shortest path tree that minimizes the latency between participants (line 8).

Starting from the sender s_i, the first shortest path of the tree is built to the first receiver in the list of sorted receivers, regardless the mode. If no path is found, due to saturated links, the call is rejected. If a path is found, three cases are considered:

- $r_{i,j}$ is equal to the closest receiver to the sender with the highest bitrate $R_i[0]$. Thus, the computed path is the first in the tree. We simply add the *path*'s edges to the tree T_i with a bitrate equal to the sender bitrate $b(s_i)$. To add an edge between two nodes we use the *AddEdge(tree, (node1, node2), bitrate)* function that takes as parameters the tree to which we are adding the edge, the edge which is a couple of two nodes and the bitrate associated to this edge (lines 10-12).
- $r_{i,j}$'s receiving bitrate $b_i(r_{i,j})$ is equal to the sender bitrate $b(s_i)$; knowing that *path*, in this case, is not the first shortest path in the tree, its edges might overlap with others in the tree. Thus, before adding new edges, we must check if they already exist in the tree. The bitrate associated to each of the new edges is equal to the sender bitrate $b(s_i)$ which is the maximum bitrate that can be sent (lines 14-16).
- $r_{i,j}$'s receiving bitrate $b_i(r_{i,j})$ is lower than the sender bitrate $b(s_i)$; the edges are added from the receiver up to the sender stopping on the first overlapping edge. In this case, the bitrate associated to each of the new edges is equal to the receiver's receiving bitrate $b_i(r_{i,j})$. Knowing that the existing edges of the tree carry bitrate equal or greater than the current receiving bitrate $b_i(r_{i,j})$, we may need to drop some bitrate layers on the intersecting node k. Thus, using *Rule(node, receiver)*(explained

bellow), node k will drop some layers on k to attend the receiver bitrate $b_i(r_{i,j})$ (lines 18-22).

Algorithm 1: Setup of the Video conference Call

```
1  Input: a sender s_i with i ∈ P
2  Output: a multicast tree T_i
3  T_i ← {s_i}
4  R_i ← SortReceivers(R_i, s_i)
5  higestBitrate ← max(b_i(r_{i,j})) with j ∈ P/{i}
6  foreach r_{i,j} ∈ R_i do
7      path ← BuildShortestPathToTree(r_{i,j}, T_i) if mode = MST
         # Builds the shortest path from the receiver the closest node in T_i
8      path ← BuildShortestPathToSender(r_{i,j}, s_i) if mode = SPT
         # Builds the shortest path from the sender s_i
9      if path ≠ {∅} then
10         if r_{i,j} == R_i[0] ∧ b_i(r_{i,j}) == higestBitrate then
11             foreach edge ∈ path do
12                 AddEdge(T_i, edge, b(s_i))
13         else
14             if b_i(r_{i,j}) == b(s_i) then
15                 foreach edge ∈ path do
16                     AddEdge(T_i, edge, b(s_i)) if edge ∉ T_i
17             else
18                 foreach edge ∈ path do
19                     AddEdge(T_i, edge, b_i(r_{i,j})) if edge ∉ T_i
20                     k = edge[1]
                         # k is the intersection node of the added edge and the tree
21                     Rule(k, r_{i,j}) ← (b_i(P_i(k), k), b_i(r_{i,j})))
```

The goal of this algorithm is to minimize the consumed bandwidth. To do so, it places rules that transcodes from a higher quality bitrate as close as possible to the sender, thus saving the bandwidth in the core network. `Rule(`k`,` $r_{i,j}$`)` takes as input a node k and a receiver $r_{i,j}$ and precises that the bitrate will be transcoded to a lower bitrate at node k to reach the acceptable receiving bitrate $b_i(r_{i,j})$ (line 21).

2.3.2 Complexity

As the problem of generating an optimal multicast tree (also known as the Steiner tree problem) is NP-complete (Winter, 1987), we use a single source (sender) approach and the two modes mentioned above for defining a non-optimal heuristic that builds the tree backward from each receiver

to a given sender. We then iterate the procedure for every sender. We use Dijkstra algorithm for all nodes in the network, in order to obtain the shortest paths needed by our algorithm, The complexity of an all-pairs Dijkstra's algorithm on a sparse network of size n, with a binary heap implementation, is $O(n^2 \times \log n)$. This can be done upfront. For a video conference call with p participants and k video layers, on a network of diameter D, the time/computation complexity of our algorithm is $O(p^2 \times k \times D)$. Given that k and D are usually constant during a video call, and that $p << n$, the time complexity of our algorithm is very small.

2.4 Simulation of setting up a MVoIP call

In this section, we evaluate our algorithm for setting up an MVoIP call. We detail the simulation methodology and parameters, then we show the results assessing the efficiency of our algorithm. The algorithm is implemented in Python 2.7.10, using the NetworkX package [2].

2.4.1 Simulations methodology and parameters

In order to evaluate our solution, we need to model topologies of telco operators. As this information is hard to obtain, we have produced various maps of different sizes using three different topologies:

- Erdős-Rényi model (ER) (Erdös and Rényi, 1959); that generates graphs by connecting nodes randomly. The probability of including an edge is independent from every other edge.
- Magoni-Pansiot model (MP) (Magoni and Pansiot, 2002); where the degree distribution follows a power law (as on the Internet and in large communication networks). It is based on an algorithm that performs a sampling on measured Internet maps.
- Waxman model (WM) (Waxman, 1988); where the probability of creating an edge decreases with the distance. Each pair of nodes (u, v) at Euclidean distance d is joined by an edge with a probability of $\beta \exp \frac{-d(u,v)}{\alpha L}$, where L is the maximum distance between any pair of nodes and β, α are constants usually set to 0.4 and 0.1 respectively.

These three models define different ways of creating edges as shown in Table 2.2. Using three different topologies, allows us to evaluate the effect of the topologies on our algorithms.

[2] https://networkx.github.io/

TABLE 2.2: Topology Models

Topology model	Degree or edge probability	Reference
Erdös-Rényi (ER)	$p(deg(v) = k) = \binom{n-1}{k} p^k (1-p)^{(n-1-k)}$	(Erdös and Rényi, 1959)
Magoni-Pansiot (MP)	$p(deg(v) = k) \propto k^{-\gamma}$	(Magoni and Pansiot, 2002)
Waxman (WM)	$p(u,v) = \beta \exp \frac{-dist(u,v)}{\alpha L}$	(Waxman, 1988)

We have implemented Algorithm 1 in two different modes (MST, SPT) and compared them to the unicast (UNI), ALM and MCU approaches:

- UNI: In this solution, the sender emits streams to a receiver through a one-way shortest path. In this case, two shortest paths are created between each pair of participants.
- ALM: This solution, presented in Section 1.2.2.2, is usually based upon a hierarchical clustering of the application layer multicast peers. However, since our call sessions contain few participants (up to 12), the creation of the clusters is not necessary. Thus, we have implemented a simple form of ALM that creates a multicast tree by connecting each participant to the closest existent participant in the tree already computed. When the tree is built, the sender emits multicast streams to the other participants. The bitrate of the emitted multicast stream corresponds to the sender's capacity. Once the stream is received, each participant adapts it with respect to his own capacity (see Appendix A).
- MCU: All the participants are connected to the MCU node, as presented in Section 1.2.1.2. To build the tree of each sender, the algorithm first builds the shortest path from the sender to the MCU (sender-MCU). Secondly, it builds the shortest paths from the MCU to each receiver (MCU-receiver(s)). Then, it connects the first path (sender-MCU) to the others (MCU-receiver(s)) to get the final tree (sender-MCU-receiver(s)). A sender emits a stream with a certain bitrate to the MCU. Knowing the capacity of each receiver, the MCU adapts this stream by choosing the minimum bitrate between the initially sent stream and the receiver's capacity. We note that, while implementing MCU, the algorithm finds the shortest path from MCU to all participants and stores it in memory, which may affect the processing time of the algorithm.

In our simulation, we consider the following parameters:

- We assume the SDN network to be WAN-sized. The network topology sizes implemented are 500, 1000, 2000 and 4000 nodes. Which allows us to evaluate the impact of the size of the network on the efficiency of our algorithms.
- The access downlink (receiving) capacity between a participant and its access switch is randomly selected from 144 kbps to 384 kbps and the

uplink (sending) is set to 64 kbps, which are conservative real-life values observed in current 3G UMTS networks (Glabowski *et al.*, 2007).

- The core links are supposed to have an infinite capacity as we currently evaluate the efficiency of one session at a time. We leave the study of the usage of the overall network capacity for the next chapters.
- The access link delay are set to 20 ms (a typical average value as shown in (Laner *et al.*, 2012)) and core link delays are set to 5 ms (as observed in (Hoerdt and Magoni, 2005))

Table 2.3 presents the various input parameters used in our simulations. To obtain our results, we first start by creating 10 topologies for each topology type (ER, MP and WM) and for each network size (500, 1000, 2000 and 4000), using a different instance each time. Then, for each one of these topologies, we connect randomly the client with different access capacities, then we repeat this step 100 times. Thus, each point on the following plots is the average of the values obtained from 1000 simulations. This average is presented accurately using a confidence interval of 95% displayed on the plots. Table 2.3 summarizes the parameters values of this chapter.

TABLE 2.3: Simulation Parameters

Parameter	Values
Modes	Unicast, SPT, MST, ALM, MCU
Bitrate levels	(8, 16, 24, 32) kbps
Max number of transcoding per path	1
Placement of participants	Uniformly random
Access uplink bandwidth	64 kbps
Access downlink bandwidth	From 144 to 384 kbps
Access downlink bandwidth distribution	Uniformly random
Access link delay	20 ms
Core link bandwidth	unlimited
Core link delay	5 ms
Number of runs	1000

2.5 Simulations results

This section presents the results that were obtained by the simulations carried out with the parameters previously described. As stated above, each point is the average of 1000 measured values.

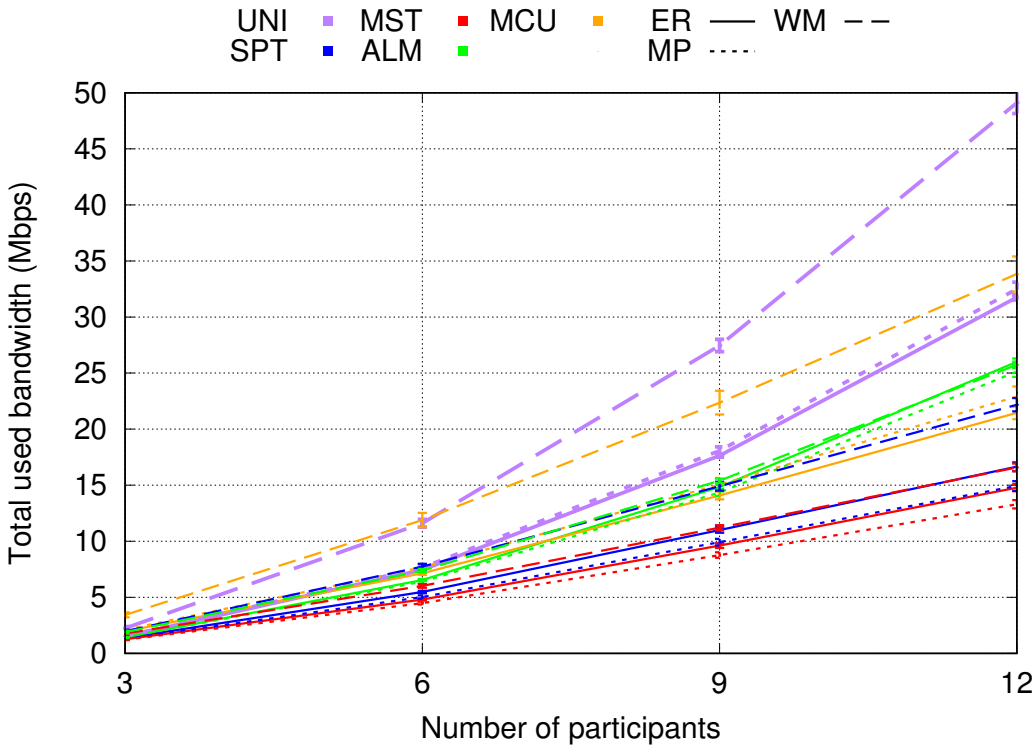


FIGURE 2.3: Total bandwidth consumption *vs* modes used, topology models and number of participants, in a 2k-nodes network

Figure 2.3 shows the impact of the number of participants and the topology model on the total bandwidth used by one session on average (only the audio data streams). The total bandwidth represents the sum of the used bandwidth of each link. All topology models show similar behavior with the number of participants. When the number of participants increases, the trees connecting them become larger, which increases the total bandwidth usage. As expected, the MST mode is the least bandwidth consuming, closely followed by the SPT mode. ALM and MCU modes show more bandwidth consumption than MST and SPT modes, however they are close to SPT modes in WM topology. The unicast mode, on the other hand consumes much more bandwidth. This is due to the redundancy of the streams on the shared links. For all modes, WM topologies consume more bandwidth than the two others topologies (ER et MP), because they mostly have very short links. Thus the average path length in WM topologies is much longer than ER and MP topologies, requiring more bandwidth (as bandwidth is computed as the sum of bandwidth consumed on every link).

With 12 participants, SPT bandwidth consumption is 23% lower than the MCU bandwidth consumption in ER topology and 40% lower in MP topology. Comparing to ALM mode, SPT bandwidth consumption is lower by a margin of, 35% in ER topology and 44% in MP topology, than the ALM bandwidth consumption. On the other hand, MST bandwidth consumption is 29% lower than the bandwidth consumed by MCU mode in ER topology,

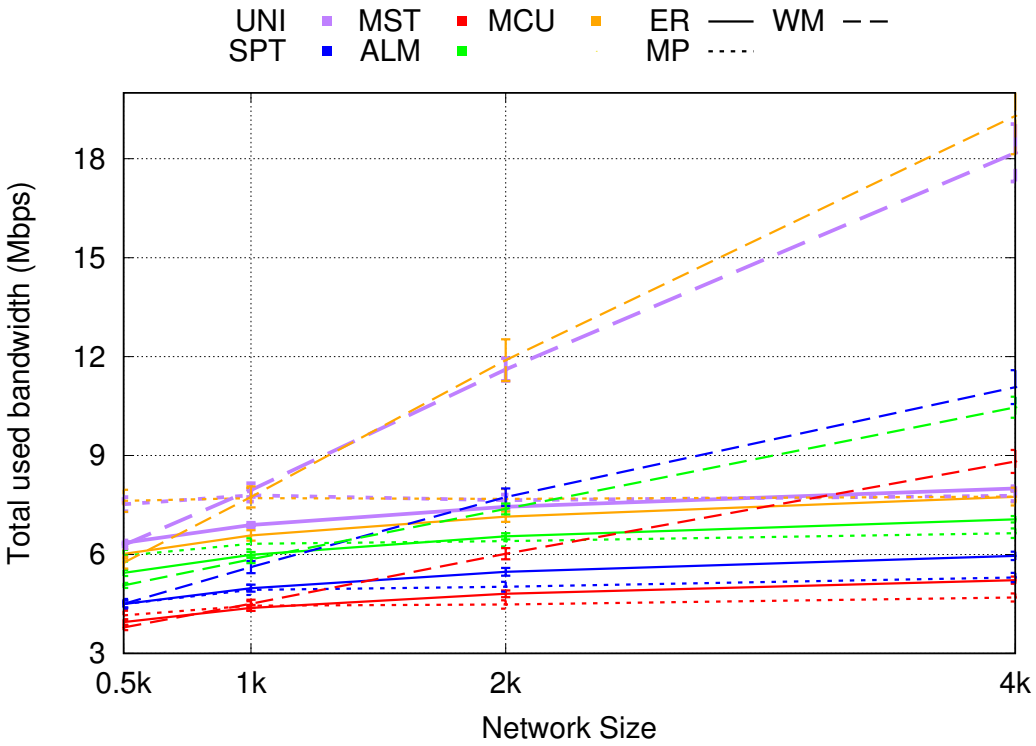


FIGURE 2.4: Total bandwidth consumption *vs* modes used, topology models and network size, in a call of 6 participants

and 22% lower in MP topology. Comparing to ALM mode, MST bandwidth consumption is lower by a margin of, 40% in ER topology and 28% in MP topology, than the ALM bandwidth consumption.

Figure 2.4 shows the impact of the network size and the topology model on the bandwidth. For the ER model and the MP model, an increase in the size of the network increases very moderately the bandwidth consumption. However, for the WM model, there is a nearly linear relationship between network size and bandwidth usage. This is expected because of the nature of WM model, where the number of edges is inversely proportional to the size of the network. Thus, when the network size increases, the distances linearly increase leading to increased bandwidth usage. The network topology characteristics of the SDN network will have to be thoroughly studied in order to control its impact on bandwidth usage. Our two modes (SPT and MST) show better results in MP and ER topologies than all the other modes and they are not very affected by the network size.

The impact of the number of participants and the topology model on the average path latency is shown in Figure 2.5. The latency takes into account access and core link delays but not codec and jitter delays. The SPT and unicast modes yield the same results whatever the number of participants, which is expected as they both use shortest paths. However, we observe that the values of the MST mode are higher than the other two modes and they increase when the number of participants increases. MST mode still show lower results than the ALM and MCU modes in MP and ER topologies. The

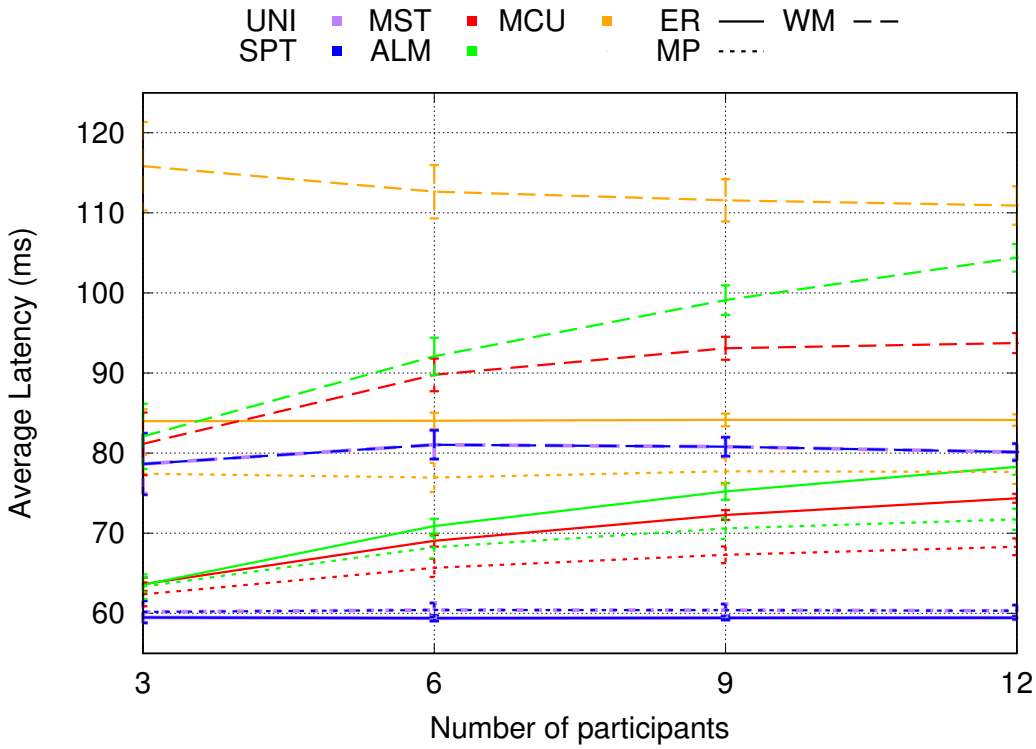


FIGURE 2.5: Average path latency *vs* modes used, topology models and number of participants, in a 2k-nodes network

average latency in MST mode is 12% higher than SPT mode in MP topology and 14% higher in ER and WM topologies. It must be noted that this latency represents communications between participants located inside the same network operator as stated in our hypotheses. Most of the path latency comes from the uplink and downlink access latency (i.e., 2×20 ms), thus the distances covered in the core of the network are only moderately impacting the overall latency. This result would of course change if the network were a worldwide one with large distances or if several network operators were involved in the session.

The influence of the network size and the topology model on the average path latency is shown in Figure 2.6. Similarly to the results shown in Figure 2.4, only the network size in WM model has a significant influence on the latency, because of the linear increase of the distances with respect to the size of the network.

The maximum path latency, which measures the biggest latency between any pair of participants in a session, is shown in Figure 2.7 and Figure 2.8. As explained before, only the network size in WM topology affects the most the latency for all modes. However, The maximum latency of MST and SPT does not exceed 123 ms in MP and ER topologies.

Figure 2.9 shows the impact of the number of participants on the average number of transcoded streams in a call. Since SPT computes the shortest path to the sender, it distributes the streams on a larger number of nodes

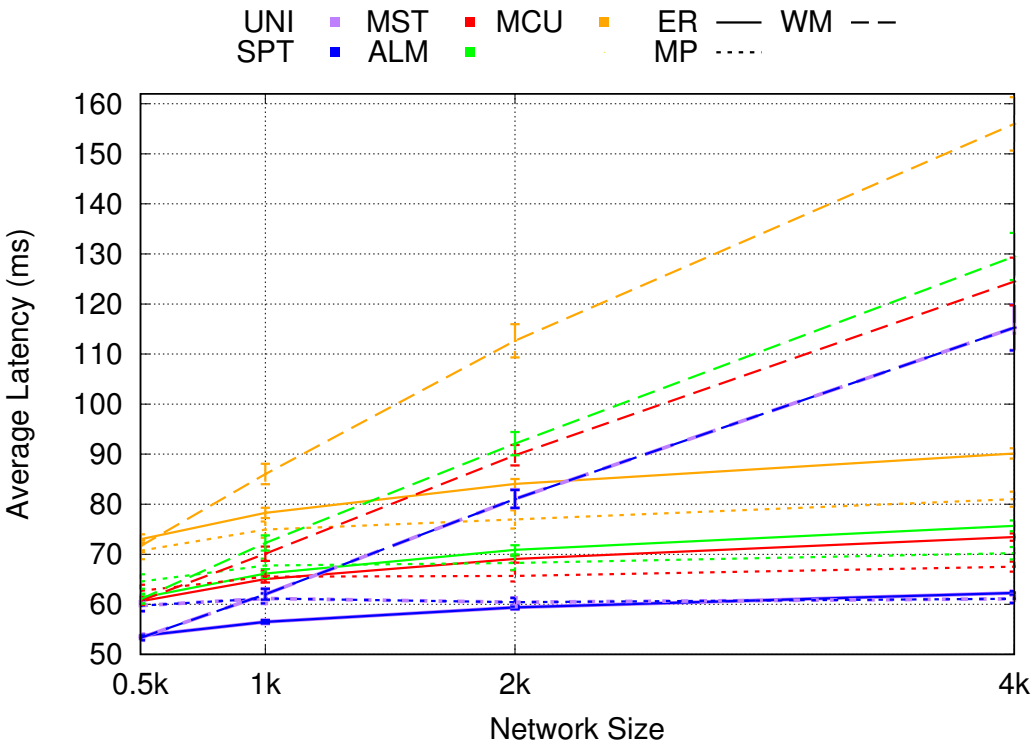


FIGURE 2.6: Average path latency *vs* modes used, topology models and network size, in a call of 6 participants

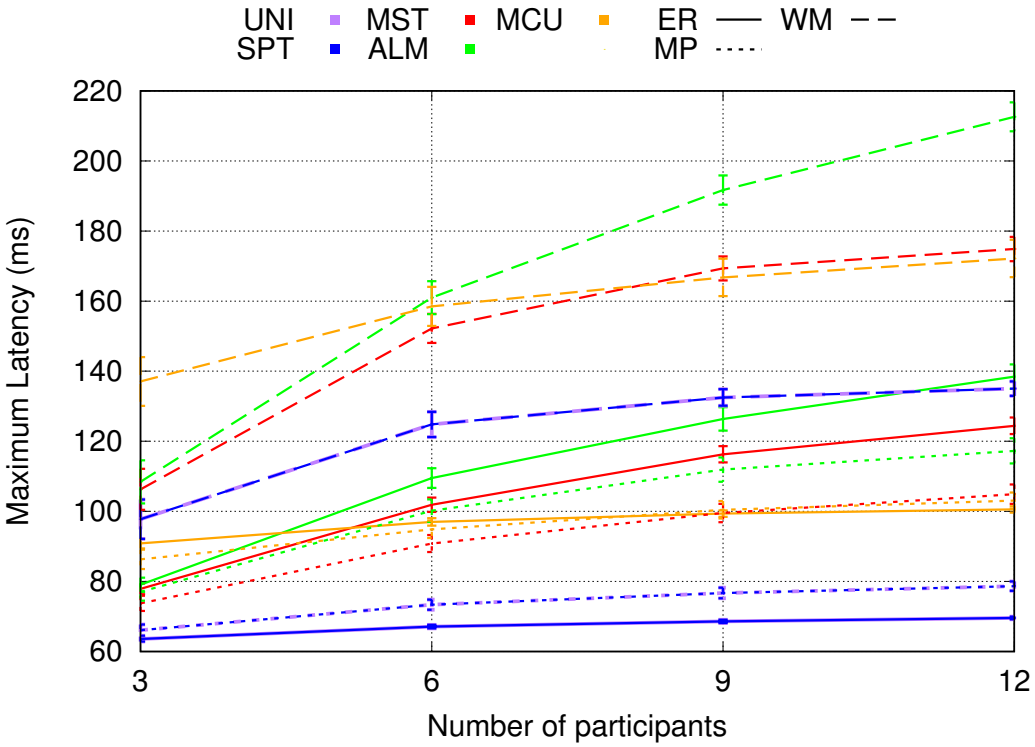


FIGURE 2.7: Maximum path latency *vs* modes used, topology models and number of participants, in a 2k-nodes network

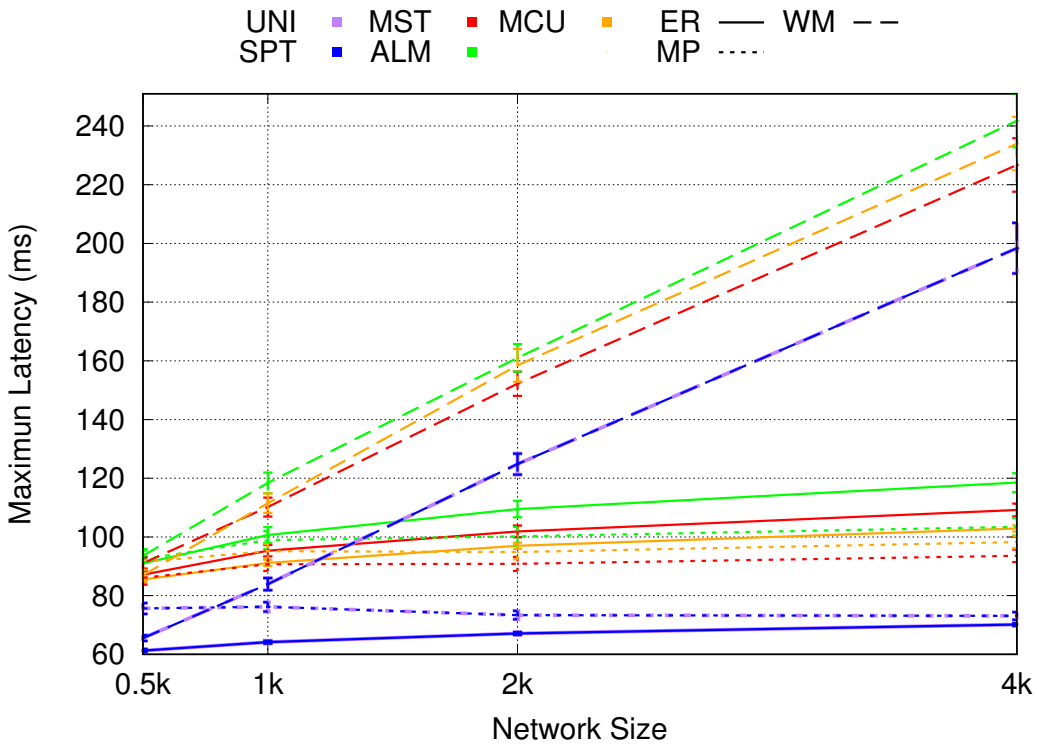


FIGURE 2.8: Maximum path latency *vs* modes used, topology models and network size, in a call of 6 participants

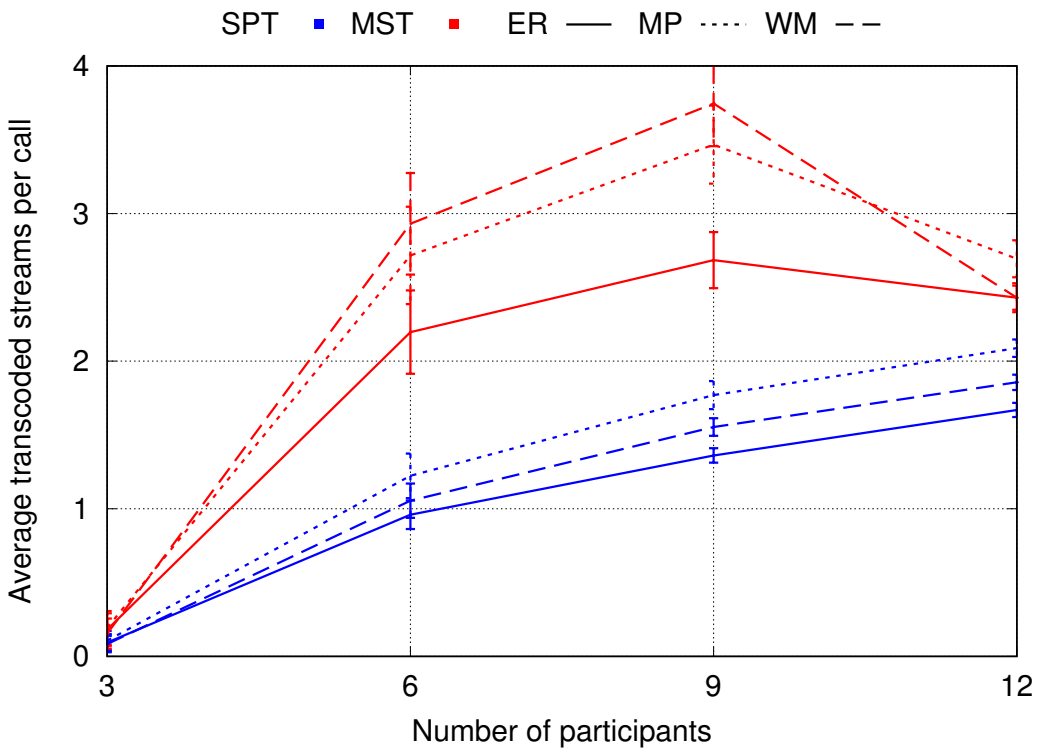


FIGURE 2.9: Average number of transcoded streams per call *vs* modes used, topology models and number of participants, in a 2k-nodes network

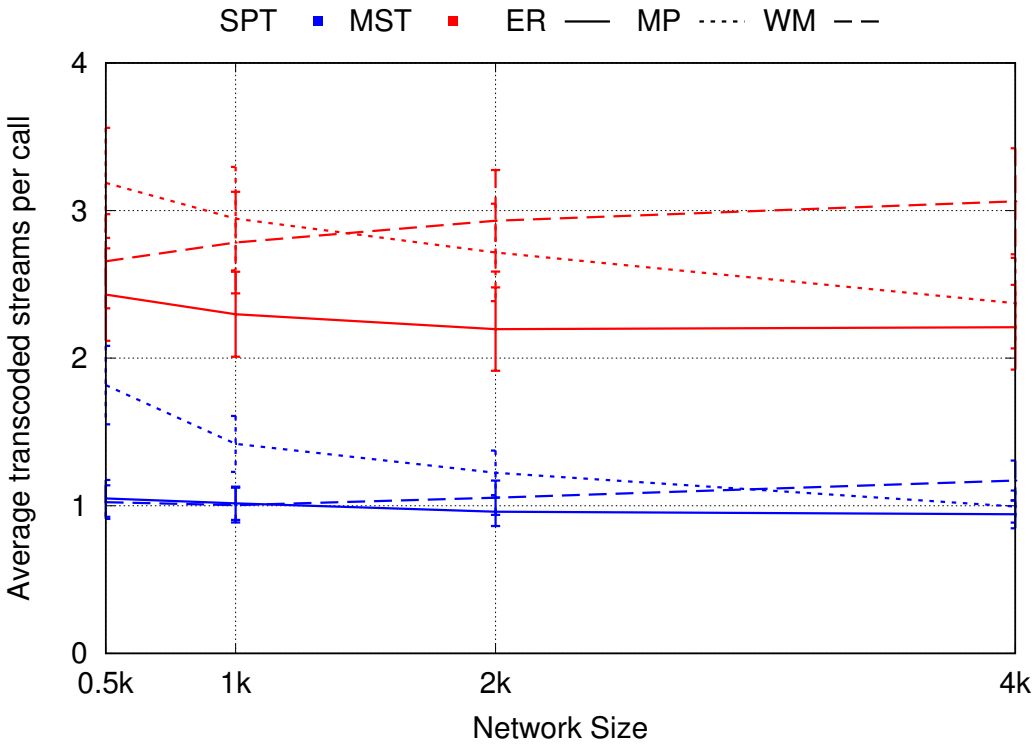


FIGURE 2.10: Average number of transcoded streams per call *vs* modes used, topology models and network size, in a call of 6 participants

(switches) which results in less transcoded streams in a call. On the contrary, MST mode uses the existing branches of the tree to create the paths, which leads to a higher concentration of streams on some nodes, hence having more transcoded streams in a call. When the number of participants is 12, the MST tree becomes larger and the streams are distributed on a higher number of nodes which leads to a drop of the number of transcoded streams.

Figure 2.10 shows the impact of the network size on the average number of transcoded streams in a call. When the network size increases, more edges are created which allows a larger distribution for the streams in the ER and MP networks. However, in the WM topology the probability of the edges between distant nodes is weak (it decreases with the distance), thus, when the network size increases the paths will have to pass by the same nodes which increases the number of transcoded streams.

Figure 2.11 shows the maximum number of transcoded streams per box (counting the highest number of transcoded streams through the box) *vs* the number of participants and the topology models. As described in Figure 2.9, this value increases with the number of participants whatever the mode used in MP topology. However, at 12 participants, the maximum number of transcoded streams per box reaches 20 for MST. As the MST mode aggregates more paths, we see in the figure that it has more streams per box than the SPT mode. Figure 2.12 shows the impact of network size on the maximum number of transcoded streams per box. As described in Figure 2.10, since the MST

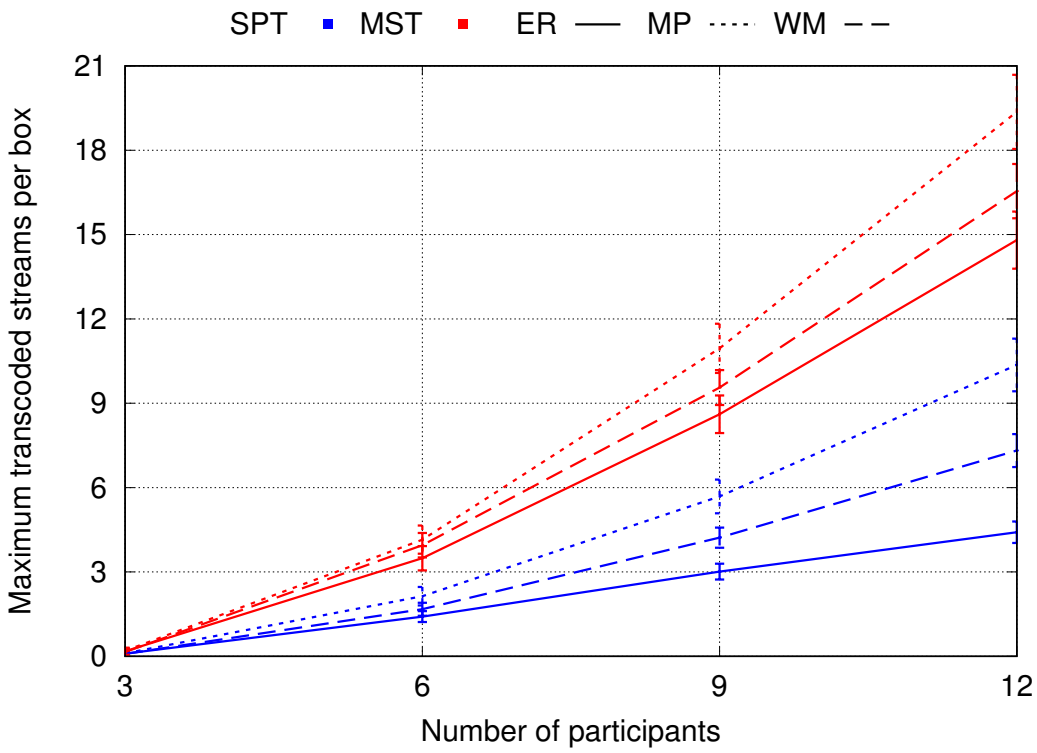


FIGURE 2.11: Maximum transcoded stream per box *vs* modes used, topology models and number of participants, in a 2k-nodes network

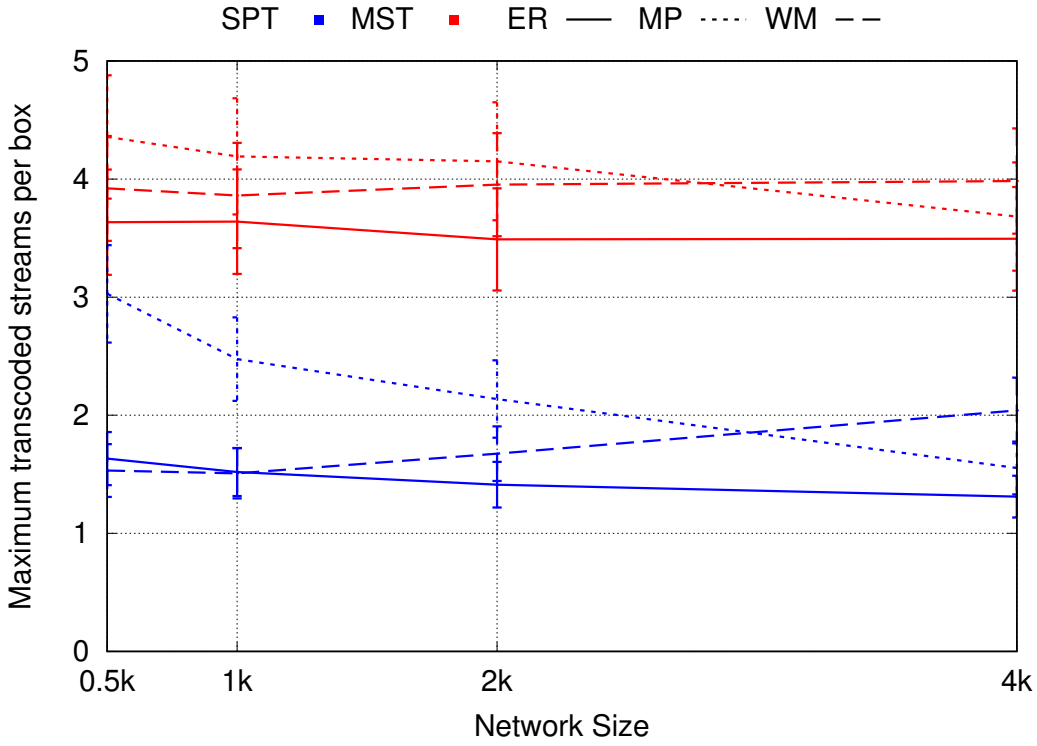


FIGURE 2.12: Maximum transcoded stream per box *vs* modes used, topology models and network size, in a call of 6 participants

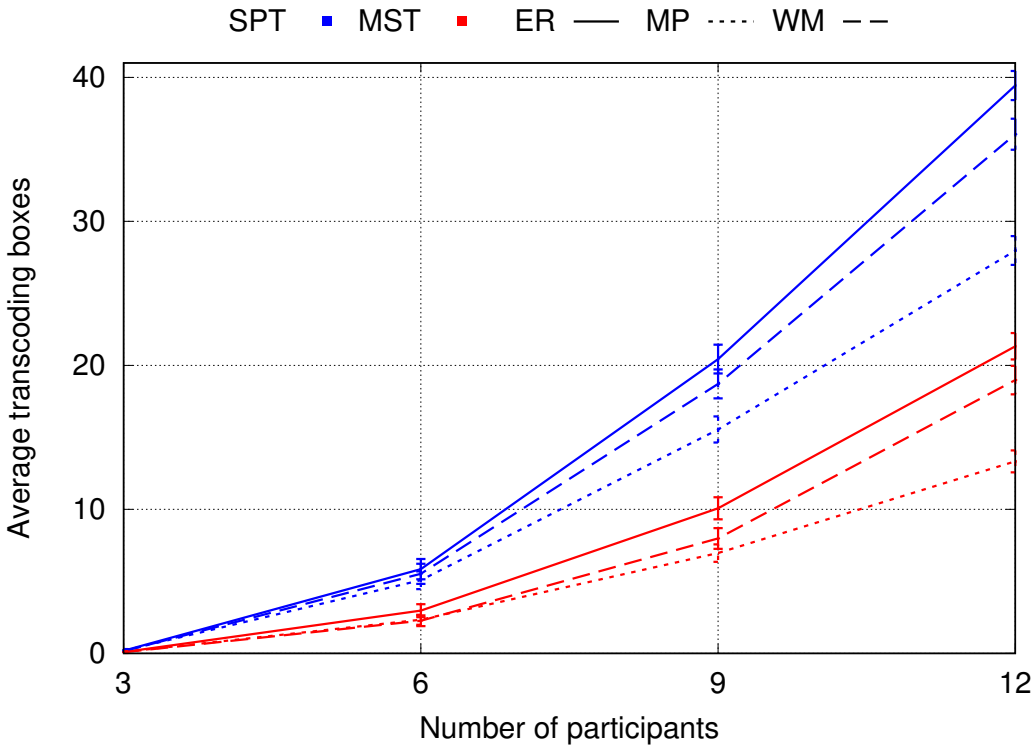


FIGURE 2.13: Average transcoding boxes *vs* modes used, topology models and number of participants, in a 2k-nodes network

mode shows more transcoding per call and it concentrates the streams on a small tree, the maximum number of transcoding streams per box is higher than the one in SPT mode. It should be noted that we do not yet understand very clearly the evolution of the number of transcoded streams according to the size and the type of network. We will study this behavior in the future work.

The influence of the number of participants on the average number of boxes is shown in Figure 2.13. The average number of boxes (counting boxes transcoding at least 1 stream) logically increases with the number of participants whatever the mode used. There is no transcoding at 3 participants as the downlink bandwidth is sufficient to handle all 32kbps streams. However, as explained before, SPT mode contains more transcoding boxes than the MST mode as the MST mode aggregates more paths in a small area.

2.6 Conclusion

In this chapter, we have defined an MVoIP model leveraging the SDN paradigm in order to provide adaptive bitrates to each participant. The combined use of multicast distribution and stream transcoding placement enables important bandwidth savings. To create the multicast tree, we used two modes: MST and SPT. MST mode consists of creating tree by connecting every new node the closest existing node in the tree. On the other hand,

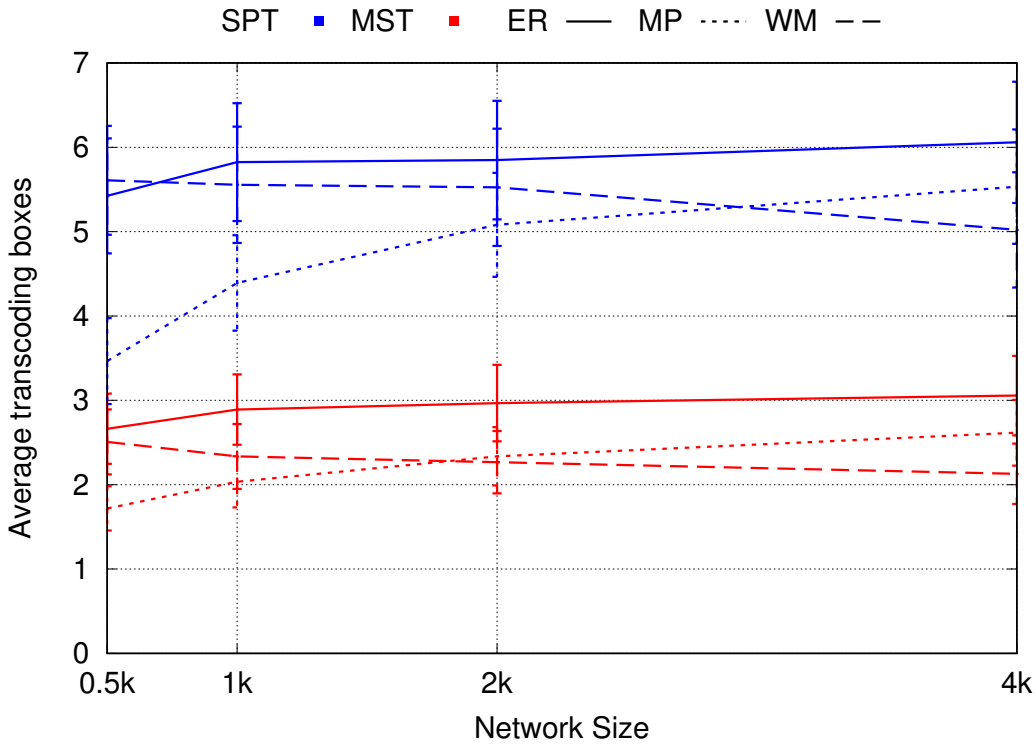


FIGURE 2.14: Average transcoding boxes *vs* modes used, topology models and network size, in a call of 6 participants

SPT mode creates trees by connecting the new node with the shortest path directly to the root.

Our results have shown that the MVoIP session based on the MST mode has the lowest bandwidth consumption comparing to existing conference modes (ALM, MCU). However, it shows higher average path latency comparing to SPT but with acceptable values remaining very low. The results also show that the number of boxes required to transcode streams is roughly halved when using the MST mode over the SPT one. Our designed model is based on some assumptions such as the infinite capacity of the network as well as the stability of the access link capacities during the session. In the following chapters, we adapt our model to more realistic networks with limited bandwidth capacities and dynamic access capacities. We will analyze our model over the duration of a session, evaluate its functionality with video conferences, and maximize the use of the total network capacity by optimizing the construction of the multicast trees over many simultaneous sessions.

Chapter 3

Bandwidth Optimization of Video Conference Calls

3.1 Introduction

For the past decade, the usage of video conference applications has considerably grown. One of the main criteria when judging a video conference call is by the quality of its video streams. These streams are usually very sensitive to the network state and have stringent QoS requirements, such as low end-to-end delay, low packet loss, high bandwidth, etc. In addition, they are also affected by the heterogeneity of the users' wireless link characteristics, which may degrade the quality of the video streams for some or all the users.

Traditional networks have difficulties to manage all those QoS requirements in real time. Earlier research has focused on developing new video codecs, such as SVC (Schwarz *et al.*, 2007), that separates a video stream into several layers of different quality levels. Thus, these codecs are able to adjust the bitrate of the video stream more easily, simply by dropping unnecessary layers. This allows a dynamic adaptation of the video stream to the characteristics of the end-user access link. While SVC permits a better control over the stream bitrate, it still only drops the unnecessary layers at the endpoints or in a MCU and is still not widely applied in practical video transmission. Since SVC has no knowledge about the network, the layers cannot be adapted inside the core network (Lahbabi and Hammouch, 2014).

The development of SDN has enabled new possibilities for the efficient control of video conferences. This architecture provides a separation between the control plane and the data plane of a network. It consists in a logical centralized view that allows the management of the network in a more efficient way.

In order to adapt the video streams inside the network and minimize the core bandwidth consumption, without the need of a specific MCU server to manage the control, we build a model leveraging the SDN architecture similar to the model described in Section 2.1.2 and give the switches SVC functionalities. Then, we use Algorithm 1 for building multicast distribution trees and dropping video layers at optimal locations (on SVC switches).

Therefore, the SDN controller is able to define where inside the network, some video layers should be dropped. Thanks to the video layer dropping, we are able to adapt to the bandwidth capacities of the receiving devices.

3.2 Video conference model

Like in MVoIP, a participant in a video conference is, at the same time, a sender of its video stream and a receiver of the video streams from all other participants. Each participant is placed randomly in the network by being connected to any node of the network. Participants are considered to be connected wirelessly to their node by a 4G cellular technology such as LTE. Each node of the network is supposed to provide a wireless access function (such as an eNode-B) and an SDN switching function. Similar to the model for MVoIP, as the network is SDN-enabled, we assume that all nodes within the network contain OpenFlow switches and can communicate with an SDN controller which is responsible for the management of the calls. All switches are supposed to be able to adapt SVC streams by dropping unused layers on the paths indicated by the controller. We assume that participants have appropriate devices for conferencing (e.g., large tablets, laptops, mobile A/V systems).

SVC streams are structured in layers, all built upon a base layer. We assume the SVC layers consisting of a base layer L1, and three enhanced layers L2, L3, and L4. However, our algorithms can be easily generalized to any number of layers. In addition, a given layer can be used only if all the lower layers are also received. Each participant accepts the highest number of layers allowable by its downlink capacity. With only the base layer, the participant will receive the lowest video quality. If the participant's downlink capacity falls below what is required to receive the base layer, it can still remain in the conference if it can at least receive an audio-only stream. If any participant can not receive the audio stream, the call is rejected.

In a video conference, the highest possible bitrate permitted by the access link bandwidth is recommended in order to achieve a better QoE. Therefore, during a video call, each participant sends the maximum bitrate stream acceptable among the receivers, and this stream is then degraded (i.e., higher layers are dropped) to adapt to the participants with lower downlink bandwidth capacity. After discovering the network topology and channel conditions using SDN global view, it is significantly easier to identify the locations of the switches where it is necessary to degrade SVC streams.

3.2.1 Technical assumptions

The algorithm, executed in the SDN controller, relies on several assumptions:

- All SVC layers belonging to the video stream emitted by one sender, follow the same paths. There is one tree for all layers of a given video stream. Since the adaptation are performed in the core network, some branches of that tree may carry only a subset of the layers.
- The SDN controller knows the complete topology of the network, the position of the participants as well as their available uplink and downlink access bandwidth.
- Any SDN switch can degrade (i.e., drop higher quality layers) an SVC stream. The codec bitrate (CB) is equal to the SVC levels $\{L1, L2, L3, L4\}$ and the audio layer. The audio layer enables receivers with very low access link bandwidth to participate in the call.

3.2.2 The algorithm

To build the video conferencing sessions, we use Algorithm 1 explained in Chapitre 3. The same two modes (SPT and MST) are implemented for the tree construction since both modes performed, for audio conferences, good bandwidth savings while providing to each participant the maximum quality that it can receive. Recall that SPT mode computes the shortest path tree between each sender and the other participants (the receivers), and MST mode proceeds iteratively. MST first computes the shortest path between the sender and the first receiver (according to a specific ranking rule), then the shortest path between any node of the previous path and the second receiver. At each step, it computes the shortest path between the current multicast tree and the next receiver.

3.3 Simulations of setting up a video call

In this section, we evaluate the efficiency of Algorithm 1 through extended simulations. We compare it to the main state of the art algorithms: unicast, MCU and ALM. More precisely, we study the performance of the different modes in terms of bandwidth savings, maximum delay and network call capacity.

3.3.1 Simulations methodology and parameters

We evaluate the different algorithms on random topologies according to the two models: the Erdös-Rényi model (ER), as well as the Magoni-Pansiot model (MP). We haven't taken into consideration, in this chapter, Waxman model (WM) since it is not realistic as seen in Chapter 2.

We have implemented Algorithm 1 including its two different modes (MST, STP) and have compared them to the unicast, MCU and ALM algorithms. To build our simulation model, we consider the following parameters:

- For p participants, p multicast trees will be built. Noting that many participants can be connected to the same access node.
- We assume the SDN network to be WAN-sized. The network topology sizes evaluated are 500, 1000, 2000 and 4000 nodes.
- The access downlink bandwidths are chosen from a range of plausible 4G data rates for each participant. The access downlink capacity for each participant is set to a value in the range from 4 Mbps to 14Mbps and its uplink bandwidth is set to 1.5Mbps.
- The core links are supposed to all have the same bandwidth dedicated for this type of A/V traffic. This means that links may have different bandwidth capacities but they all reserve the same amount of bandwidth for the video conferencing calls. The core link bandwidth is set to 1Gbps.
- The maximum latency authorized over any path is set to 250ms.
- Four SVC profiles and one fall back audio-only profile are used in the following simulations. The indicated bitrates include network headers' overhead and audio streams (for SVC profiles):
  - Audio-only, 32kbps.
  - Layer 1: Scalable Constrained Baseline, Level 1, 90kbps.
  - Layer 2: Scalable Baseline, Level 1.1, 250kbps.
  - Layer 3: Scalable Constrained High, Level 1.2, 0.5Mbps.
  - Layer 4: Scalable High, Level 1.3, 1Mbps.

  The audio-only profile enables receivers with very low access link bandwidth to participate to the call.

The difference with building audio/video sessions occurs in the parameters. Table 3.1 shows the simulation's parameters for setting up the video conference. We mark in bold the distinct parameters of those of the MVoIP simulations.

To obtain our results, we first start by creating 20 topologies for each topology type (ER and MP) and for each network size (500, 1000, 2000 and 4000), using a different instance every time. Then, for each one of these topologies, we connect randomly the client with different access capacities, then we repeat this step 20 times. Thus, each point on the following plots is the average of the values obtained from 400 simulations. This average is presented accurately using a confidence interval of 95% displayed on the plots.

TABLE 3.1: Simulation Parameters

Parameter	Values
Modes	Unicast, SPT, MST, ALM, MCU
Audio Bitrate levels	32 kbps
Video Bitrate levels	**SVC layers (BL, L1, L2, L3, L4)**
Max number of transcoding per path	**unlimited**
Placement of participants	Random
Access uplink bandwidth	**1.5 Mbps**
Access downlink bandwidth	**[4, 14] Mbps**
Access downlink bandwidth distribution	uniform
Access link delay	20 ms
Core link bandwidth	**1 Gbps**
Core link delay	5 ms
Number of runs	400

Figures 3.1, 3.2 show consecutively the impact of the network size and the number of participants on the average bandwidth usage obtained for MP topologies.

Figure 3.1 shows the influence of the network size (from 500 to 4000 nodes) on the bandwidth usage of a call with 6 participants. In Figure 3.1 every solution exhibits a flat relationship between the network size and the bandwidth consumption. Our MST and STP solutions are better in saving bandwidth than the other solutions. As expected, the MST mode is the least bandwidth consuming, closely followed by the SPT mode, due to MST nature that finds the shortest path to the closest node existing in the tree, unlike SPT which creates the shortest path up to the tree root. ALM mode indicates higher bandwidth consumption since it creates the tree by connecting each new participant to the closest participant in the tree. The MCU mode, on the other hand, consumes the highest bandwidth usage since the stream needs to attend the MCU before getting to the receivers. Similarly, Unicast mode has high bandwidth consumption due to the redundancy of the streams on the shared links. With 4k network size, MST mode roughly consumes 56.7% of the bandwidth used by the MCU mode and 60% of the one used by unicast mode. The ratio drops to 26% for ALM. We notice that the network size affects the average bandwidth usage less than the number of participants does.

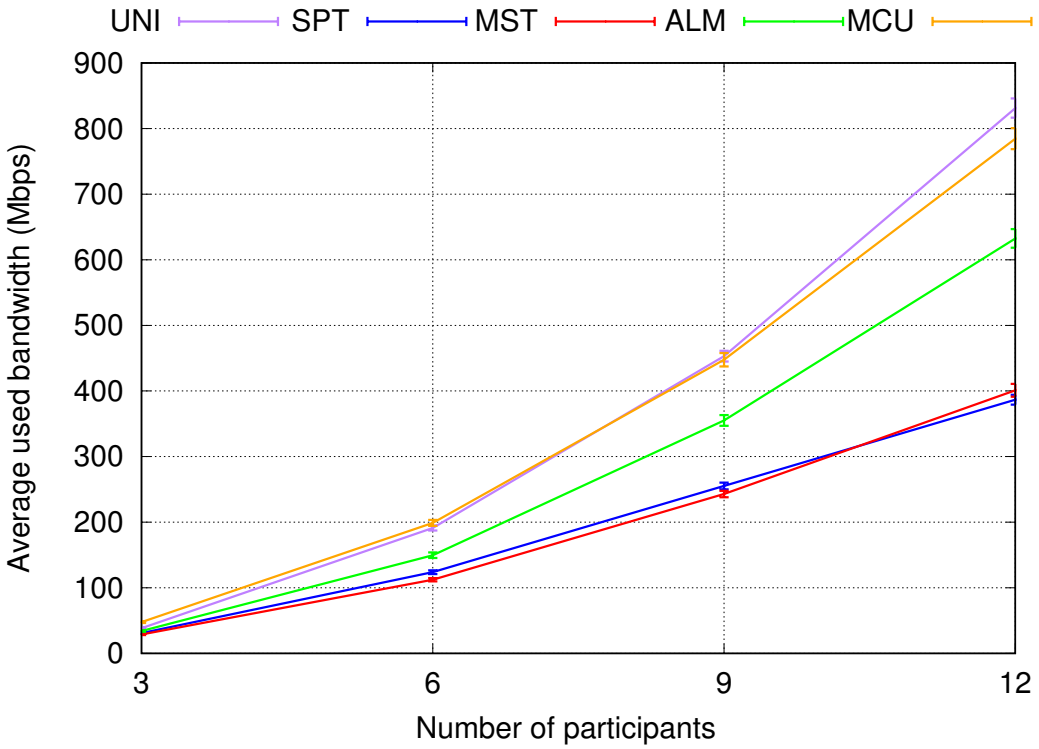


FIGURE 3.2: Average bandwidth usage *vs* number of participants on MP topology with a network size of 2k nodes

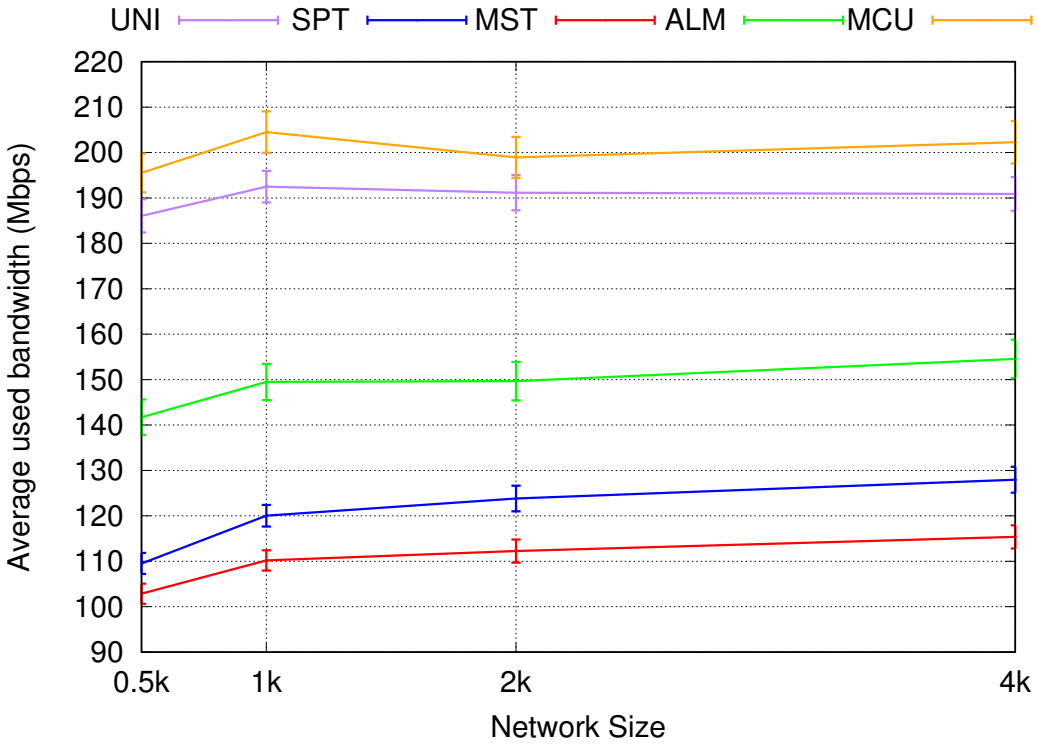


FIGURE 3.1: Average bandwidth usage *vs* network size on MP topology with 6 participants per call

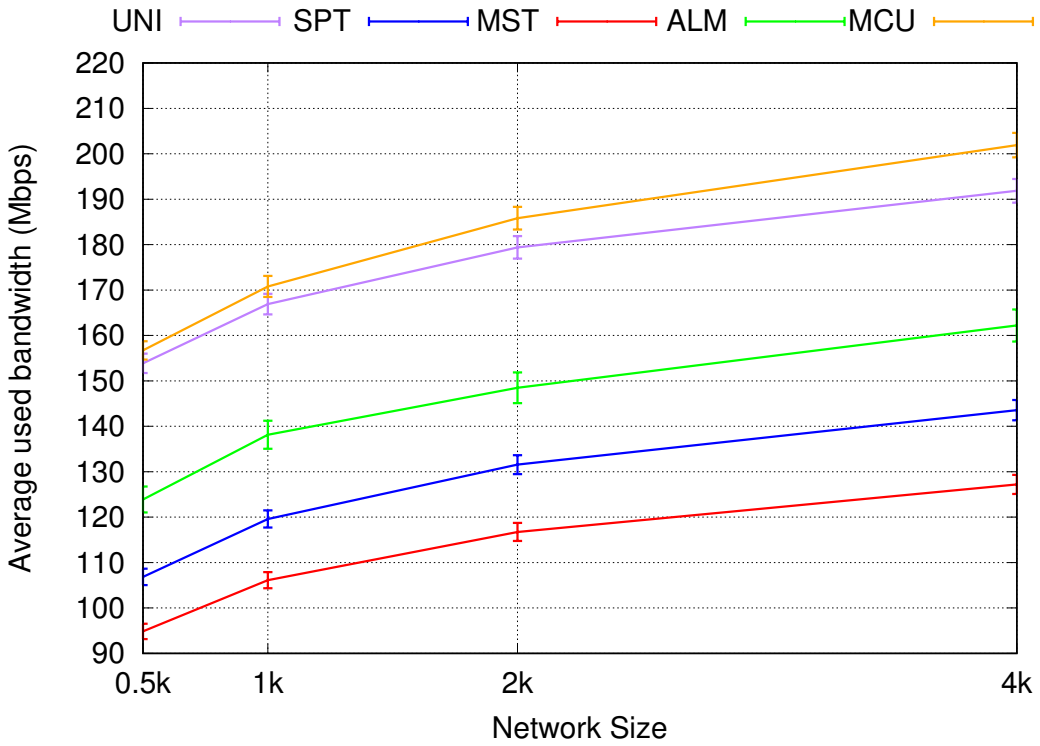


FIGURE 3.3: Average bandwidth usage *vs* network size on ER topology with 6 participants per call

Figure 3.2 shows the influence of the number of participants $(3, 6, 9, 12)$ on the bandwidth usage of a call with a network size of 2000 nodes. Obviously, the bandwidth consumption increases according to the number of participants. With 3 participants, the bandwidth usage of all mode is very close to each other due to the small size of the created trees. As expected and shown in Figure 3.1, MST and SPT modes have the lowest results. On the other hand, with unicast and MCU, the bandwidth usage is more affected by the participants' number.

Figures 3.3, 3.4 show consecutively the impact of the network size and the number of participants on the average bandwidth usage obtained for ER topologies. The results are similar to those obtained for MP topologies. MST and SPT are still the most efficient in terms of bandwidth savings. But we can notice in figure 3.3 that the network size in ER has more effect on the average bandwidth usage than the one in MP. This is due to the sparse nature of MP topologies.

Figures 3.5, 3.6 show consecutively the impact of the network size and the number of participants on the processing time obtained for MP topologies.

On Figure 3.5, we observe the average processing time of the different algorithms per call of 6 participants. MST follows ALM, UNI and STP with very close processing time and has the similar results as they all use shortest paths to a specific point in the tree.

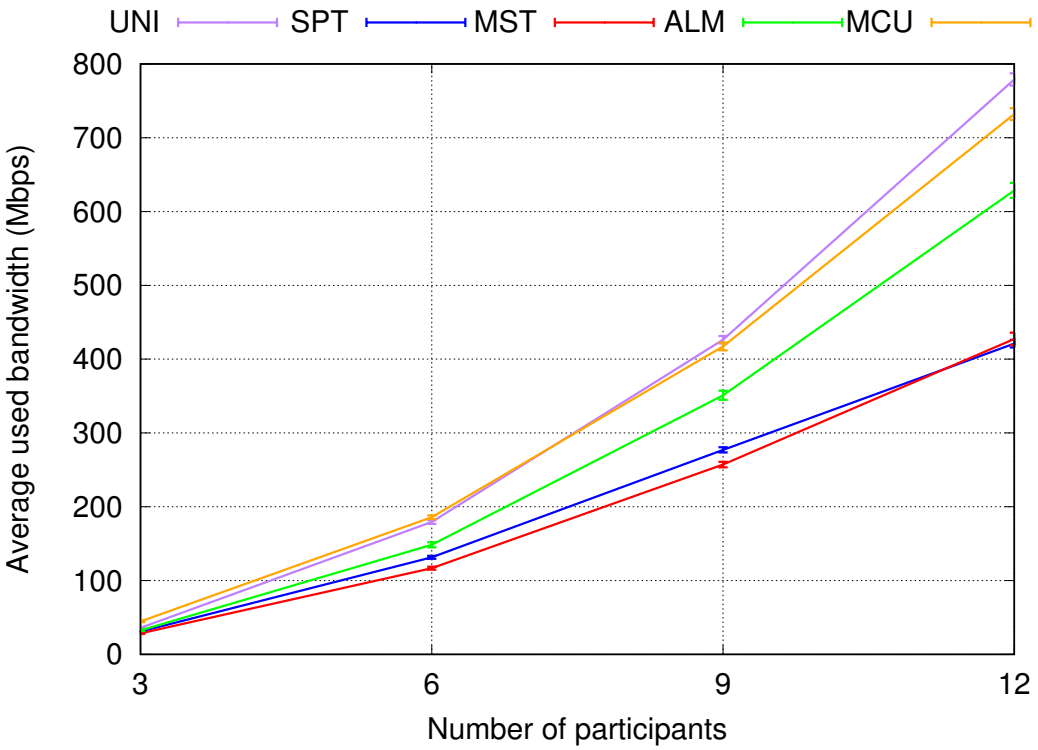


FIGURE 3.4: Average bandwidth usage *vs* number of participants on ER topology with a network size of 2k nodes

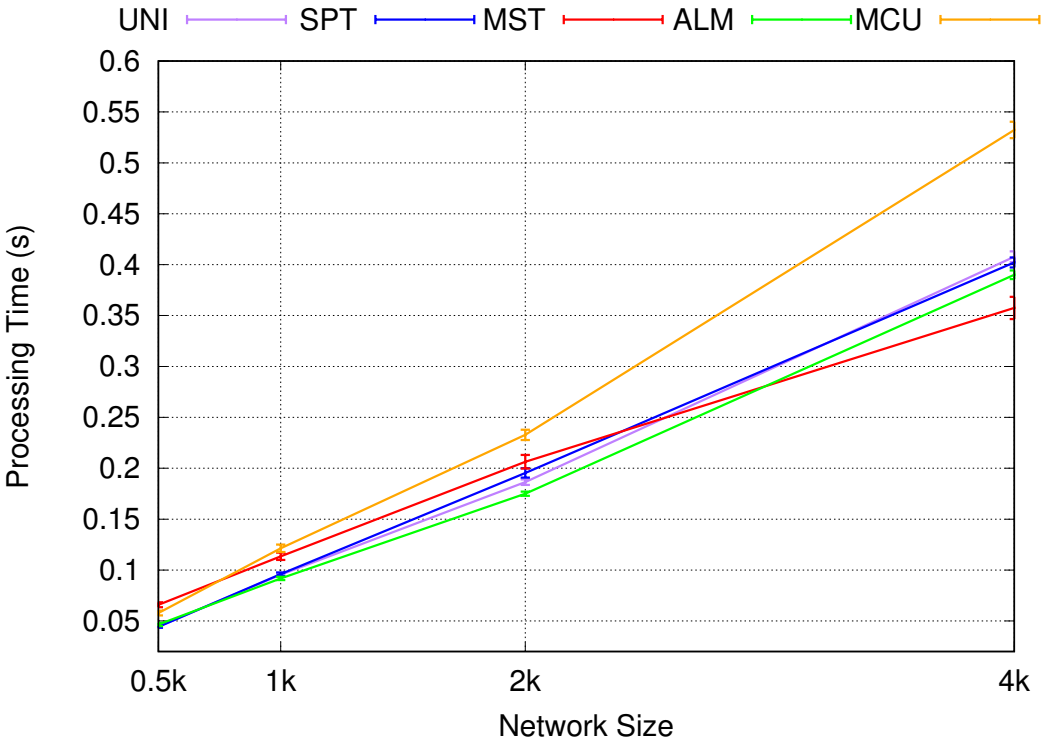


FIGURE 3.5: Average processing time *vs* network size on MP topology with 6 participants per call

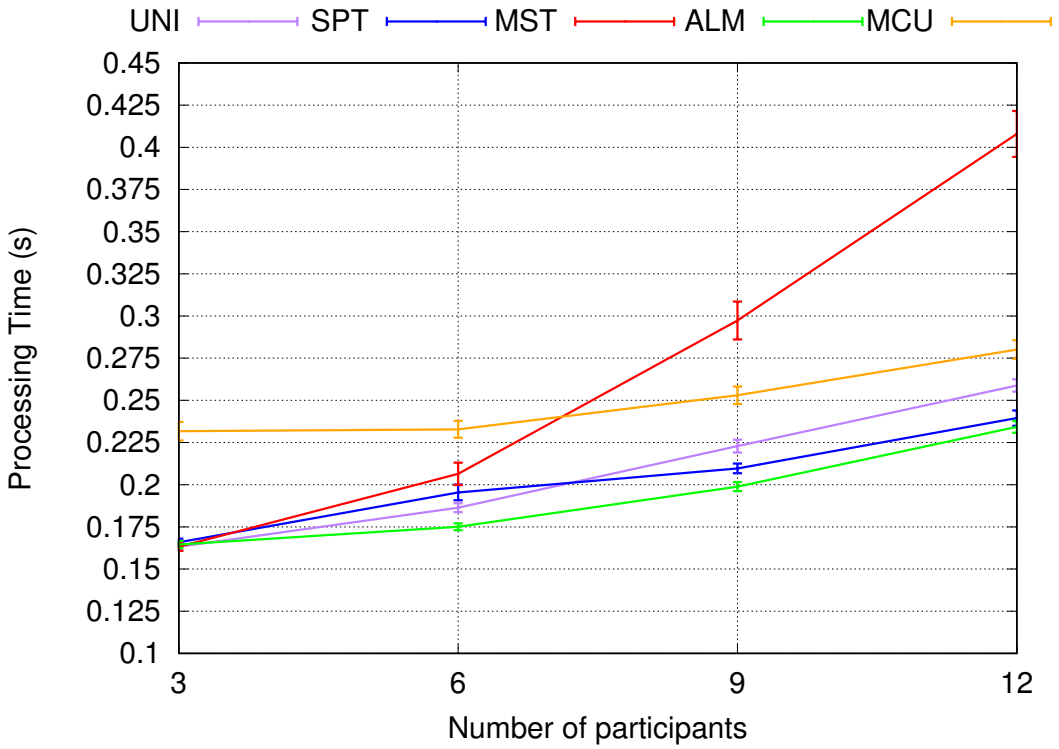


FIGURE 3.6: Average processing time *vs* number of participants on MP topology with a network size of 2k nodes

The impact of the number of participants on the processing time is shown in 3.6. The processing time of MST grows much faster than the others. This is due to its complexity as explained in Section 2.3.1. However, it remains very applicable since it requires only 0.4s to compute paths for establishing a video call with 12 participants.

Figures 3.7 and 3.8 show the impact of the network size and the number of participants respectively, on the processing time for ER topologies. Both figures show the same results as in Figures 3.5 and 3.6. The processing time with MCU mode is still the most affected by the network size and exhibit the highest values. On the other hand, the processing time of MST mode is the most affected by the number of participants.

Figures 3.9, 3.10 show consecutively the impact of the network size and number of participants, on the maximum latency obtained for MP topologies.

Figure 3.9 shows the influence of the network size on the maximum latency of a call. The maximum latency of a call is defined as the maximum latency measured over all paths between all participant pairs. It takes into account access and core link delays. We observe that the network size does not have a big influence on the latency since MP topologies have a small diameter. As expected, ALM shows the highest latency: it reaches 125 ms ($\simeq$17$\times$core link delays of 5 ms + 2$\times$access link delay of 20 ms) with 4k network node. MCU has also a high latency (120 ms with 4k network node) due

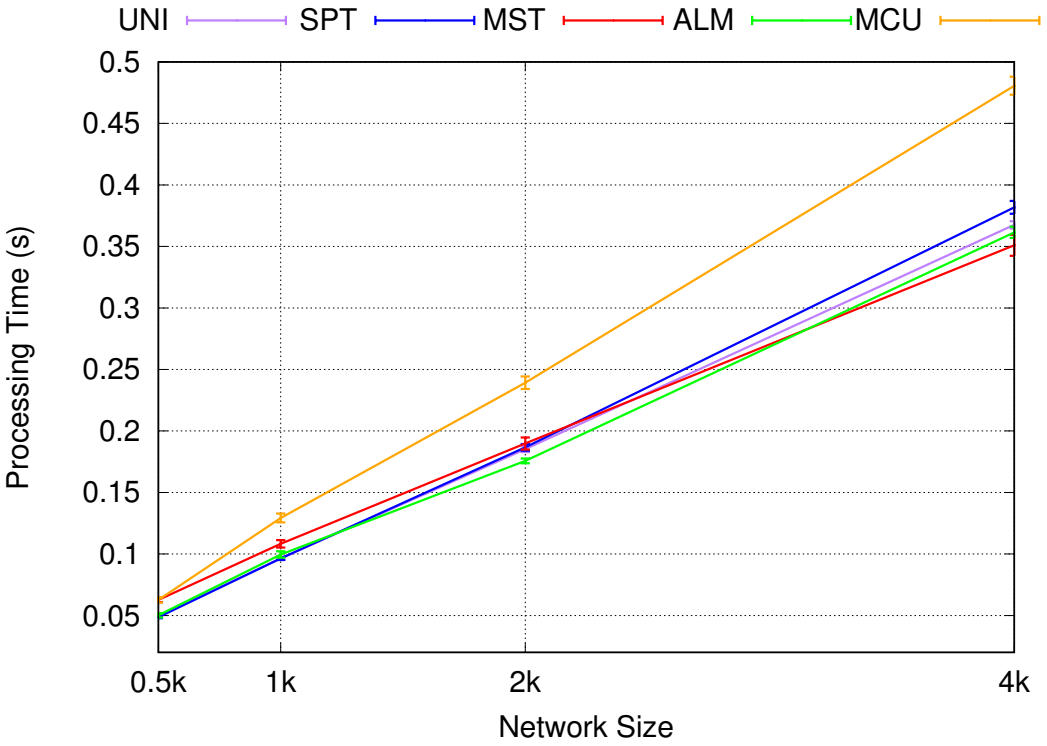


FIGURE 3.7: Average processing time *vs* network size on ER topology with 6 participants per call

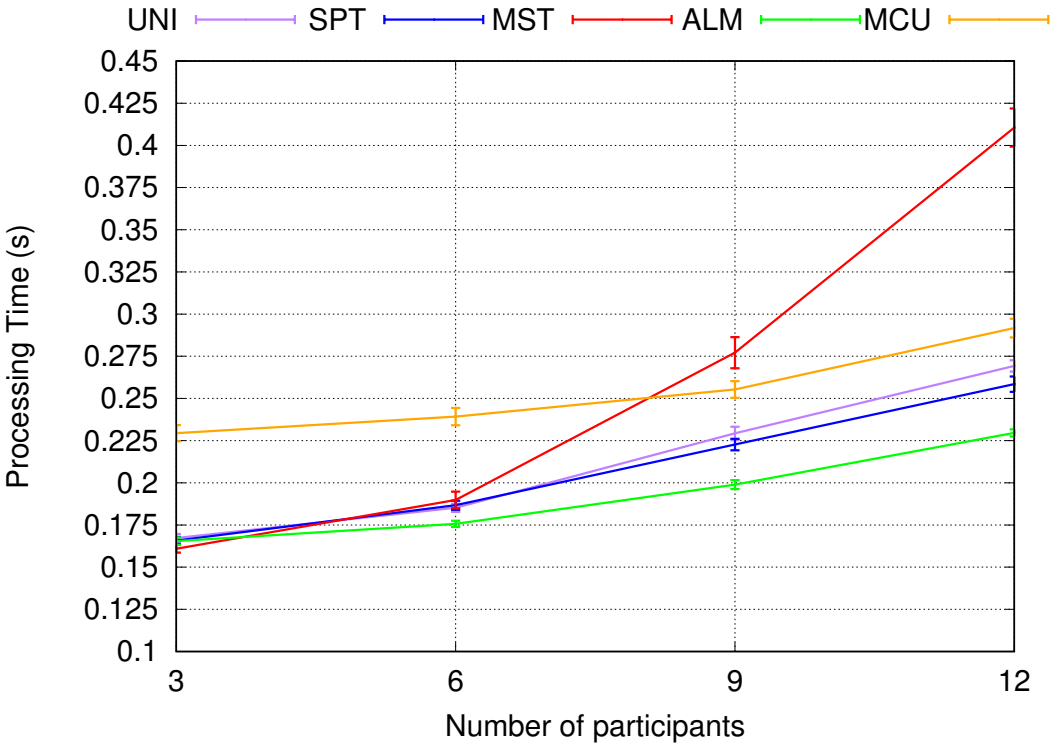


FIGURE 3.8: Average processing time *vs* number of participants on ER topology with a network size of 2k nodes

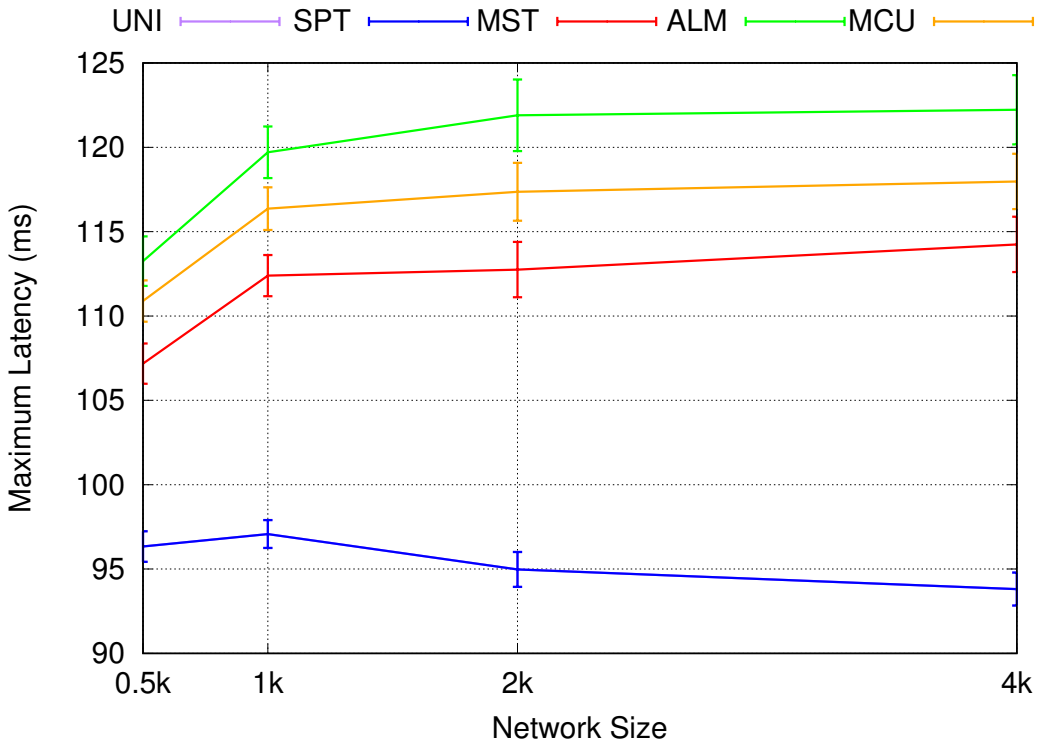


FIGURE 3.9: Maximum latency *vs* network size on MP topology with 6 participants per call

to the fact that all the streams need to reach the MCU before reaching the destination. MST mode shows a maximum latency of 115 ms since MST, unlike SPT, is designed primarily to save bandwidth. Latency is partially considered when finding the shortest path between a new node and the existing distribution tree. The latency is exactly the same for the SPT and unicast modes, as expected, and decreases with the increase of the network size. This is also expected as MP topologies are so-called "power-law" graphs where the average distance and the diameter do not increase much when the network size increases.

Figure 3.10 shows the influence of the number of participants on the maximum latency of a call. The number of participants affects mostly ALM mode since it based on finding the closest participant to any other participant to create the tree. MST mode exhibits a higher latency than MCU with more than 9 participants. However, MST latency values remain acceptable (<200 ms) on MP. The latency in MCU, SPT and unicast is barely affected by the number of participants as they create the shortest paths to a specific point (either the source or the MCU). The SPT and unicast modes yield the same and lowest results regardless of the number of participants, which is expected as they both use shortest paths to the source of the tree, unlike MCU mode which uses the shortest path to the MCU node.

Figures 3.11 and 3.12 show a similar behavior on both ER and MP topologies. However, the latency on ER is higher than the one on MP and it becomes high with MST or ALM mode when the number of participants is 12.

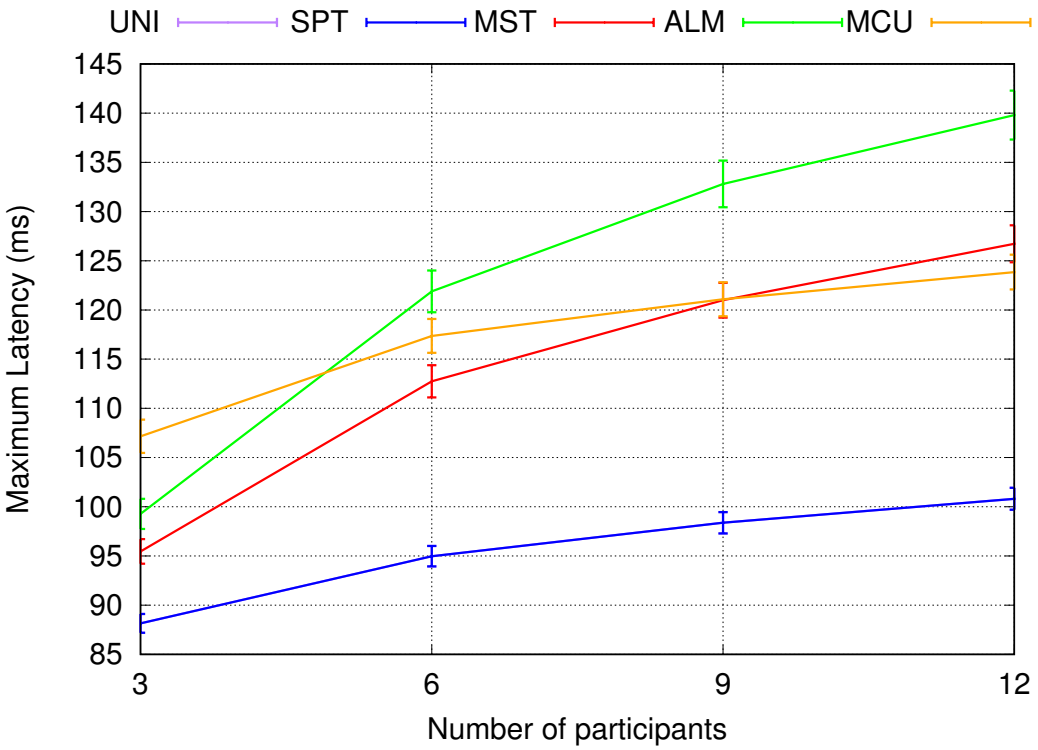


FIGURE 3.10: Maximum latency *vs* number of participants on MP topology with a network size of 2k nodes

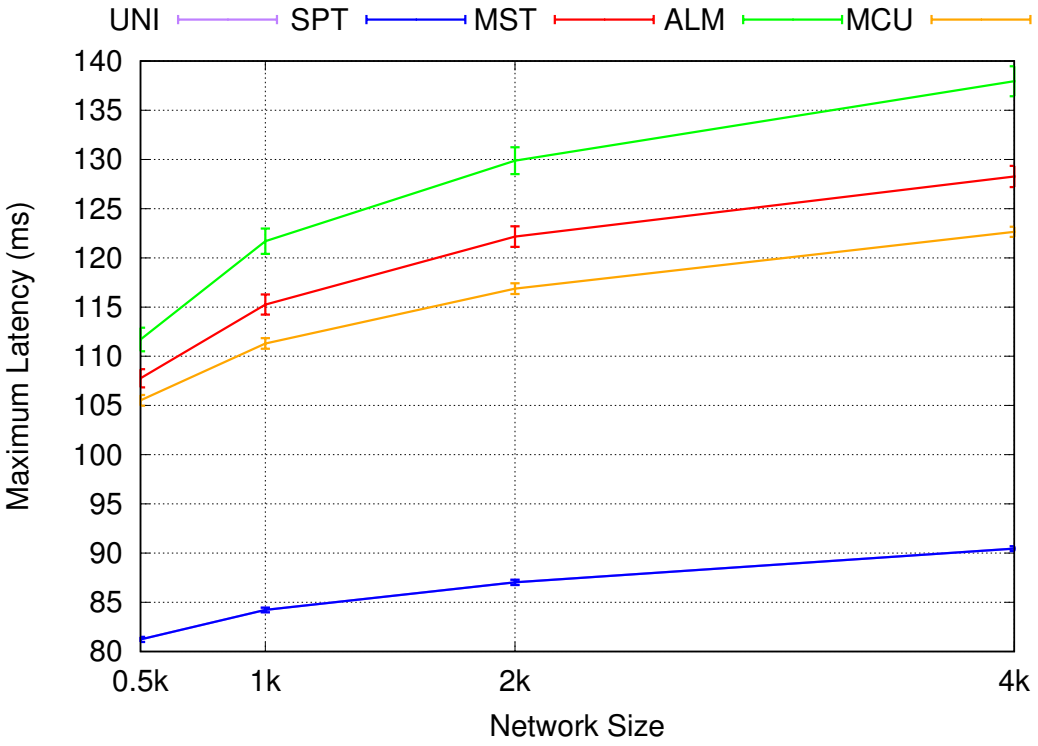


FIGURE 3.11: Maximum latency *vs* network size on ER topology with 6 participants per call

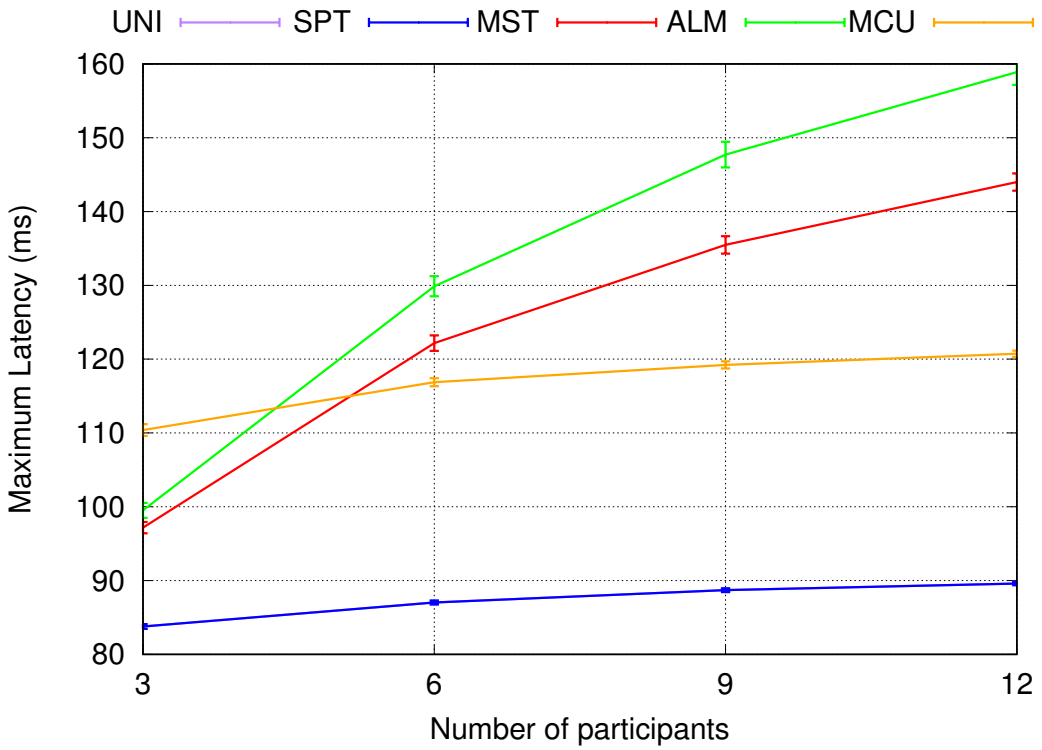


FIGURE 3.12: Maximum latency *vs* number of participants on ER topology with a network size of 2k nodes

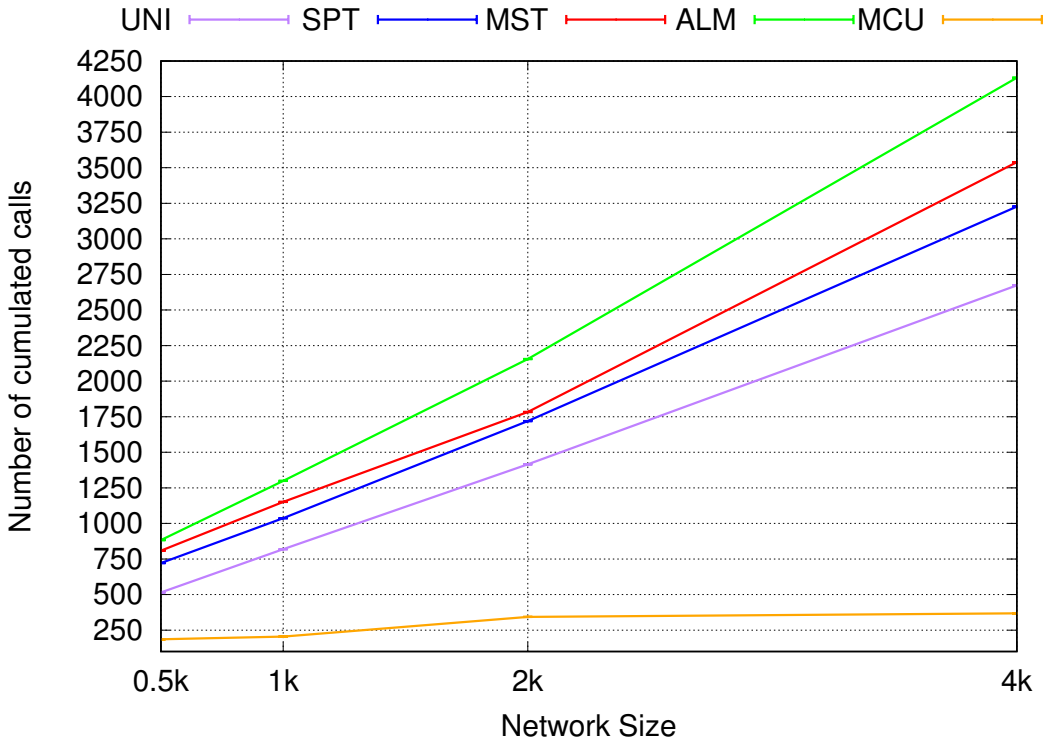


FIGURE 3.13: Number of calls supported by the network usage depending on the network size on MP.

Figure 3.13 shows the impact of the network size on the network call capacity. The network call capacity is determined as follows: the call arrival model is a Poisson distribution with a mean arrival time depending on the expected number of simultaneous calls in a network with specific core link bandwidth capacity. We assume that the call duration model follows an exponential distribution with an average of 23 minutes and the call arrival model is a Poisson distribution. At some point, the *BuildShortestPath*() function defined in Algorithm 1 will return false, indicating that the call could not be set up because one or more links involved in the call were saturated (i.e., filled at the maximum of their bandwidth capacity). In this case, the call is rejected. After 100 rejected calls, the network call capacity is considered reached. The number of supported calls depends on the sequence of construction of all the calls which are generated according to our model. Therefore, for each experiment, the network capacity will slightly vary. We have performed 400 experiments on MP networks with core link bandwidth set to 1Gbps. In Figure 3.13, for all the algorithms except MCU, the call capacity quickly increases with the network size. ALM mode exhibits the best results with the highest number of supported calls (4200 with 4k node), as the trees in ALM are not optimized, and the paths are more distributed among the topology which lead to a slower saturation of the network, thus more supported calls. MST mode shows good results as it can support up to 3500 simultaneous calls, followed by SPT which can support up to 3250 simultaneous calls. On the other hand, MCU does not seem to be affected by the network size. That is due to the fact that all the calls need to pass through the MCU which may saturate some links and quickly cause too many call rejections.

3.4 Conclusion

Video conference applications have strong bandwidth and latency requirements and consume large portions of a network's bandwidth. Current video conferencing solutions are not efficient as they often rely on a central server and do not leverage in-network video layering capabilities. In this chapter, we investigated the impact of SVC and SDN techniques on video conferences. Specifically, we used the algorithm proposed in the last chapter to reduce the bandwidth consumed by video conferences. Using SDN for computing and deploying multicast trees, we took advantage of the SVC layering feature to allow video layer dropping for optimizing quality.

We showed that by smartly dropping video layers at specific locations in the network, the overall bandwidth usage by video conference calls decreases. Our solutions, MST and SPT modes, save a lot of bandwidth comparing to the other modes (ALM and MCU). MST shows better performance than SPT for bandwidth usage on both ER and MP topologies. In term of latency, both SPT and MST modes have good performance and do not exceed the maximum allowed latency on MP topology. However, with ER topology,

MST exceeds the latency limits when the number of participants is too high. Regarding the network capacity, we have shown that our solutions allow networks to support a higher number of video conference calls compared to existing solutions, such as MCU. With ALM mode, the network can support the highest number of simultaneous calls, followed by MST and SPT modes. However, since ALM has a very high latency, our solutions SPT and MST seem more suitable than the other algorithms.

Chapter 4

Dynamic Adaptation of Video Conference Calls

Besides usual network problems (delay, congestion, link limitations) and hardware problems (device limitations), video conferences may experience an external load on the endpoints and/or problems in the wireless medium. An external load on the network can cause bandwidth variation on the access links. For example, if someone at work is downloading or streaming a large amount of data during the conference, it will affect the video conferencing resources since it will be shared with the streaming. A long running download introduces queuing delay that may make the conference session less responsive. Many other cases can create interference and reduce the stream quality such as the presence of obstacles or a reflector surrounding the sender or the receiver. In this case, the transmitted signal will be replicated in multiple copies each having a with different amplitude and delay. When these signals reach the destination, they affect the bandwidth and this can result in a disruption or failure in the communication. To evaluate a conference quality, we should measure the adaptation speed to bandwidth variation in real time, not only looking at the capacity of the bandwidth. Even if most conferences can adapt to the participants' capacities, it is still critical to adapt the stream smoothly in real time.

The video conference system of the previous chapter aims to minimize the bandwidth consumption in the core network by accurately computing the multicast trees and opportunely placing the SVC adaptation rules inside the network. However, the algorithm used does not consider a dynamic environment where access bandwidth variation occurs during the call. In this chapter, we address this lack in Algorithm 1 when under random access bandwidth variations. Thus, we are able to evaluate how this algorithm behaves in a dynamic environment and how we can create effective derived algorithms able to face network dynamics.

4.1 Our solution

Given the problems of access bandwidth variations explained above, at each variation we may need to reallocate the adaptation rules to the trees. To do so, we can recompute the multicast trees; In other words, we can run the algorithm that computes the multicast trees and replace the adaptation rules at every bandwidth variation. Tree re-computing may appear to be easy but it is very costly in term of computation time, especially, if the variations are too frequent.

To avoid re-computing the trees, we need a solution with low complexity, hence more responsiveness, less computational load for the controller and less risk of disruption. To do so, we have created a solution that consists of adapting the existing multicast trees; This solution does not rebuild the multicast trees but replaces optimally the adaptation rules. Unlike the re-computing, it does not require any path computation but just requires moving upward or downward the rules in the trees.

To analyze our static system over the duration of a call when time-varying access bandwidths are experienced, we have created adaptation algorithms. These algorithms are especially fast when the depth of the multicast trees is small, thus they improve considerably the complexity, the processing time and the reactivity.

4.2 Adaptation Algorithm

In this section, we present algorithms that replace optimally the adaptation rules. This does not require in any way the recomputing of trees but rather just demands to move the rules' locations in the trees. To change the rules' locations, we propose algorithms that move them upward or downward along the trees depending on the type of the bandwidth variation. The speed of these algorithms depends on the depth of the multicast trees and improves the complexity, the processing time and the reactivity.

The adaptation algorithm consists of:

- An initial phase of setting up the call by creating multicast trees (section 4.2.1).
- Adaptation phases where the multicast trees should adapt to the different types of access bandwidth variations (section 4.2.2, section 4.2.3).

4.2.1 Initial phase

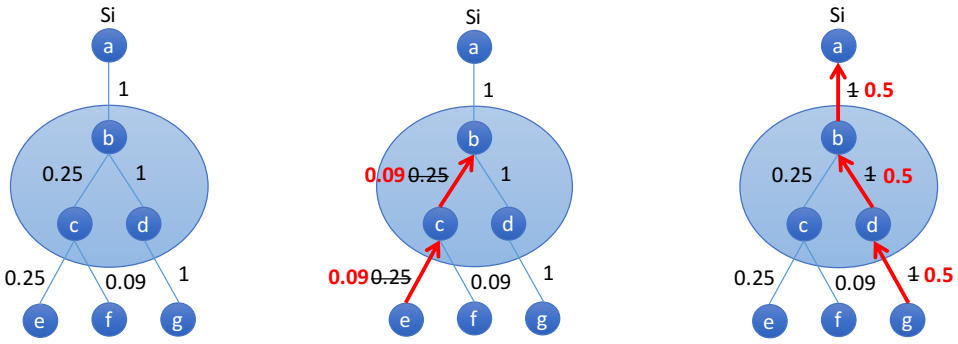
As said in section 2.3, SPT and MST methods compute the multicast trees and place the adaptation rules at appropriate tree nodes. We use them in the

initialization phase of our dynamic algorithm, i.e., when a call is established. However, when a bandwidth variation occurs, as recomputing the trees is costly, we opt for keeping them unchanged. But in order to guarantee that each participant receives the highest video quality allowed by its bandwidth, our algorithm recomputes the bitrates according to Eq.(2.1). This equation is used in 2.3 in order to set the definitive values of the bitrates. In a dynamic context, it should be used at each bandwidth variation.

The basic principle of our algorithm is to always satisfy these constraints but with minimum computation. Thus, when a bandwidth variation occurs, the bitrates are recomputed. If there is a bitrate change at a receiver, this change is propagated and the adaptation rules are pushed upward through the tree. In the same way, if a bitrate change occurs at the sender, the propagation is done downward.

This algorithm is expected to be performed by the SDN controller, and it takes as input the global model of the network in the controller. The forwarding/adaptation rules are then propagated to the SDN switches.

4.2.2 Change at a receiver



(a) Tree before bitrate change.

(b) Bitrate change propagated.

(c) Bitrate change propagated until the sender.

FIGURE 4.1: Receiver's bitrate change propagation

Figure 4.1 shows the propagation mechanism. Figure 4.1(a) depicts a multicast tree where node a is the sender and nodes e, f, and g are the receivers. The other nodes are SDN switches. The bitrate of each link is beside it. In Figure 4.1(b), because of a bandwidth change, the bitrate of (c,e) is no longer 250 Kbps but 90 Kbps. While neither e nor f need a stream of 250 Kbps, the layer corresponding to this rate is dropped at b, the parent of c. In Figure 4.1(c), the bitrate of (d,g) changes from 1 Mbps to 0.5 Mbps. This change is propagated until b, but neither c nor d require 1 Mbps, thus the bitrate of a is also adapted. This method adapts the bitrate upward from a

receiver, but it is infrequent that it reaches the sender. It happens only in the case where the bitrate of the receiver and the sender are the same, and a change of the first one affects the second one. The goal of this algorithm is to minimize the computation task while minimizing the consumed bandwidth. To do so, it places the adaptation rules that drop higher quality streams as close as possible to the sender, thus saving the bandwidth in the core network.

Algorithm 2: Relocate Up

```
1 Input: A tree T_i and a node r
2 b_max ← max_{1≤j≤k}(b_i(P_i(r), c_{i,j})) with c_{i,j} ∈ C_i(P_i(r))
3 b_min ← min(b_max, b(s_i))
4 if (b_min ≠ b_i(P_i(P_i(r)), P_i(r))) then
5     b_i(P_i(P_i(r)), P_i(r)) ← b_min
6     AdaptRule(P_i(r))
7     if P_i(P_i(r)) ≠ s_i then
8         RelocateUp(T_i, P_i(r))
9 AdaptRule(P_i(r))
```

Algorithm 2 formalizes this method. We use the same model and notations as in Section 2.2.2. It takes as input the receiver where a change occurs, then propagates the change recursively. At each step, it checks if all the children of a node $P_i(r)$ receive less than their parent (line 3). If it is the case, $P_i(r)$ does not need to receive its current bitrate from $P_i(P_i(r))$. The latter's bitrate is adapted (line 5) and the adaptation rules at $P_i(r)$ are updated by Algorithm 4 (line 6) further described in this chapter. The algorithm operates recursively (line 8) until it reaches the sender (line 7) or there is no need to go further, i.e., the bitrate received by $P_i(r)$ shall not be changed.

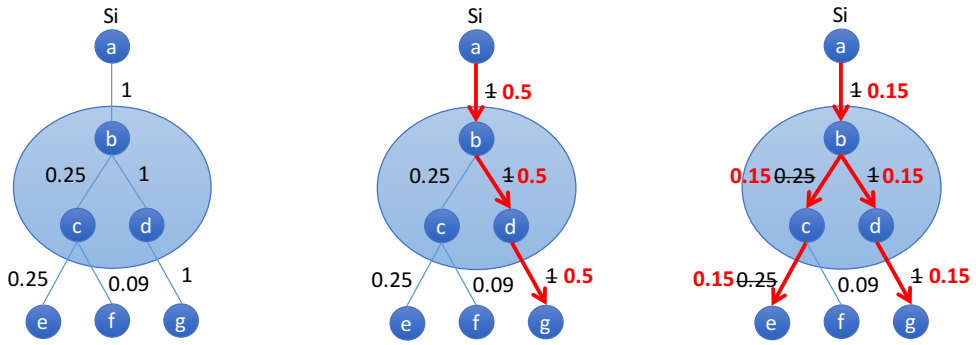
4.2.2.1 Complexity of "Relocate Up" algorithm

In Algorithm 2, the number of recursive calls is bounded by the depth of the multicast tree, which is itself bounded by the diameter of the network. The diameter is in $O(n)$ in the worst case, however, it is much smaller in random and realistic networks. It is well known that the diameter of an Erdős-Rényi[1] graph $G(n, \phi)$ is "almost constant[2]" when ϕ is fixed, and the diameter of a scale-free graph is $\sim \log n / \log\log n$ (Bollobás and Riordan, 2004). At each recursive call, Algorithm 2 checks the bandwidth between $P(r)$ and its children. This suggests that this operation is in $O(p)$. (because the number of children of any node in the tree is smaller than the number of participants). Assuming that the diameter of the network is in $O(\log n)$, the complexity of Algorithm 2 is in $O(p \log n)$.

[1] Erdős-Rényi graphs are usually denoted by $G(n, p)$ where p is a probability to exist for a link. We replaced it by ϕ in order to avoid confusion with the number of participants.

[2] The diameter of an Erdős-Rényi graph where ϕ is fixed tends toward 2 when $n \to +\infty$.

4.2.3 Change at the sender



(a) Tree before bitrate change.

(b) Bitrate change propagated.

(c) Bitrate change propagated until the receiver.

FIGURE 4.2: Sender's bitrate change propagation

When the bitrates are recomputed and the value of $b(s_i)$ is modified, the method used is almost the same, except that the propagation is performed by the sender to the receivers. Figure 4.2 shows the propagation mechanism. In Figure 4.2(b), because of a bandwidth change, the bitrate of (a, b) changed from 1000 Kbps to 500 Kbps. Since only receiver g can handle 1000 Kbps, he is the only one affected by the sender's bitrate change. Thus, all the layers of the path between a and g are dropped to 500 Kbps. In Figure 4.2(c), the bitrate of (a, d) changes from 1000 Kbps to 500 Kbps. This change affects this time both receivers e and g but not f. This method adapts the bitrate downward from a sender all the way to the receiver. Again, the goal is to keep the adaptation rules as close as possible to the sender. The main difference between this case and the previous one is that the propagation is not only done on a branch of the tree but can occur on several branches down to the leaves (receivers).

Algorithm 3: Relocate Down

```
1 Input: A tree T_i and node r
2 foreach c_{i,j} ∈ C_i(r) do
3     if b_i(r, c_{i,j}) ≠ b_i(P_i(r), r) then
4         b_i(r, c_{i,j}) ← b_i(P_i(r), r)
5         AdaptRule(P_i(r))
6         RelocateDown(c_{i,j})
```

Algorithm 3 is performed when the sender's bitrate decreases. It starts with each sender's child. From each one of these nodes, if their own children receive more than the sender's bitrate (line 3), the children bitrates should be changed (line 4) and the adaptation rules updated (line 5). The algorithm performs recursively (line 6) until reaching a leaf. Unlike Algorithm 2 that starts

from a leaf but infrequently reaches the sender, Algorithm 3 always reaches at least one receiver. Because there is always a receiver that receives the same bitrate as the sender (otherwise the sender could send a lower bitrate), and this receiver's bitrate must be updated.

4.2.3.1 Complexity of "Relocate Down" algorithm

Algorithm 3 browses all the tree in the worst case. Assuming that the diameter is in $O(\log n)$, the size of the tree is at most $O(p \log n)$, which is also the complexity of Algorithm 3.

4.2.4 Adapting the SVC Downsizing Rules

When incoming and outgoing bitrates are changed at a node, the adaptation rules should be updated. Algorithm 4 takes as input a node r and, for each one of its children, deletes the old rule if it exists and replaces it by the correct one, i.e., the pair (incoming bitrate into r, outgoing bitrate from r to its child).

Algorithm 4: Adapt rules

```
1 Input: A tree T_i and node r
2 foreach c_{i,j} ∈ C_i(r) do
3     if b_i(r, c_{i,j}) = b_i(P_i(r), r) then
4         Delete rule if exists
5     else
6         Rule(r, c_{i,j}) ← (b_i(P_i(r), r), b_i(r, c_{i,j}))
```

4.2.4.1 Complexity of "Adapt rules" algorithm

On any node, there is at most one rule per child. The number of children of any node is smaller than the number of participants, this gives a complexity of $O(p)$.

4.2.5 The general algorithm

Algorithm 5 is performed in the SDN controller. The call is first established by using algorithms MST or SPT of section 2.3. Then at each event, the controller reacts and adapts the multicast trees. We consider four events:

- $B(r)\downarrow$: The downlink bandwidth of a node r decreases. This can lead to a change of $b_i(r)$ and $b(s_i)$ for each tree T_i because of formulas (2.1). In

this case, the bitrates are recomputed and the new bitrate of r is propagated in all the trees (except the one where it is the sender) using Algorithm 2. Thus the complexity of processing this event is in $O(p^2 \log n)$.

- $B(s_i) \downarrow$: The uplink bandwidth of a node s_i decreases. This can affect the bitrate of s_i and those of the receivers, but only in the tree T_i where s_i is the sender. The change is propagated using Algorithm 3 on the access node of s_i, which is its only child. The complexity is that of Algorithm 3, i.e., $O(p \log n)$.
- $B(s_i) \uparrow$: The uplink bandwidth of a node s_i increases. Again, this can impact the bitrates of s_i and the receivers $r_{i,j}$. After recomputation, if the receivers' bitrates do not change, there is no need to propagate the new bitrate of s_i, because no receiver can get a higher bitrate. Otherwise, the new bitrates are propagated in T_i using Algorithm 2. Likewise the case $B(r) \downarrow$, the complexity is in $O(p^2 \log n)$.
- $B(r) \uparrow$: The downlink bandwidth of a node r increases. This case is more complex since, according to formulas 2.1, it can impact all the bitrates in all the trees (except the one where r is the sender). For each tree, there are two possible cases. i) The sender's bitrate is not affected, in this case, the new downlink bitrate of r is propagated upward. ii) The sender's bitrate is affected, in this case, it can, in turn, affect the other receivers' bitrates. This case is similar to the previous one where the sending bitrate increases. The complexity of processing this event is in $O(p^3 \log n)$.

4.2.5.1 Complexity of the general algorithm

Note that the number of participants is much smaller than the network size, i.e., $p \ll n$. If we consider the network size as a parameter, Algorithm 5 processes each event in $O(n)$ if the diameter is linear and in $O(\log n)$ in the more realistic case where the diameter is logarithmic. This complexity is much lower than the complexity of MST/SPT that is in $O(n^3)$ if the network is dense and $O(n^2 \log n)$ if the network is sparse (see the complexity explanation in Section 2.3.1). A lower complexity when processing an event implies less consumed resources in the controller and better reactivity for the

participants.

Algorithm 5: General algorithm

```
1  Perform MST or SPT to create a video conference
2  while an event occurs do
3      if Downlink bandwidth of a receiver r decreases (B(r) ↓) then
4          Recompute all the bitrates
5          foreach Multicast tree T_i do
6              RelocateUP(T_i, r)
7      if Uplink Bandwidth of a sender decreases (B(s_i) ↓) then
8          Recompute the uplink bitrate of s_i
9          RelocateDown(c) where {c} = C(s_i)
10     if Uplink Bandwidth of a sender increases (B(s_i) ↑) then
11         Recompute the uplink bitrate of s_i
12         Recompute the downlink bitrates of each r_{i,j}
13         foreach receiver r_{i,j} of T_i do
14             if the bitrate of r_{i,j} increased after recomputation then
15                 RelocateUP(T_i, r_{i,j})
16     if Downlink bandwidth of a receiver r increases (B(r) ↑) then
17         Recompute all the bitrates
18         foreach Tree T_i do
19             if the bitrate of s_i increases after recomputation then
20                 foreach receiver r_{i,j} of T_i do
21                     if the bitrate of r_{i,j} increased after recomputation then
22                         RelocateUP(T_i, r_{i,j})
23             else
24                 RelocateUP(T_i, r)
```

4.3 Simulations

4.3.1 Simulations parameters

For the dynamic evaluation, the topology type and size are built in the same way as the topologies presented in the parameter of the Section 2.4.1. Also the access and core links' capacities and delays requirements are the same.

The only difference occurs within the topology mode; in this section, we keep comparing our algorithms to the same mode mentioned in Chapter 2.4.1. Although, due to the dynamic environment, changes can occur in the access links' capacities. In our simulation, the bandwidth variations are

implemented by modifying, at each variation, the value of the downlink access bandwidths randomly between [1200Mbps, 14Mbps] or the value of the downlink access bandwidths between [90kbps, 1.5Mbps]. The bandwidth variation follows an exponential distribution with a positive scale parameter alpha = 0.286. To adapt to those changes, we either recompute the trees or adapt to the changes (as discussed in Section 4.1). Therefore, the results will represent the six following modes:

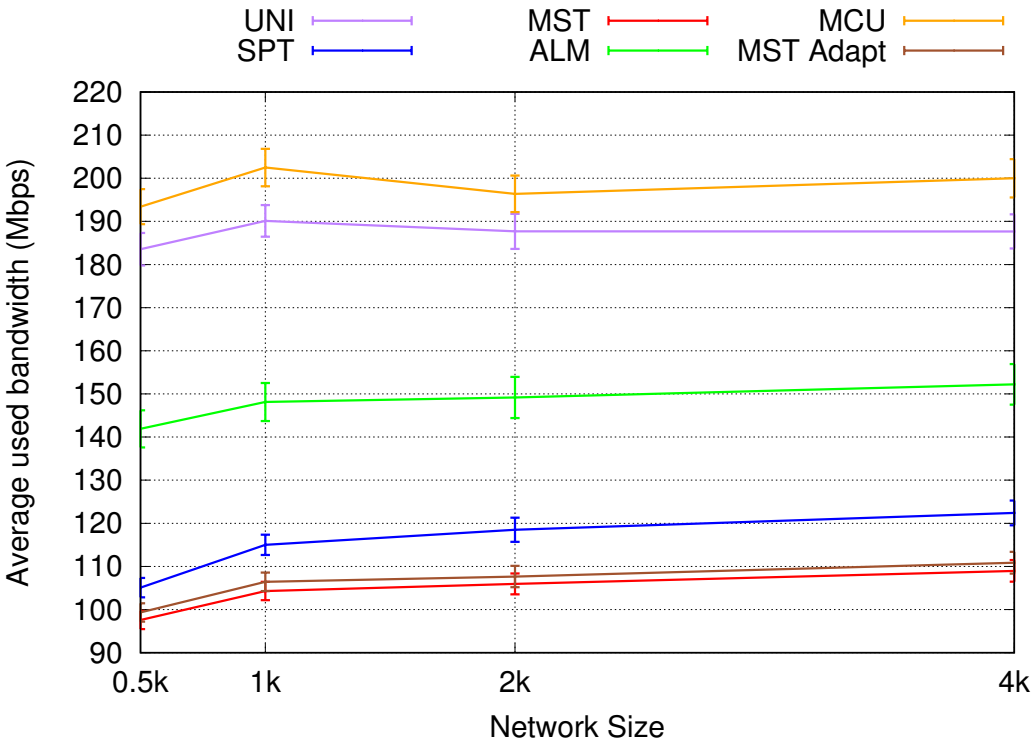
- **Uni-recomputation mode**: Unicast method with recomputation at each event. At each change of access bandwidth, the shortest paths between each pair of participants are recomputed.
- **MST-recomputation mode (MST)**: MST method with recomputation at each event.
- **SPT-recomputation mode (SPT)**: SPT mode with recomputation at each event.
- **ALM mode**: we have implemented ALMI as an ALM mode. ALMI is an Audio/video conferencing centralized protocol, where the forwarding responsibility is given to the end hosts. It consists of minimizing the total number of hops or the delay using MST. In a dynamic environment, ALMI recomputes all the trees to adapt to the bandwidth change.
- **MCU mode**: consists of connecting the end hosts to the MCU, then to create paths from the MCU to the other end hosts. In a dynamic environment, the path *end host - MCU* or *MCU - end host* is recomputed, depending on the type of access change.
- **MST-adaptation mode**: Our solution MST-adaptation mode is performed to establish the video call, then the multicast trees are adapted at each event according to Algorithm 5. We have implemented our solution only for MST, even if the results in Chapter 3 show that MST creates more latency, as it also saves a lot of bandwidth. Since we are computing the tree one time only, we favor bandwidth savings over computation complexity.

Same as in 3.3.1 each point on the following plots is the average of the values obtained from 400 simulations. This average is presented accurately using a confidence interval of 95% displayed on the plots.

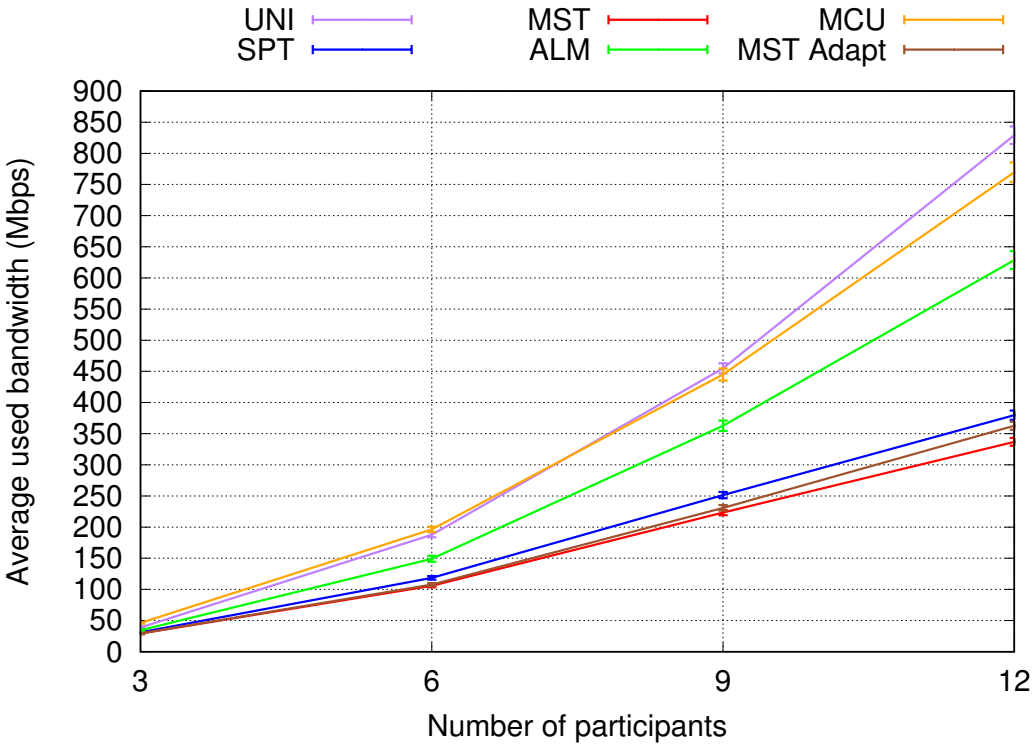
4.3.2 Simulation results

Figures 4.3 and 4.4 show the impact of the network size and participants number per call on the average bandwidth usage for Magoni-Pansiot and Erdős-Rényi topologies respectively.

In both topologies, the results are almost the same for our solution and MST-recomputing, and better than the other algorithms. It appears that the

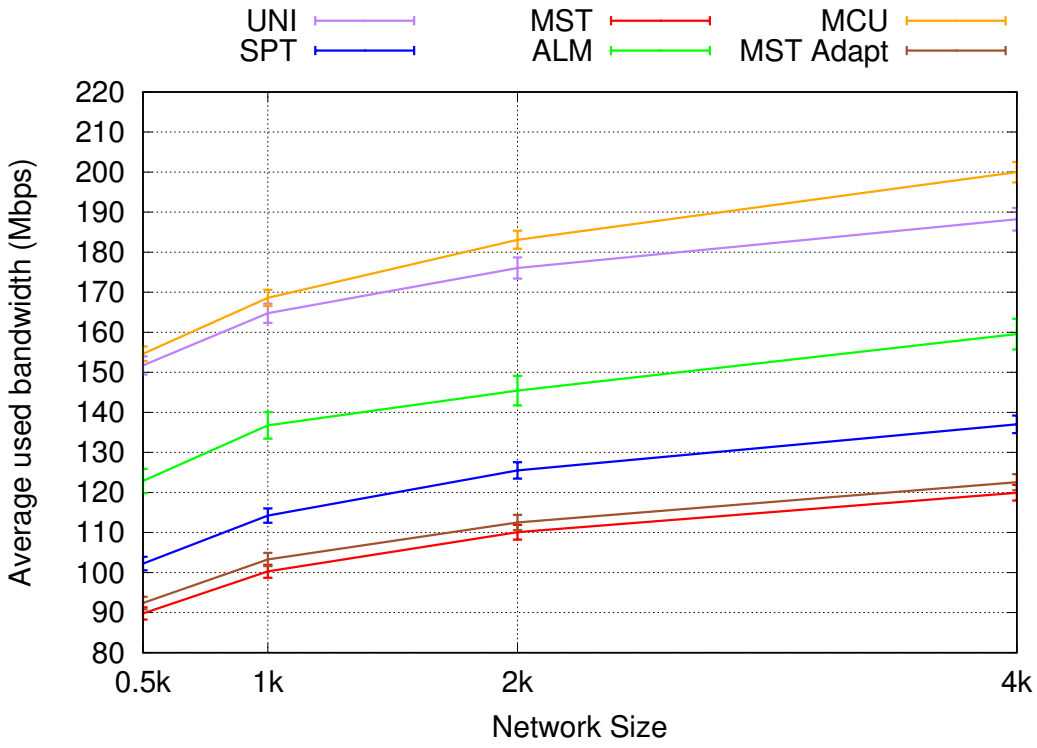


(a)

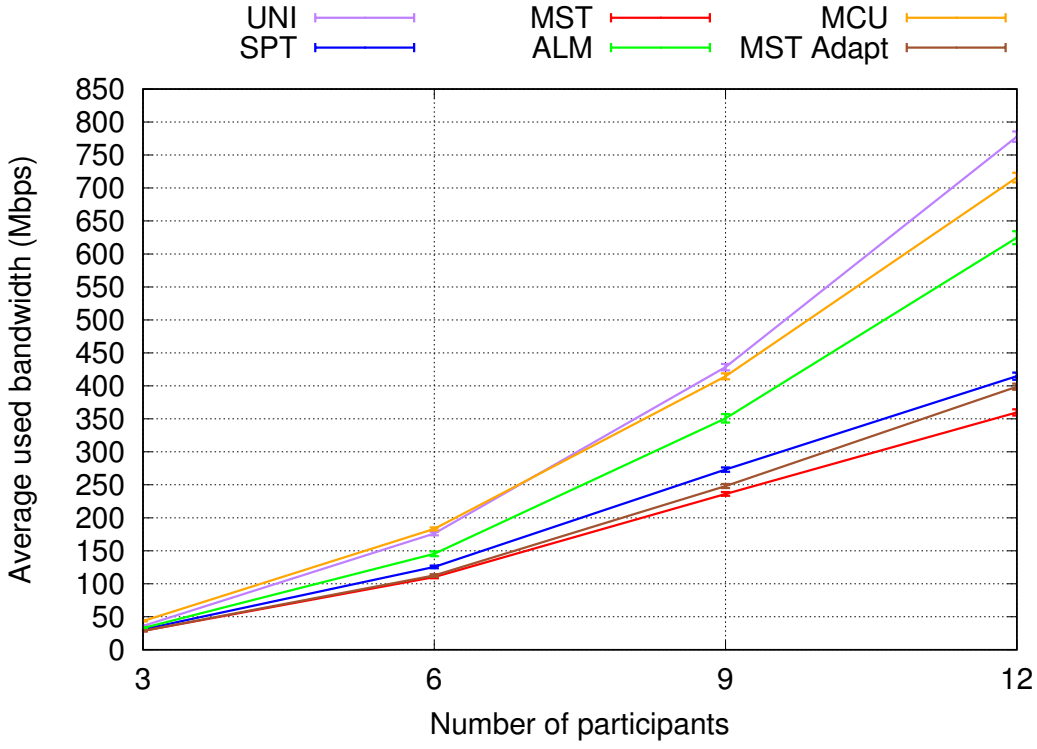


(b)

FIGURE 4.3: Average bandwidth usage depending on the network size or the number of participants on MP (with 6 participants per call for (a) and a network size of 2k node for (b)).



(a)



(b)

FIGURE 4.4: Average bandwidth usage depending on the network size or the number of participants on ER (with 6 participants per call for (a) and a network size of 2k node for (b)).

bandwidth variations at access links are very small compared to the bandwidth availability in the core links. They are too small to induce a change in the multicast tree topologies.

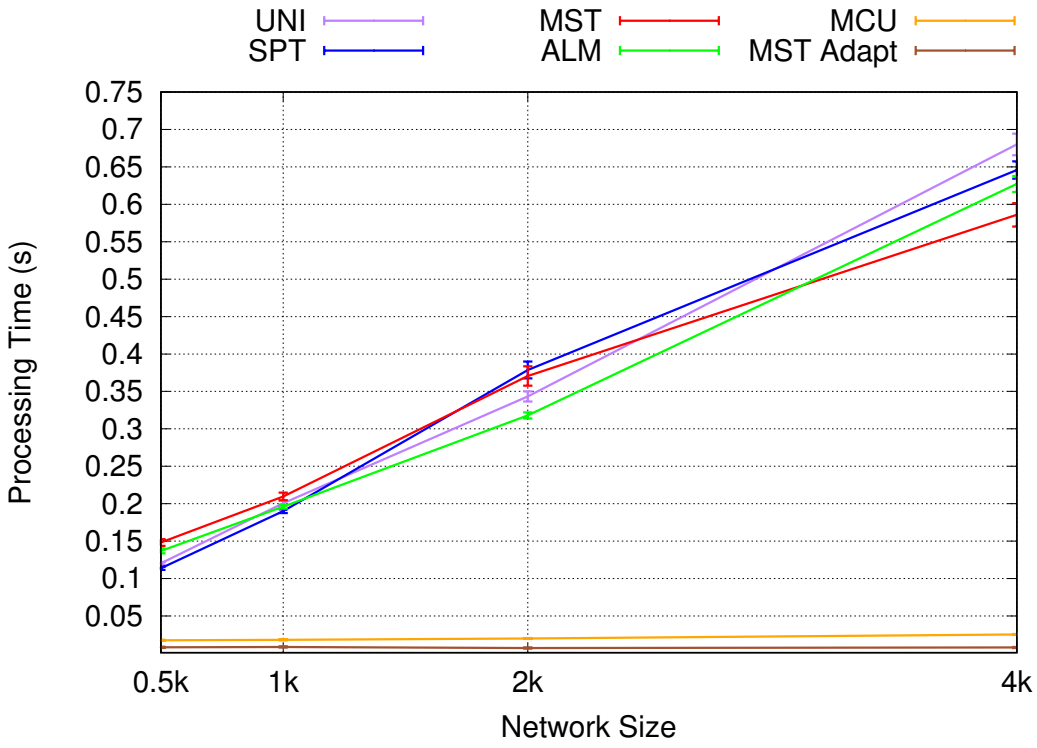
Figures 4.5 and 4.6 show the impact of the network size and participants number per call on the processing time for Magoni-Pansiot and Erdős-Rényi respectively.

All the figures show that our MST adaptation method, as well as MCU, are slightly affected by the network size and the participants' number. MST adaptation methods are much faster than the other approaches due to the fact that it does not need to recompute all the tree. On the other hand, MCU has also a very low processing time due to the fact that MCU computes all the shortest paths and stores them so it can use them to recompute the trees without the need for recomputing the shortest path.

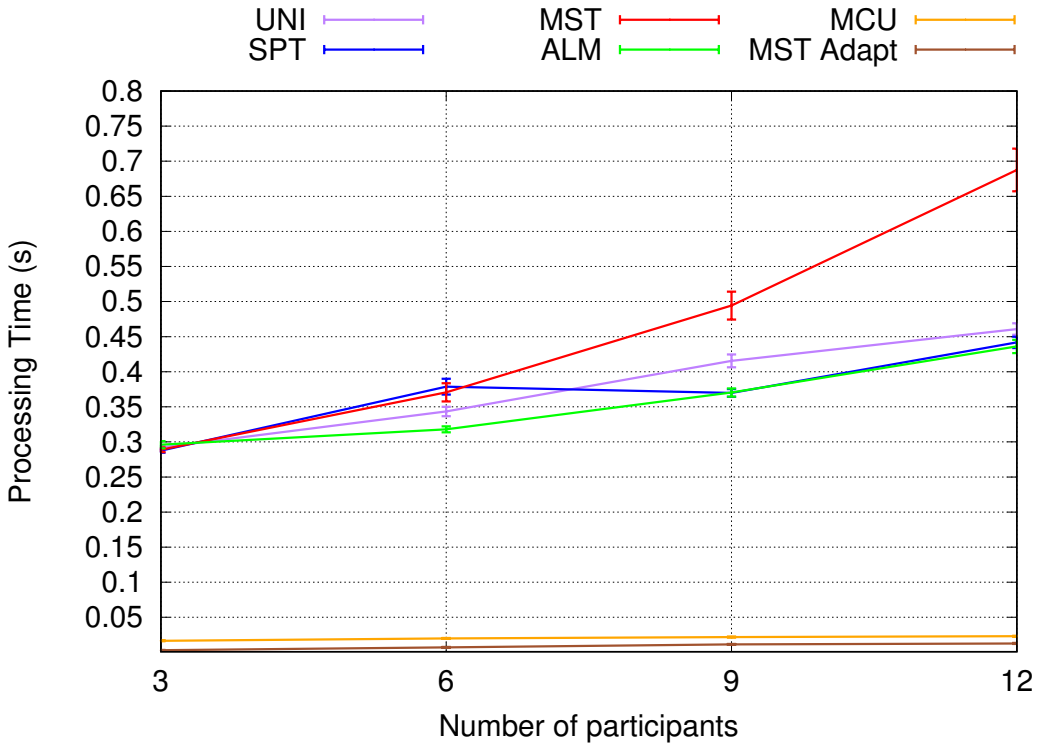
Consequently, since MCU consumes a lot of bandwidth, we can see that our MST solution is the best solution regarding the processing time compared to all the other solutions while providing the lowest bandwidth consumption.

4.4 Conclusion

Network dynamics affecting access links can impact enormously the video quality. In order to establish a video conference call with the best possible video quality for the users, we have evaluated the methods defined in the previous chapter in a dynamic context where access channel bandwidth variations occur. We have created a fast adaptive algorithm, based on tree traversal ideas, that adapt the video layering without recomputing the multicast trees while the call is ongoing. The low complexity of this algorithm allows high reactivity to network changes and low resource consumption in the SDN controller. The simulation results confirm the efficiency of our algorithms in terms or processing time. Moreover, they show that our solution provides as much bandwidth savings as more costly and elaborate methods.

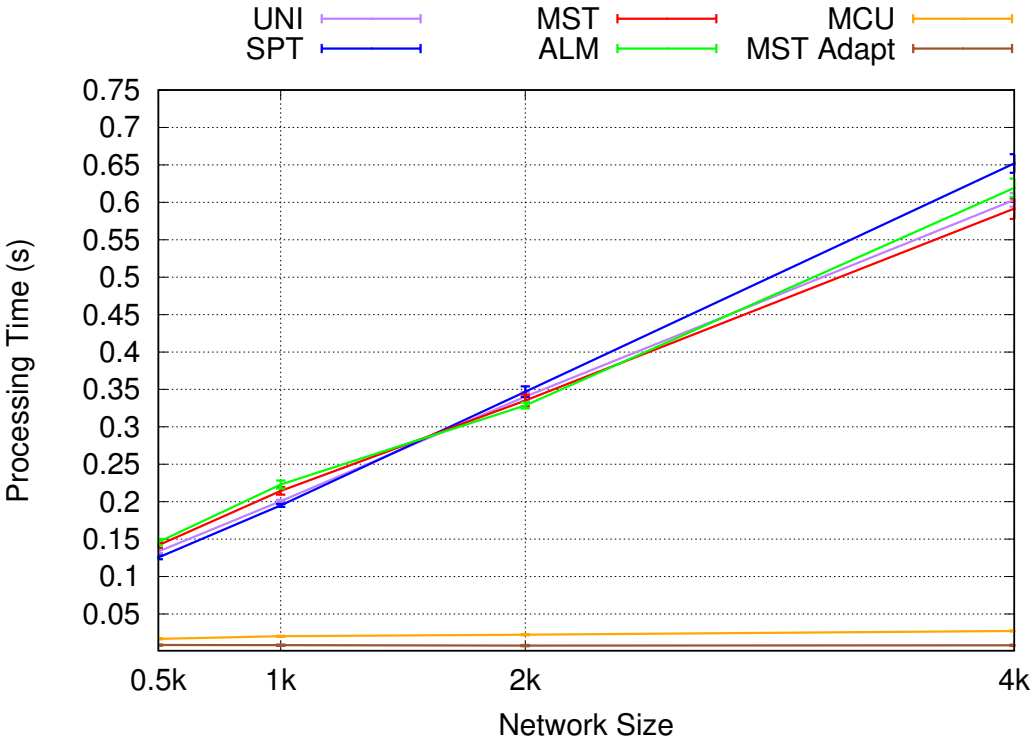


(a)

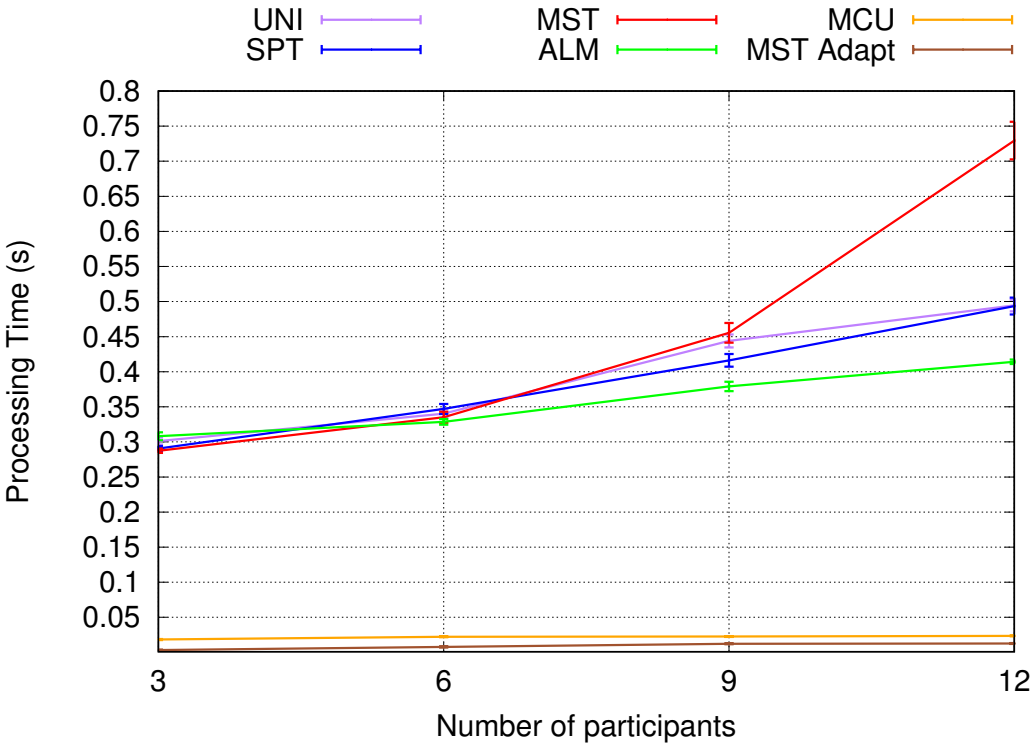


(b)

FIGURE 4.5: Average processing time depending on the network size or the number of participants on MP (with 6 participants per call for (a) and a network size of 2k node for (b)).



(a)



(b)

FIGURE 4.6: Average processing time depending on the network size or the number of participants on ER (with 6 participants per call for (a) and a network size of 2k node for (b)).

Conclusion and perspectives

With the advent of the Internet as a cheaper alternative to the private telecommunication networks, teleconference technologies have gained a wide interest among individual users and organizations. However, teleconference systems should provide the same uniformity as dedicated telephone networks in terms of performances, as well as, a high video quality. Thus, high quality conference systems have always been challenging to achieve due to their specific constraints on network, user requirements, and device heterogeneity. Many architectures and protocols were adopted by conference systems in order to provide the best quality of experience to the users.

Many companies use centralized teleconference systems in order to save money. Currently, the most used centralized architecture for teleconference systems is based on the Multipoint Control Unit (MCU). The MCU device represents the central server in a centralized conference system. Besides the control functionality, MCU enables the adaptation of media traffic in order to meet the capabilities of heterogeneous user devices. However, the adaptation process in centralized teleconference systems with heterogeneous users requires a high computation cost and resource usage on the MCU. In addition, the access channel to the MCU, will need a large bandwidth in order to avoid bottlenecks. These bandwidth and resource limitations prevent centralized conferences systems from being scalable.

To improve scalability, researchers have developed new conference systems based on distributed architectures. Distributed teleconference systems, such as P2P and ALM teleconference systems, overcome the scalability issue, since the system does not rely anymore on one central server. The stream adaptation, in distributed teleconference systems with heterogeneous users is done at the end user devices, or at an intermediate node in the network. However, P2P systems increase the network load, and ALM systems increase delays. Researchers have tried to reduce the network load by using IP-multicast connections. IP-multicast allows a better distribution of the media compared to IP-unicast since it does not require stream duplication. However, IP-multicast is still not widely supported by network providers and operators due to its requirements on the network (i.e., it requires router resources for storing multicast states).

In recent years, the introduction of software-defined networking (SDN) enabled new possibilities for better network management. SDN enables the separation between a control plane and the media plane. The SDN controller, responsible for the control plane, has an abstract global view of the network.

It can collect infrastructure information and can decide how to redirect the traffic in order to provide better resources savings. Then, the media plane transfers the traffic based on the controller instructions. The communication system can benefit from the SDN controller's global view on the network condition, in order to provide better control and management. Therefore, in a conference system, the SDN controller can be considered as an MCU and can make decisions about stream redirection and adaptation in order to maximize the network utility, reduce network load and meet end-user requirements. Unlike MCU-based systems, the traffic does not have to go through the controller in order to reach its destination. This has motivated us to design new distribution and adaptation algorithms for teleconferencing systems over an SDN architecture. We have contributed to the following works:

- In a multi-party teleconference call, users can be affected by the channel/devices limitations that requires a stream adaptation to a lower quality. In both centralized and distributed conference architecture mentioned before, this adaption is done either at the end user device or at a intermediate node in the network (central server), using a transcoder or some form of video layering (e.g., SVC). If the adaption is done at the end user device, the original stream will pass through all the network without being changed until it reaches the receiving end user. This method is simple, however, it causes a wasteful usage of the network bandwidth. If the adaption is made at a central server (e.g., MCU), the adaptation will cause an overload on the central server and may increase the latency. To overcome the adaptation problem, we have defined a multi-party audio conference model based on the SDN architecture. We have implemented an algorithm that creates multicast trees from each sender to each participant and places adaptation rules in specific nodes in order to adapt the stream to every receiver's limitations. The results show that using multicast with stream adaptation placement enables important bandwidth savings compared to typical solutions (e.g., MCU, unicast n times, ALM).

- Similarly, we have investigated using SDN capabilites for implementing SVC adaptation inside the network for video conferencing systems. We used the algorithm discussed before to reduce the bandwidth consumed by video-conferences. We took advantage of the SVC layering feature and multicasting to allow video layer dropping at specific locations. The results show that, with the same network infrastructure, we are able to fit, simultaneously, a higher number of teleconference calls based on our model than with the MCU-based conference calls.

- During a conferencing session, the access link can be affected by interference which reduces the users' capacity of sending or receiving the best quality streams. We have tested our previous designed algorithm in a dynamic context where access channel bandwidth variations occur. We have created adaptation algorithms, in order to adapt to the access bandwidth variation without re-computing the multicast trees.

We compared the behavior of our system to MCU- and ALM-based systems. The results shows that our algorithm have faster processing adaptation than the other systems.

We believe that our work is an important step towards designing a better conferencing system. However, our proposed model and algorithms are implemented in Python 2.7.10, using the NetworkX library as a first step for estimating the efficiency of our proposal.We aim to implement our system in a real SDN environment by using an SDN controller such as NOX, POX, Floodlight, etc. The SDN controller will be composed of different modules for better managing teleconference calls. Each module will be responsible for one or more of these functions:

- topology management: information about the network topology will be collected to enable a global view of the controller in order to help to create routes and multicast trees. This information includes node and link capacities, it also notifies about any link failure.
- management of the participants: information about all the conference calls in the system as well as their participants (i.e., IP and MAC addresses). Video capabilities will be collected to deal with the heterogeneity of devices and deliver the most convenient video stream quality,
- SVC layering management: each SVC layer will be forwarded in a specific flow, and will be distributed through specific ports.
- multicast tree management: multiple multicast trees will be created for call establishment (one per participant), stream distribution.
- routing management: new routes will be found in the case of network congestion or link/node failures, taking into account the link capacities and the network load.
- QoS management: network state statistics will be collected during the call to adapt to the network changes and adjust the video streams.
- call admission/exit: the entry and exit of participants will be managed in order to adapt the trees to this interactivity without affecting the other participants.

The modules will communicate with each other in order to improve bandwidth savings, reduce delays and guarantee the quality of the video conferencing for every participant.

There are interesting perspectives that discuss algorithms at the theoretical level. Among these perspectives, there are approximation algorithms. α-Approximation algorithms for a NP-hard optimization problem, give a solution that is at worst $\alpha \times$the optimal solution. MST and SPT work well in practice, it would be interesting to analyze MST theoretically, and especially to know if MST gives a guarantee of approximation. The problem of building a multicast tree corresponds to the Steiner tree problem. There is a trivial approximation algorithm for the Steiner tree that gives a guarantee of approximation of $2 - 2/P$, with P, the number of participants. This algorithm works by computing the metric closure of the graph then by calculating the minimum spanning tree on this closure. The weight of the resulting tree is at worst $2 - 2/P$-approximations of the optimal tree. Robins and Zelikovsky (2000) propose a purely combinatorial algorithm with an approximation ratio of 1.55, however, the complexity of this algorithm is very high. Byrka *et al.* (2010) propose an algorithm using Linear Programming (LP) and giving a ratio of 1.39. It would be interesting to study the approximation ratio of our MST algorithm (if any), and to determine its efficiency compared to other approximation algorithms.

During a conference call, one or more participants can connect or exit the call at any time. This interactivity generates what is called the problem of online Steiner tree. Resolving the problem of online Steiner tree consists in creating a minimum number of operations to adapt the tree while keeping a good guarantee of approximation. Gupta and Kumar (2014) proposes an algorithm that mentions a logarithmic approximation factor at each connection/disconnection while performing a constant number of operations. It will be interesting to study this in practice and see how we can improve on this research to provide better interactivity.

Appendix A

Algorithms ALM and MCU

This appendix explains the MCU and ALM solution algorithms used in our simulations in order to evaluate our proposed solution.

Algorithm for setting up the ALM tree

Algorithm 6 creates the global ALM tree that connects all participants in a conference call together. First, it starts by finding the shortest path between each pair of participants (p, p'), and saves them in the list *allPaths* (line 5). Then, $SortPaths()$ function sorts the paths in *allPaths* according to their length (line 6). Thus, the shortest path *path*, between the pair of participants $(p1, p2)$, contains the smallest number of nodes (lines 7-9). The algorithm adds the first path to the tree using $AddShortestPathToTree(path)$ function (line 11), and defines the participants $p1$ and $p2$ as participants already existing in the tree by adding them to the list *treeParticipants* (line 10). Now, the tree contains its first two participants and the shortest path connecting them. The goal of the algorithm is to connect the remaining participants to a closest participant already existent in the tree (i.e. to a participant in *treeParticipants*). *allPaths* is used to choose a path in order to connect the new participants. When the participant is connected to the tree, all the other shortest paths connecting him to participants already existent in the tree should be removed from the *allPaths*. For now only *path* will be removed(line 12). Since the paths in *allPaths* are sorted, the first path between one participant existent in the tree and another out of the tree (line 17) will be chosen as the next shortest path connecting an outside participant to the tree (lines 17-22). Since a new participant is added to the tree (lines 18-21), all the remaining paths connecting him to another participant existent in the tree should be remove (lines 23-27).

This algorithm creates the ALM tree, however, in order to distribute the streams, a sender will send a certain stream resolution to his neighbors in the

tree. The neighbors will adapt this stream to their capacities and send the adapted stream to their neighbors.

Algorithm 6: Setup ALM Tree

```
1  Input: a set of participants in a call P
2  Output: a multicast tree T
3  T ← {∅}
4  treeParticipants ← {∅}
     # treeParticipants is a set of participants added to the tree
5  allPaths ← BuildShortestPath(p, p′) with p, p′ ∈ P
     # It contains one shortest path from each participant to another
6  SortPaths(allPaths)
     # sort the shortest paths by their length
7  path ← allPaths[0]
8  p1 ← path[0]
9  p2 ← path[path.length() − 1]
     # p1 and p2 are the closest participants to each other
10 treeParticipants.add(p1, p2)
11 AddShortestPathToTree(T, path)
12 allPaths.remove(path)
13 while allPaths ≠ {∅} do
14     foreach path ∈ allPaths do
15         p1 ← path[0]
16         p2 ← path[path.length() − 1]
17         if p1 ∈ treeParticipants ∨ p2 ∈ treeParticipants then
18             if p1 ∈ treeParticipants then
19                 treeParticipants.add(p1)
20             else
21                 treeParticipants.add(p2)
22             AddShortestPathToTree(T, path)
23             foreach path1 ∈ allPaths do
24                 p1 ← path1[0]
25                 p2 ← path1[path1.length() − 1]
26                 if p1 ∈ treeParticipants ∧ p2 ∈ treeParticipants then
27                     allPaths.remove(path1)
28             Break
```

Algorithm for setting up the MCU tree

Algorithm 7 creates a distribution tree for each participants. First, it starts by finding the shortest path between each participant and the MCU, and saves them in the list *allPaths* (line 4). To create the tree T_s of a sender s, the algorithm runs $FindShortestPathToMCU(s, allPaths)$ function in order to

Algorithm 7: Setup MCU Tree

```
1  Input: a set of participants in a call P and the MCU
2  Output: a trees T_p with p ∈ P
3  allPaths ← BuildShortestPath(p, MCU) with p ∈ P
     # It contains one shortest path from each participant to the MCU
4  foreach s ∈ P do
5      T_s ← {∅}
6      pathToMCU ← FindShortestPathToMCU(s, allPaths)
7      AddShortestPathToTree(T_s, pathToMCU, b(s))
8      foreach r ∈ P/{s} do
9          pathToMCU' ← FindShortestPathToMCU(r, allPaths)
10         pathFromMCU ← ReversePath(pathToMCU')
11         AddShortestPathToTree(T_s, pathFromMCU, min(b(s), b(r)))
```

search, in the list *allPaths*, for its shortest path to the MCU *pathToMCU* (line 6). Then, it adds this shortest path to the tree T_s, using *AddShortestPathToTree*() function (line 7) that takes as parameters the tree to which we are adding the path and the bitrate associated to it. Since the sender in MCU-based will send its maximum bitrate stream to the MCU, the bitrate on *pathToMCU* path is the bitrate of the sender s, $b(s)$. To complete the tree, the algorithm search for the path connecting the receivers to the MCU (line 9), *pathToMCU*$'$, reverse them (line 10) to get the path from the MCU to the receivers, *pathFromMCU*, then add them to the tree with a bitrate handled by the receiver capacity $min(b(s), b(r))$, using *AddShortestPathToTree*() (line 11).

Appendix B

List of Publications

1. Al Hasrouty, Christelle and Autefage, Vincent and Olariu, Cristian and Magoni, Damien and Murphy, John (2016) "*SDN-driven multicast streams with adaptive bitrates for VoIP conferences*", IEEE International Conference on Communications 2016.
2. Al Hasrouty, Christelle and Autefage, Vincent and Olariu, Cristian and Magoni, Damien and Murphy, John (2016) "*Arbres de diffusion pour sessions MVoIP avec flux hétérogènes*", ALGOTEL 2016-18èmes Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications 2016.
3. Al Hasrouty, Christelle and Olariu, Cristian and Autefage, Vincent and Magoni, Damien and Murphy, John (2017) "*SVC Videoconferencing Call Adaptation and Bandwidth Usage in SDN Networks*", IEEE Global Communications Conference 2017.
4. Al Hasrouty, Christelle and Olariu, Cristian and Autefage, Vincent and Magoni, Damien and Murphy, John (2017) " *Adaptation des flux de vidéoconférence en coeur de réseau SDN*", Rencontres Francophones sur la Conception de Protocoles, l'Évaluation de Performance et l'Expérimentation des Réseaux de Communication 2017.
5. Al Hasrouty, Christelle and Lamali, Mohamed Lamine and Magoni, Damien and Murphy, John (2017) " *SDN-enabled Adaptation of Videoconference Streams to Network Dynamics*", IEEE Global Communications Conference 2017.
6. Al Hasrouty, Christelle and Lamali, Mohamed Lamine and Olariu, Cristian and Autefage, Vincent and Magoni, Damien and Murphy, John (2018), "*Adaptive Multicast Streaming for Videoconferences on Software-Defined Networks*", to appear in Computer Communications - The International Journal for the Computer and Telecommunications Industry 2018.

Bibliography

Adams, Andrew, Jonathan Nicholas, and William Siadak (2004). *Protocol independent multicast-dense mode (PIM-DM): Protocol specification (revised)*. Tech. rep.

Adel, Mohamed, Haytham Assem, Brendan Jennings, David Malone, Jonathan Dunne, and Pat O'Sullivan (2013). "Improved E-model for monitoring quality of multi-party VoIP communications". In: *IEEE Globecom Workshops*, pp. 1180–1185.

Alessandria, Eugenio, Massimo Gallo, Emilio Leonardi, Marco Mellia, and Michela Meo (2009). "P2P-TV systems under adverse network conditions: a measurement study". In: *INFOCOM 2009, IEEE*. IEEE, pp. 100–108.

Amad, Mourad, Ahmed Meddahi, and Djamil Aissani (2009). "Asynchronous distributed P2P-SIP model for VoIP". In: *Information Infrastructure Symposium, 2009. GIIS'09. Global*. IEEE, pp. 1–4.

Amirante, Alessandro, Tobia Castaldi, Lorenzo Miniero, and Simon Pietro Romano (2013). "On the seamless interaction between webRTC browsers and SIP-based conferencing systems". In: *IEEE Communications Magazine* 51.4, pp. 42–47.

Apu, Khalid Ibn Zinnah, Nafiz Mahmud, Firoz Hasan, and Sabbir Hossain Sagar (2017). "P2P video conferencing system based on WebRTC". In: *Electrical, Computer and Communication Engineering (ECCE), International Conference on*. IEEE, pp. 557–561.

Arango, Mauricio, Andrew Dugan, Isaac Elliott, Christian Huitema, and Scott Pickett (1999). *Media gateway control protocol (MGCP) version 1.0*. Tech. rep.

Arazi, Nitzan, Yaron Soffer, and Haim Barak (2002). *Wireless private branch exchange (WPBX) and communicating between mobile units and base stations*. US Patent 6,430,395.

Avramova, Zlatka, Danny De Vleeschauwer, Kathleen Spaey, Sabine Wittevrongel, Herwig Bruneel, and Chris Blondia (2007). "Comparison of simulcast and scalable video coding in terms of the required capacity in an IPTV network". In: *Packet Video 2007*. IEEE, pp. 113–122.

Banerjee, Suman, Bobby Bhattacharjee, and Christopher Kommareddy (2002b). "Scalable Application Layer Multicast". In: *Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pp. 205–217.

— (2002a). *Scalable application layer multicast*. Vol. 32. 4. ACM.

Banerjee, Suman, Christopher Kommareddy, Koushik Kar, Bobby Bhattacharjee, and Samir Khuller (2003). "Construction of an efficient overlay multicast infrastructure for real-time applications". In: *22nd Annual Joint Conference of the IEEE Computer and Communications (ICCC)*. Vol. 2, pp. 1521–1531.

Bankoski, James, John Koleszar, Lou Quillio, Janne Salonen, Paul Wilkins, and Y Xu (2011). *VP8 data format and decoding guide*. Tech. rep.

Bertó-Monleón, Ricardo, Enrico Casini, Rob van Engelshoven, Rob Goode, Klaus-Dieter Tuchs, and Tamas Halmai (2011). "Specification of a policy based network management architecture". In: *MILITARY COMMUNICATIONS CONFERENCE, 2011-MILCOM 2011*. IEEE, pp. 1393–1398.

Bhattacharyya, Supratik (2003). *An overview of source-specific multicast (SSM)*. Tech. rep.

Bollobás, Béla and Oliver Riordan (2004). "The diameter of a scale-free random graph". In: *Combinatorica* 24.1, pp. 5–34.

Boucadair, Mohamed and Christian Jacquenet (2014). "Software-defined networking: A perspective from within a service provider environment". In:

Brandenburg, Karlheinz (1999). "MP3 and AAC explained". In: *Audio Engineering Society Conference: 17th International Conference: High-Quality Audio Coding*. Audio Engineering Society.

Brief, Solution (2013). *SDN Security Considerations in the Data Center*.

Byrka, Jaroslaw, Fabrizio Grandoni, Thomas Rothvoß, and Laura Sanità (2010). "An improved LP-based approximation for Steiner tree". In: *Proceedings of the forty-second ACM symposium on Theory of computing*. ACM, pp. 583–592.

Cacheda, Rafael Asorey, Daniel Castro García, Antonio Cuevas, Francisco Javier González Castano, Javier Herrero Sánchez, Georgios Koltsidas, Vincenzo Mancuso, José Ignacio Moreno Novella, Seounghoon Oh, and Antonio Pantò (2007). "QoS requirements for multimedia services". In: *Resource Management in Satellite Networks*. Springer, pp. 67–94.

Caesar, Matthew, Donald Caldwell, Nick Feamster, Jennifer Rexford, Aman Shaikh, and Jacobus van der Merwe (2005). "Design and implementation of a routing control platform". In: *Proceedings of the 2Nd Conference on Symposium on Networked Systems Design & Implementation-Volume 2*. USENIX Association, pp. 15–28.

Cain, Brad (2006). "Source-specific multicast for IP". In:

Cain, Brad, Steve Deering, Isidor Kouvelas, Bill Fenner, and Ajit Thyagarajan (2002). *Internet group management protocol, version 3*. Tech. rep.

Campbell, Andrew T, Irene Katzela, Kazuho Miki, and John Vicente (1999). "Open signaling for ATM, internet and mobile networks (OPENSIG'98)". In: *ACM SIGCOMM Computer Communication Review* 29.1, pp. 97–108.

Casado, Martin, Michael J Freedman, Justin Pettit, Jianying Luo, Nick McKeown, and Scott Shenker (2007). "Ethane: Taking control of the enterprise". In: *ACM SIGCOMM Computer Communication Review*. Vol. 37. 4. ACM, pp. 1–12.

Castellanos, Wilder, Juan Guerri, and Pau Arce (2017). "SVCEval-RA: an evaluation framework for adaptive scalable video streaming". In: *Multimedia Tools and Applications* 76.1, pp. 437–461.

Chandra, S.P., K.M. Senthil, and M.P.P. Bala (2009). "Audio mixer for multiparty conferencing in VoIP". In: *IEEE International Conference on Internet Multimedia Services Architecture and Applications*, pp. 1–6.

Chandrasekar, V. and K. Baskaran (2011a). "Performance of Video Conferencing in Unicast and Multicast Communication using Protocol Independent Multicast Routing". In: *International Journal of Computer Science and Telecommunications* 2 (9).

Chandrasekar, V and K Baskaran (2011b). "Performance of video conferencing in unicast and multicast communication using protocol independent multicast routing". In: *International Journal of Computer Science and Telecommunications* 2.

— (2012). "Performance of Video Conferencing using Protocol Independent Multicast Routing with Core failure". In: *International Journal of Computer Applications (0975–8887) Volume*.

Chaubey, Nishant and Aditya Trivedi (2014). "Analysis and design of decentralized conferencing using Wi-Fi based on P2P architecture". In: *Computer and Communication Technology (ICCCT), 2014 International Conference on*. IEEE, pp. 233–239.

Chen, Shigang and Klara Nahrstedt (1998). "An overview of quality of service routing for next-generation high-speed networks: problems and solutions". In: *IEEE network* 12.6, pp. 64–79.

Chowdhury, Iffat Sharmin, Jutheka Lahiry, and Syed Faisal Hasan (2009). "Performance analysis of datagram congestion control protocol (DCCP)". In: *Computers and Information Technology, 2009. ICCIT'09. 12th International Conference on*. IEEE, pp. 454–459.

Civanlar, Seyhan, Murat Parlakisik, A Murat Tekalp, Burak Gorkemli, Bulent Kaytaz, and Evren Onem (2010). "A QoS-enabled openflow environment for scalable video streaming". In: *GLOBECOM Workshops (GC Wkshps), 2010 IEEE*. IEEE, pp. 351–356.

Coddington, Carl D, Bernard J Craig, Larry A Litteral, Arthur A Richard III, Jeffrey B Gold, Donald C Klika Jr, Daniel B Konkle, and James M McHenry (1995). *Video-on-demand services using public switched telephone network*. US Patent 5,410,343.

Cofano, Giuseppe, Luca De Cicco, Thomas Zinner, Anh Nguyen-Ngoc, Phuoc Tran-Gia, and Saverio Mascolo (2016). "Design and experimental evaluation of network-assisted strategies for HTTP adaptive streaming". In: *Proceedings of the 7th International Conference on Multimedia Systems*. ACM, p. 3.

Cuervo, Fernando, Nancy Greene, A Rayhan, C Huitema, Brian Rosen, and John Segers (2000). *Megaco protocol version 1.0*. Tech. rep.

Curtis, Andrew R, Jeffrey C Mogul, Jean Tourrilhes, Praveen Yalagandula, Puneet Sharma, and Sujata Banerjee (2011). "DevoFlow: scaling flow management for high-performance networks". In: *ACM SIGCOMM Computer Communication Review*. Vol. 41. 4. ACM, pp. 254–265.

De Amorim, Marcelo Dias, OCMB Duarte, and Guy Pujolle (1999). "Single-loop packet merging for receiver-oriented multicast multi-layered video". In: *PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON COMPUTER COMMUNICATION*. 2, pp. I–3.

De Cicco, Luca, Saverio Mascolo, and Vittorio Palmisano (2008). "Skype Video Responsiveness to Bandwidth Variations". In: *18th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV) (2008)*. ACM, pp. 81–86.

Deering, Stephen E (1988). "Host extensions for IP multicasting". In:

Deering, Stephen E and David R Cheriton (1990). "Multicast routing in datagram internetworks and extended LANs". In: *ACM Transactions on Computer Systems (TOCS)* 8.2, pp. 85–110.

Deshpande, Madhura and SP Mohani (2015). "Integration of WebRTC with SIP–Current Trends". In: *Int. J. Innov. Eng. Technol. Integr.* 6.2, pp. 92–96.

Detti, Andrea, Giuseppe Bianchi, Claudio Pisa, Francesco Saverio Proto, Pierpaolo Loreti, Wolfgang Kellerer, Srisakul Thakolsri, and Joerg Widmer (2009). "SVEF: an open-source experimental evaluation framework for H. 264 scalable video streaming". In: *Computers and Communications, 2009. ISCC 2009. IEEE Symposium on*. IEEE, pp. 36–41.

Ding, Zhuang, Jun Wu, Wei Yu, Yuqi Han, and Xianghuang Chen (2016). "Pseudo analog video transmission based on LTE physical layer". In: *IEEE/CIC International Conference on Communications in China (2016)*. IEEE, pp. 1–6.

Diot, Christophe, Brian Neil Levine, Bryan Lyles, Hassan Kassem, and Doug Balensiefen (2000). "Deployment issues for the IP multicast service and architecture". In: *IEEE network* 14.1, pp. 78–88.

Doria, Avri, Fiffi Hellstrand, Kenneth Sundell, and Tom Worster (2002). *General switch management protocol (GSMP) V3*. Tech. rep.

Dunigan, Tom, F Fowler, *et al.* (2004). "Almost TCP over UDP (atou)". In: *last modified Jan* 12.

Edan, Naktal Moaid, Ali Al-Sherbaz, and Scott Turner (2017). "WebNSM: A Novel Scalable WebRTC Signalling Mechanism for Many-to-Many Video Conferencing". In: *Collaboration and Internet Computing (CIC), 2017 IEEE 3rd International Conference on*. IEEE, pp. 27–33.

Egilmez, Hilmi E and A Murat Tekalp (2014). "Distributed QoS architectures for multimedia streaming over software defined networks". In: *IEEE Transactions on Multimedia* 16.6, pp. 1597–1609.

Egilmez, Hilmi E, S Tahsin Dane, K Tolga Bagci, and A Murat Tekalp (2012). "OpenQoS: An OpenFlow controller design for multimedia delivery with end-to-end Quality of Service over Software-Defined Networks". In: *Signal & Information processing association annual summit and conference (APSIPA ASC), 2012 Asia-Pacific*. IEEE, pp. 1–8.

Egilmez, Hilmi E, Seyhan Civanlar, and A Murat Tekalp (2013). "An optimization framework for QoS-enabled adaptive video streaming over OpenFlow networks". In: *IEEE Transactions on Multimedia* 15.3, pp. 710–715.

Elleuch, W. and A.C. Houle (2008). "Multiparty Voice over IP (MVoIP) Peer-Based System for Large-Scale Conference Support". In: *IEEE International Conference on Wireless and Mobile Computing*, pp. 415–416.

Elleuch, Wajdi (2013). "Models for multimedia conference between browsers based on WebRTC". In: *Wireless and Mobile Computing, Networking and Communications (WiMob), 2013 IEEE 9th International Conference on*. IEEE, pp. 279–284.

Erdös, Paul and Alfréd Rényi (1959). "On Random Graphs I." In: *Publicationes Mathematicae* 6, pp. 290–297.

Estrin, Deborah, Dino Farinacci, Ahmed Helmy, David Thaler, Stephen Deering, Mark Handley, Van Jacobson, Ching-Gung Liu, Puneet Sharma, and Liming Wei (1998). *Protocol independent multicast-sparse mode (PIM-SM): Protocol specification*. Tech. rep.

Fan, Fujie, Bing Hu, and Kwan L Yeung (2016). "Distributed and dynamic multicast scheduling in fat-tree data center networks". In: *Communications (ICC), 2016 IEEE International Conference on*. IEEE, pp. 1–6.

Farinacci, Dino, C Liu, S Deering, D Estrin, M Handley, Van Jacobson, L Wei, Puneet Sharma, David Thaler, and A Helmy (1998). "Protocol independent multicast-sparse mode (PIM-SM): Protocol specification". In:

Fenner, William C (1997). "Internet group management protocol, version 2". In:

Fiedler, Markus, Tobias Hossfeld, and Phuoc Tran-Gia (2010). "A generic quantitative relationship between quality of experience and quality of service". In: *IEEE Network* 24.2.

Fraczek, Wojciech, Wojciech Mazurczyk, and Krzysztof Szczypiorski (2010). "Stream Control Transmission Protocol Steganography". In: *arXiv preprint arXiv:1006.0247*.

Friedman, Timur, Ramon Caceres, and Alan Clark (2003). *RTP control protocol extended reports (RTCP XR)*. Tech. rep.

Garfinkel, Simson L (2005). "VoIP and Skype security". In: *Tactical Technology Collective* 12.

Georgopoulos, Panagiotis, Yehia Elkhatib, Matthew Broadbent, Mu Mu, and Nicholas Race (2013). "Towards network-wide QoE fairness using openflow-assisted adaptive video streaming". In: *Proceedings of the 2013 ACM SIGCOMM workshop on Future human-centric multimedia networking*. ACM, pp. 15–20.

Glabowski, Mariusz, Slawomir Hanczewski, and Maciej Stasiak (2007). "Calculation of available bandwidth for umts-hsdpa/hsupa users". In: *The International Conference on "Computer as a Tool"*, pp. 1145–1152.

Glasmann, Josef, Wolfgang Kellerer, and Harald Muller (2003). "Service architectures in H. 323 and SIP: A comparison". In: *IEEE Communications Surveys & Tutorials* 5.2.

Greenberg, Albert, Gisli Hjalmtysson, David A Maltz, Andy Myers, Jennifer Rexford, Geoffrey Xie, Hong Yan, Jibin Zhan, and Hui Zhang (2005). "A clean slate 4D approach to network control and management". In: *ACM SIGCOMM Computer Communication Review* 35.5, pp. 41–54.

Group, Network Working *et al.* (2003). "RTP: A transport protocol for real-time applications". In: *RFC3550*.

Gu, Weidong, Xinchang Zhang, Bin Gong, and Lu Wang (2015). "A survey of multicast in software-defined networking". In: *Proc. 5th Int. Conf. Inf. Eng. Mech. Mater.(ICIMM)*, pp. 1096–1100.

Guha, Saikat and Neil Daswani (2005). *An experimental study of the skype peer-to-peer voip system*. Tech. rep. Cornell University.

Gupta, Anupam and Amit Kumar (2014). "Online steiner tree with deletions". In: *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, pp. 455–467.

Handley, Mark, Jon Crowcroft, Carsten Bormann, and J Ott (1997). *The internet multimedia conferencing architecture*. Tech. rep. Internet Draft, Internet Engineering Task Force.

Handley, Mark, Isidor Kouvelas, Tony Speakman, and Lorenzo Vicisano (2007). *Bidirectional protocol independent multicast (BIDIR-PIM)*. Tech. rep.

He, Dongliang, Cuiling Lan, Chong Luo, Enhong Chen, Feng Wu, and Wenjun Zeng (2017). "Progressive pseudo-analog transmission for mobile video streaming". In: *IEEE Transactions on Multimedia* 19.8, pp. 1894–1907.

Heller, Brandon (2009). "OpenFlow Switch Specification". In:

Hoeldtke, K. and A. Raake (2011). "Conversation Analysis of Multi-Party Conferencing and Its Relation to Perceived Quality". In: *IEEE International Conference on Communications*, pp. 1–5.

Hoerdt, Mickaël and Damien Magoni (2005). "Cartographie distribuée du coeur de l'Internet". In: *Ann. des Télécom.* 60.5-6, pp. 558–587.

Hosseini, Mojtaba, Dewan Tanvir Ahmed, Shervin Shirmohammadi, and Nicolas D Georganas (2007a). "A survey of application-layer multicast protocols". In: *IEEE Communications Surveys & Tutorials* 9.3, pp. 58–74.

— (2007b). "A survey of application-layer multicast protocols." In: *IEEE Communications Surveys and Tutorials* 9.1-4, pp. 58–74.

Humernbrum, Tim, Bastian Hagedorn, and Sergei Gorlatch (2016). "Towards efficient multicast communication in software-defined networks". In: *Distributed Computing Systems Workshops (ICDCSW), 2016 IEEE 36th International Conference on*. IEEE, pp. 106–113.

Jabbar, Waheb A, Mahamod Ismail, and Rosdiadee Nordin (2013). "Peer-to-peer communication on android-based mobile devices: Middleware and protocols". In: *Modeling, Simulation and Applied Optimization (ICMSAO), 2013 5th International Conference on*. IEEE, pp. 1–6.

Jacobson, Van and Mark Handley (1998). "SDP: session description protocol". In:

Jacobson, Van, Ron Frederick, Steve Casner, and H Schulzrinne (2003). "RTP: A transport protocol for real-time applications". In:

Jan Willem, Kleinrouweler, Cabrero Sergio, and César Pablo (2016). "Delivering stable high-quality video: an SDN architecture with DASH assisting network elements". In: *MMSys*.

Jarschel, Michael, Simon Oechsner, Daniel Schlosser, Rastin Pries, Sebastian Goll, and Phuoc Tran-Gia (2011). "Modeling and performance evaluation of an OpenFlow architecture". In: *Proceedings of the 23rd international teletraffic congress*. International Teletraffic Congress, pp. 1–7.

Jennings, Cullen, Ted Hardie, and Magnus Westerlund (2013). "Real-time communications for the web". In: *IEEE Communications Magazine* 51.4, pp. 20–26.

Jivorasetkul, S., M. Shimamura, and K. Iida (2013). "Better network latency with end-to-end header compression in SDN architecture". In: *IEEE Pacific Rim Conference on Communications, Computers and Signal Processing*, pp. 183–188.

Kang, Nanxi, Zhenming Liu, Jennifer Rexford, and David Walker (2013). "Optimizing the one big switch abstraction in software-defined networks". In: *Proceedings of the ninth ACM conference on Emerging networking experiments and technologies*. ACM, pp. 13–24.

Kanizo, Yossi, David Hay, and Isaac Keslassy (2013). "Palette: Distributing tables in software-defined networks". In: *INFOCOM, 2013 Proceedings IEEE*. IEEE, pp. 545–549.

Karaman, M.A., B. Gorkemli, S. Tatlicioglu, M. Komurcuoglu, and O. Karakaya (2015). "Quality of service control and resource prioritization with Software Defined Networking". In: *1st IEEE Conference on Network Softwarization*, pp. 1–6.

Kaur, Gaganpreet and Parmjit Kaur (2015). "Open Source VoIP Standards: A Review". In: *International Journal of Exploring Emerging Trends in Engineering (IJEETE) Vol* 2.

Kbar, Ghassan, Wathiq Mansoor, and Aryan Naim (2010). "Voice over IP mobile telephony using WIFI P2P". In: *Wireless and Mobile Communications (ICWMC), 2010 6th International Conference on*. IEEE, pp. 268–273.

Kim, Wonho, Puneet Sharma, Jeongkeun Lee, Sujata Banerjee, Jean Tourrilhes, Sung-Ju Lee, and Praveen Yalagandula (2010). "Automated and Scalable QoS Control for Network Convergence." In: *INM/WREN* 10.1, pp. 1–1.

Klauck, Ronny and Michael Kirsche (2009). "Integrating P2PSIP into collaborative P2P applications: A case study with the P2P videoconferencing system BRAVIS". In: *Collaborative Computing: Networking, Applications and Worksharing, 2009. CollaborateCom 2009. 5th International Conference on*. IEEE, pp. 1–10.

Klaue, Jirka, Berthold Rathke, and Adam Wolisz (2003). "Evalvid–A framework for video transmission and quality evaluation". In: *International conference on modelling techniques and tools for computer performance evaluation*. Springer, pp. 255–272.

Kohler, Eddie, Mark Handley, and Sally Floyd (2006). *Datagram congestion control protocol (DCCP)*. Tech. rep.

Kuhn, D Richard (1997). "Sources of failure in the public switched telephone network". In: *Computer* 30.4, pp. 31–36.

Kumar, H., H.H. Gharakheili, and V. Sivaraman (2013). "User control of quality of experience in home networks using SDN". In: *IEEE International Conference on Advanced Networks and Telecommuncations Systems*, pp. 1–6.

Laga, Sebastiaan, Thomas Van Cleemput, Filip Van Raemdonck, Felix Vanhoutte, Niels Bouten, Maxim Claeys, and Filip De Turck (2014). "Optimizing scalable video delivery through OpenFlow layer-based routing". In: *Network Operations and Management Symposium (NOMS)*. IEEE, pp. 1–4.

Lahbabi, Youssef, Ahmed Hammouch, *et al.* (2014). "Quality adaptation using scalable video coding (SVC) in peer-to-peer (P2P) video-on-demand (VoD) streaming". In: *Multimedia Computing and Systems (ICMCS), 2014 International Conference on*. IEEE, pp. 1140–1146.

Laner, Markus, Philipp Svoboda, and Markus Rupp (2012). "Latency analysis of 3g network components". In: *18th European Wireless Conference (EW)*, pp. 1–8.

Li, Jin, Philip A Chou, and Cha Zhang (2004). "Mutualcast: An efficient mechanism for content distribution in a peer-to-peer (p2p) network". In: *Microsoft Research, MSR-TR-2004* 100.

Liang, Chao, M Zhao, and Y Liu (2011). "Optimal resource allocation in multisource multi-swarm P2P video conferencing swarms". In: *in IEEE/ACM Trans. on Networking*.

Lien, Yao-Nan and Yu-Chi Ding (2011). "Can DCCP Replace UDP in Changing Network Conditions?" In: *Advanced Information Networking and Applications (AINA), 2011 IEEE International Conference on*. IEEE, pp. 716–723.

Luo, Chong, Wei Wang, Jian Tang, Jun Sun, and Jiang Li (2007). "A multiparty videoconferencing system over an application-level multicast protocol". In: *IEEE Transactions on Multimedia* 9.8, pp. 1621–1632.

Maenpaa, Jouni (2013). "Reducing P2PSIP session setup delays". In: *2013 International Conference on Computing, Networking and Communications (ICNC)*. IEEE, pp. 872–878.

Magoni, Damien and Jean-Jacques Pansiot (2002). "Internet Topology Modeler Based on Map Sampling". In: *7th IEEE Symposium on Computers and Communications (ISCC)*, pp. 1021–1027.

Mahy, Rohan, Philip Matthews, and Jonathan Rosenberg (2010). *Traversal using relays around nat (turn): Relay extensions to session traversal utilities for nat (stun)*. Tech. rep.

Mani, S.K., P. Bala Manikya Prasad, M. Kamesh, and N. Mani (2009). "DSP subsystem for multiparty conferencing in VoIP". In: *IEEE International Conference on Internet Multimedia Services Architecture and Applications*, pp. 1–6.

McKeown, Nick, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner (2008). "OpenFlow: enabling innovation in campus networks". In: *ACM SIGCOMM Computer Communication Review* 38.2, pp. 69–74.

Merwe, Jacobus E van der and Ian M Leslie (1997). "Switchlets and dynamic virtual ATM networks". In: *Integrated Network Management V*. Springer, pp. 355–368.

Mhatarmare, Vaishali R and AD Raut (2013). "Mobile Telephony Over Wi-Fi Using VOIP". In: *International Journal of Advanced Research in Computer Science* 4.6.

Mukherjee, Debargha, Jim Bankoski, Adrian Grange, Jingning Han, John Koleszar, Paul Wilkins, Yaowu Xu, and Ronald Bultje (2013). "The latest open-source video codec VP9-an overview and preliminary results". In: *Picture Coding Symposium (PCS), 2013*. IEEE, pp. 390–393.

Munther, Alhamza, Salah Noori, Azlan Osman, Ayad Hussain, Imad Jasim, and Ali Shanoon (2012). "Peer-to-peer video conferencing using hybrid content distribution model". In: *Journal of Computer Science* 8.7, p. 1134.

Nicholas, Jonathan, William Siadak, and Andrew Adams (2005). "Protocol independent multicast-dense mode (pim-dm): protocol specification (revised)". In:

Noghani, Kyoomars Alizadeh and M Oguz Sunay (2014). "Streaming multicast video over software-defined networks". In: *Mobile Ad Hoc and Sensor Systems (MASS), 2014 IEEE 11th International Conference on*. IEEE, pp. 551–556.

Noll, A Michael (1992). "Anatomy of a failure: Picturephone revisited". In: *Telecommunications policy* 16.4, pp. 307–316.

Nor, Shahrudin Awang, Suhaidi Hassan, Osman Ghazali, and Mohd Hasbullah Omar (2012). "Enhancing DCCP congestion control mechanism for long delay link". In: *Telecommunication Technologies (ISTT), 2012 International Symposium on*. IEEE, pp. 313–318.

Nor, Shahrudin Awang, Raaid Alubady, and Wisam Abduladeem Kamil (2017). "Simulated performance of TCP, SCTP, DCCP and UDP protocols over 4G network". In: *Procedia computer science* 111, pp. 2–7.

Nurminen, Jukka K, Antony JR Meyn, Eetu Jalonen, Yrjo Raivio, and Raúl Garcıa Marrero (2013). "P2P media streaming with HTML5 and WebRTC". In: *Computer Communications Workshops (INFOCOM WKSHPS), 2013 IEEE Conference on*. IEEE, pp. 63–64.

Ogirima, Sanni Abubakar Omuya, Adeolu .O Afolabi, Abimbola Adebisi Adigun, and Noah Oluwatobi Akande (2014). "Development of Campus Video-Conference System Based on Peer-To-Peer Architecture". In: *https://pdfs.semanticscholar.org/eab0/d3c1bad9f229c0ad32afb435b7f0829e71ee.pdf*.

Oliveira, Francisco, Eduardo Tavares, Erica Sousa, and Bruno Nogueira (2018). "Video Conferencing Evaluation Considering Scalable Video Coding and SDN Network". In: *Revista de Informática Teórica e Aplicada* 25.2, pp. 38–46.

Pendarakis, Dimitrios, Sherlia Shi, Dinesh Verma, and Marcel Waldvogel (2001). "ALMI: An Application Level Multicast Infrastructure". In: *3rd Conference on USENIX Symposium on Internet Technologies and Systems*, pp. 49–60.

Perros, Harry and Khaled Elsayed (1996). "Call admission control schemes: a review". In: *IEEE Communications Magazine* 34.11, pp. 82–91.

Ponec, Miroslav, Sudipta Sengupta, Minghua Chen, Jin Li, and Philip A Chou (2009). "Multi-rate peer-to-peer video conferencing: A distributed approach using scalable coding". In: *Multimedia and Expo, 2009. ICME 2009. IEEE International Conference on*. IEEE, pp. 1406–1413.

Postel, Jon (1980). *User datagram protocol*. Tech. rep.

— (1981). *Transmission control protocol*. Tech. rep.

Prasad, R Venkatesha, HS Jamadagni, and HN Shankar (2003). "On the problem of specifying the number of floors for a voice-only conference on packet networks". In: *Information Technology: Research and Education, 2003. Proceedings. ITRE2003. International Conference on*. IEEE, pp. 22–26.

Rahaman, Md Habibur (2015). "A Survey on Real-Time Communication for Web". In: *Scientific Research Journal (Scirj)* 3.VII, pp. 39–45.

Rahman, Joy, Sajeeb Saha, and Syed Faisal Hasan (2012). "A new congestion control algorithm for datagram congestion control protocol (DCCP) based real-time multimedia applications". In: *Electrical & Computer Engineering (ICECE), 2012 7th International Conference on*. IEEE, pp. 533–536.

Ramadass, Sureswaran (2010). *Multimedia Conferencing System: An Innovative System for Multimedia Conferencing*. VDM Verlag.

Rämö, Anssi and Henri Toukomaa (2011). "Voice quality characterization of IETF Opus codec". In: *Twelfth Annual Conference of the International Speech Communication Association*.

Rec, ITUT (2008). "G. 711.1,"Wideband embedded extension for G. 711 pulse code modulation"". In: *Int. Telecomm. Union, Geneva*.

Recommendation, H (1998). "323, Packet-based multimedia communications systems". In: *ITU-T (December 2009)*.

Robins, Gabriel and Alexander Zelikovsky (2000). "Improved Steiner tree approximation in graphs." In: *SODA*. Citeseer, pp. 770–779.

Rodríguez, Pedro, Álvaro Alonso, Joaquín Salvachúa, and Javier Cerviño (2016). "Materialising a new architecture for a distributed MCU in the Cloud". In: *Computer Standards & Interfaces* 44, pp. 234–242.

Rosenberg, Jonathan, Henning Schulzrinne, Gonzalo Camarillo, Alan Johnston, Jon Peterson, Robert Sparks, Mark Handley, and Eve Schooler (2002). *SIP: session initiation protocol*. Tech. rep.

Rückert, Julius, Jeremias Blendin, Rhaban Hark, Timm Wächter, and David Hausheer (2015). "An Extended Study of DYNSDM: Software-Defined Multicast using Multi-Trees". In: *Peer-to-Peer Systems Engineering Lab, TU Darmstadt, Germany, Tech. Rep*.

Sahasrabuddhe, Laxman H and Biswanath Mukherjee (2000). "Multicast routing algorithms and protocols: A tutorial". In: *IEEE network* 14.1, pp. 90–102.

Salami, Redwan, Claude Laflamme, Bruno Bessette, and J-P Adoul (1997). "ITU-T G. 729 Annex A: reduced complexity 8 kb/s CS-ACELP codec for digital simultaneous voice and data". In: *IEEE Communications Magazine* 35.9, pp. 56–63.

Saldana, J., F. Pascual, D. de Hoz, J. Fernandez-Navajas, J. Ruiz-Mas, D.R. Lopez, D. Florez, J.A. Castell, and M. Nunez (2014). "Optimization of

low-efficiency traffic in OpenFlow Software Defined Networks". In: *International Symposium on Performance Evaluation of Computer and Telecommunication Systems*, pp. 550–555.

Sat, B., Zixia Huang, and B. Wah (2007). "The Design of a Multi-party VoIP Conferencing System over the Internet". In: *9th IEEE International Symposium on Mutimedia*, pp. 3–10.

Schier, Michael and Michael Welzl (2012). "Using DCCP: Issues and improvements". In: *Network Protocols (ICNP), 2012 20th IEEE International Conference on*. IEEE, pp. 1–9.

Schulzrinne, Henning and Stephen Casner (2003). *RTP profile for audio and video conferences with minimal control*. Tech. rep.

Schulzrinne, Henning, Anup Rao, and Robert Lanphier (1998). *Real time streaming protocol (RTSP)*. Tech. rep.

Schuster, Guido, Ikhlaq Sidhu, Michael Borella, and Jacek Grabiec (2002). *System for dynamic jitter buffer management based on synchronized clocks*. US Patent 6,360,271.

Schwarz, Heiko, Detlev Marpe, and Thomas Wiegand (2007). "Overview of the scalable video coding extension of the H. 264/AVC standard". In: *IEEE Transactions on circuits and systems for video technology* 17.9, pp. 1103–1120.

Segeč, Pavel, Peter Palúch, Jozef Papán, and Milan Kubina (2014). "The integration of WebRTC and SIP: way of enhancing real-time, interactive multimedia communication". In: *Emerging eLearning Technologies and Applications (ICETA), 2014 IEEE 12th International Conference on*. IEEE, pp. 437–442.

Sezer, Sakir, Sandra Scott-Hayward, Pushpinder Kaur Chouhan, Barbara Fraser, David Lake, Jim Finnegan, Niel Viljoen, Marc Miller, and Navneet Rao (2013). "Are we ready for SDN? Implementation challenges for software-defined networks". In: *IEEE Communications Magazine* 51.7, pp. 36–43.

Shaffer, Shmuel and Neufito Fernandes (1998). *Collaborative conference bridges*. US Patent 5,825,858.

Sieber, Christian, Andreas Blenk, David Hock, Marc Scheib, Thomas Hohn, Stefan Kohler, and Wolfgang Kellerer (2015). "Network configuration with quality of service abstractions for SDN and legacy networks". In: *IFIP/IEEE International Symposium on Integrated Network Management*, pp. 1135–1136.

Stallings, William (1989). *ISDN: an introduction*. Macmillan New York.

Sullivan, Gary J, Jens Ohm, Woo-Jin Han, and Thomas Wiegand (2012). "Overview of the high efficiency video coding (HEVC) standard". In: *IEEE Transactions on circuits and systems for video technology* 22.12, pp. 1649–1668.

Takeuchi, Shigeki, Hiroyuki Koga, Katsuyoshi Iida, Youki Kadobayashi, and Suguru Yamaguchi (2005). "Performance evaluations of dccp for bursty traffic in real-time applications". In: *null*. IEEE, pp. 142–149.

Tang, Siyuan, Bei Hua, and Dongyang Wang (2014). "Realizing video streaming multicast over SDN networks". In: *Communications and Networking in China (CHINACOM), 2014 9th International Conference on*. IEEE, pp. 90–95.

Topiwala, Pankaj, Wei Dai, Madhu Krishnan, Adeel Abbas, Sandeep Doshi, and David Newman (2017). "Performance comparison of AV1, HEVC, and JVET video codecs on 360 (spherical) video". In: *Applications of Digital Image Processing XL*. Vol. 10396. International Society for Optics and Photonics, p. 1039609.

Tran, Duc A, Kien A Hua, and Tai T Do (2004). "A peer-to-peer architecture for media streaming". In: *IEEE Journal on Selected Areas in Communications (JSAC)* 22.1, pp. 121–133.

Tripathi, Saurabh, Shaurya Gupta, Tushar Babbar, and Vishal Gupta (2013). "TCP-over-UDP for Real Time Applications". In: *International Journal of Scientific & Engineering Research-IJSER* 4, pp. 357–360.

Uhl, Tadeus (2004). "Quality of service in VoIP communication". In: *AEU-International Journal of Electronics and Communications* 58.3, pp. 178–182.

Valin, Jean-Marc, Timothy B Terriberry, Christopher Montgomery, and Gregory Maxwell (2010). "A High-Quality Speech and Audio Codec With Less Than 10-ms Delay." In: *IEEE Trans. Audio, Speech & Language Processing* 18.1, pp. 58–67.

Valin, Jean-Marc, Koen Vos, and Timothy Terriberry (2012). *Definition of the Opus audio codec*. Tech. rep.

Valin, Jean-Marc, Gregory Maxwell, Timothy B Terriberry, and Koen Vos (2016). "High-quality, low-delay music coding in the opus codec". In: *arXiv preprint arXiv:1602.04845*.

Van Adrichem, Niels LM, Christian Doerr, and Fernando A Kuipers (2014). "Opennetmon: Network monitoring in openflow software-defined networks". In: *Network Operations and Management Symposium (NOMS), 2014 IEEE*. IEEE, pp. 1–8.

Varga, Imre, RD De Lacovo, and Paolo Usai (2006). "Standardization of the AMR wideband speech codec in 3GPP and ITU-T". In: *IEEE Communications Magazine* 44.5, pp. 66–73.

Varshney, Upkar, Andy Snow, Matt McGivern, and Christi Howard (2002). "Voice over IP". In: *Communications of the ACM* 45.1, pp. 89–96.

Vos, Koen, Soeren Jensen, and Karsten Soerensen (2010). "SILK speech codec". In: *IETF draft*.

Vutukuru, Mythili, Hari Balakrishnan, and Kyle Jamieson (2009). "Cross-layer wireless bit rate adaptation". In: *ACM SIGCOMM Computer Communication Review* 39.4, pp. 3–14.

Wang, Wenpeng and Lingli Mei (2017). "A design of multimedia conferencing system based on WebRTC Technology". In: *Information Technology, Electronics and Mobile Communication Conference (IEMCON), 2017 8th IEEE Annual*. IEEE, pp. 148–153.

Watzlaf, Valerie JM, Sohrab Moeini, and Patti Firouzan (2010). "VoIP for telerehabilitation: A risk analysis for privacy, security, and HIPAA compliance". In: *International Journal of Telerehabilitation* 2.2, p. 3.

Waxman, Bernard (1988). "Routing of Multipoint Connections". In: *IEEE Journal on Selected Areas in Communications* 6.9, pp. 1617–1622.

Wiegand, Thomas, Gary J Sullivan, Gisle Bjontegaard, and Ajay Luthra (2003). "Overview of the H. 264/AVC video coding standard". In: *IEEE Transactions on circuits and systems for video technology* 13.7, pp. 560–576.

Willebeek-LeMair, MH, Dilip D Kandlur, and Z-Y Shae (1994). "On multipoint control units for videoconferencing". In: *19th Conference on Local Computer Networks (LCN)*, pp. 356–364.

Winter, Pawel (1987). "Steiner problem in networks: a survey". In: *Networks* 17.2, pp. 129–167.

Wu, Dapeng, Yiwei Thomas Hou, Wenwu Zhu, Ya-Qin Zhang, and Jon M Peha (2001). "Streaming video over the Internet: approaches and directions". In: *IEEE Transactions on circuits and systems for video technology (TCSVT)* 11.3, pp. 282–300.

Xu, Xun, Li-Wei He, D. Florencio, and Yong Rui (2006). "PASS: Peer-Aware Silence Suppression for Internet Voice Conferences". In: *IEEE International Conference on Multimedia and Expo*, pp. 2149–2152.

Xu, Yang, Chenguang Yu, Jingjiang Li, and Yong Liu (2012). "Video telephony for end-consumers: measurement study of Google+, iChat, and Skype". In: *Proceedings of the 2012 Internet Measurement Conference*. ACM, pp. 371–384.

Xue, Nana, Xiaoliang Chen, Long Gong, Suoheng Li, Daoyun Hu, and Zuqing Zhu (2015). "Demonstration of OpenFlow-controlled network orchestration for adaptive SVC video manycast". In: *IEEE Transactions on Multimedia* 17.9, pp. 1617–1629.

Yang, En-Zhong, Lin-Kai Zhang, Zhen Yao, and Jian Yang (2016). "A video conferencing system based on SDN-enabled SVC multicast". In: *Frontiers of Information Technology & Electronic Engineering* 17.7, pp. 672–681.

Yang, Jian, Enzhong Yang, Yongyi Ran, and Shuangwu Chen (2015). "SDMˆ 2 Cast An OpenFlow-Based, Software-Defined Scalable Multimedia Multicast Streaming Framework". In: *IEEE Internet Computing* 19.4, pp. 36–44.

Yang, Jian, Kunjie Zhu, Yongyi Ran, Weizhe Cai, and Enzhong Yang (2017). "Joint Admission control and routing via approximate dynamic programming for streaming video over software-defined networking". In: *IEEE Transactions on Multimedia* 19.3, pp. 619–631.

Yang, Jian, Enzhong Yang, Yongyi Ran, Yifeng Bi, and Jun Wang (2018). "Controllable Multicast for Adaptive Scalable Video Streaming in Software-Defined Networks". In: *IEEE Transactions on Multimedia* 20.5, pp. 1260–1274.

Yeganeh, Soheil Hassas, Amin Tootoonchian, and Yashar Ganjali (2013). "On scalability of software-defined networking". In: *IEEE Communications Magazine* 51.2, pp. 136–141.

Yu, Linchen (2012). "Improving query for P2P SIP VoIP". In: *Trust, Security and Privacy in Computing and Communications (TrustCom), 2012 IEEE 11th International Conference on*. IEEE, pp. 1735–1740.

Yu, Minlan, Jennifer Rexford, Michael J Freedman, and Jia Wang (2010). "Scalable flow-based networking with DIFANE". In: *ACM SIGCOMM Computer Communication Review* 40.4, pp. 351–362.

Yu, Xiaoyan and Zhenyu Yu (2012). "A Distributed Architecture of Video Conference Using P2P Technology". In: *Journal of Networks* 7.11, p. 1852.

Yu, Yun-Shuai and Chih-Heng Ke (2018). "Genetic algorithm-based routing method for enhanced video delivery over software defined networks". In: *International Journal of Communication Systems* 31.1.

Zeidan, Adham, Armin Lehmann, and Ulrich Trick (2014). "WebRTC enabled multimedia conferencing and collaboration solution". In: *WTC 2014; World Telecommunications Congress 2014; Proceedings of*. VDE, pp. 1–6.

Zhao, Miao, Bin Jia, Mingquan Wu, Heather Yu, and Yang Xu (2014). "Software defined network-enabled multicast for multi-party video conferencing systems". In: *IEEE International Conference on Communications (ICC) (2014)*, pp. 1729–1735.

Abstract

Adaptive Multicast Live Streaming
for A/V Conferencing Systems
over Software-Defined Networks

Real-time applications, such as multi-party conferencing systems, have strong Quality of Service requirements for ensuring a decent Quality of Experience. Nowadays, most of these conferences are performed on wireless devices. Thus, mobile devices' heterogeneity and network dynamic, especially the access points' bandwidth, must be properly managed to provide a good Quality of Experience.

In this thesis, we propose several algorithms for building and maintaining conference sessions based on Software-Defined Networks. These algorithms use both multicast distribution and bitrate adaptation. Our algorithms operate in two main steps. The first step consists in configuring the conference call by building multicast trees. The second step consists in optimally placing the bitrate adaptation rules inside the network in order to minimize the bandwidth consumption. We focus on two goals: latency minimization, by building shortest path trees, and bandwidth consumption minimization, by building trees of minimizing weight. Afterward, we address the issue of network dynamics, especially bandwidth variations occurring during a call. We provide an algorithm that optimally relocates the bitrate adaptation rules without rebuilding the multicast trees. It requires very low computation at the controller, thus making it fast and highly reactive. Extensive simulation results confirm the efficiency of our solution in terms of processing time and bandwidth savings, compared to existing conferencing systems based on a Multipoint Control Unit or on Application Layer Multicast.

Résumé

Diffusion multipoint adaptable pour les systèmes de télé- et visio-conférences déployés sur des réseaux à définition logicielle

Les applications en temps réel, telles que les systèmes de conférence multi-utilisateurs, ont des exigences de Qualité de Service élevées pour garantir une Qualité d'Expérience optimale. De nos jours, la plupart de ces conférences sont effectuées sur des appareils sans fil. Ainsi, l'hétérogénéité des appareils mobiles, et la dynamique du réseau, particulièrement en ce qui concerne la bande passante des points d'accès, doivent être correctement gérés pour fournir une bonne Qualité d'Expérience.

Dans cette thèse, nous proposons des algorithmes pour construire et gérer des sessions de conférences basées sur des réseaux à définition logicielle (*Software Defined Network, SDN*). Ces algorithmes utilisent à la fois la distribution multipoint et l'adaptation du débit en fonctions des caractéristiques des participants. Nos algorithmes sont constitués de deux étapes principales. La première est la configuration de la conférence audio/vidéo en créant des arbres de diffusion multipoint permettant la communication. La deuxième est le placement de manière optimale des règles d'adaptation du débit dans le réseau. Nous nous concentrons particulièrement sur deux objectifs : le premier est la minimisation de la latence en construisant l'arbre des plus courts chemins, le deuxième est la minimisation de la consommation de bande passante dans le réseau, et ce en construisant des arbres de poids minimum. Dans un deuxième temps, nous nous intéressons à la dynamique du réseau, particulièrement aux variations de la bande passante des points d'accès pendant une conférence. Nous proposons un algorithme qui replace de manière optimale les règles d'adaptation de débit sans reconstruire les arbres multipoints. Cela occasionne un calcul très faible au niveau du contrôleur, ce qui rend notre solution rapide et hautement réactive. Les résultats de simulation confirment l'efficacité de notre solution en termes de temps de traitement et d'économie de bande passante par rapport aux systèmes de conférence existants basés soit sur une unité de contrôle multipoint, soit sur une diffusion multipoint au niveau de la couche application.