N° 4662

UNIVERSITÉ BORDEAUX 1

ÉCOLE DOCTORALE DE MATHÉMATIQUES ET D'INFORMATIQUE

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ BORDEAUX 1

Discipline : **INFORMATIQUE**

par

Abbas Bradai

TITLE

Scheduling and Bandwidth Allocation in P2P Layered Streaming Systems

Titre: Ordonnancement et Allocation de Bande Passante dans les Systèmes de Streaming Pair-à-Pair Multicouches

Sous la direction de : Toufik Ahmed, Professeur des Universités

Devant le jury composé de:

Pr Toufik AHMED	Professeur, ENSEIRB-MATMECA – IPB	Directeur de Thèse
Pr Dominique Gaiti	Professeur, Université de Troys	Rapporteur
Pr Pascal Lorenz	Professeur, Université de Haute Alsace	Rapporteur
Pr Damien Maguoni	Professeur, Université Bordeaux-1	Président de jury
Dr Djamel Meddour	Responsable R&D Orange, Lannion	Examinateur
Dr Olivier Fourmaux	Maitre des conférence, UPMC, Paris	Examinateur

Abstract

Recently we witnessed an increasing demand for scalable deployment of real-time multimedia streaming applications over Internet. In this context, Peer-to-Peer (P2P) networks are playing a significant role for supporting large-scale and robust distribution of multimedia content to end-users. However, due to peers' dynamicity, heterogeneity of terminals and access networks, the deployment of real-time video streaming applications over P2P networks arises lot of challenges. Indeed, an important issue in P2P overlays is the capacity to self-organize in the face of the dynamic behavior of peers in order to ensure content availability and continuity. In addition, the heterogeneity in networks, terminals, and P2P characteristics make the situation more challenging. In this context, layered video streaming in P2P networks has drawn great interest to overcome these challenges, since it can not only accommodate large numbers of users, but also handle heterogeneity of peers. However, there is still a lack of comprehensive studies on video data blocks (chunks) scheduling and bandwidth allocation for the smooth playout in layered streaming over P2P networks.

The aim of this thesis is to analyze these concerns and to propose an efficient real-time chunks scheduling and bandwidth allocation mechanisms for QoS provisioning of layered streaming applications over P2P networks. Our contributions in this thesis are threefold. First, we propose a scheduling mechanism for layered P2P streaming. The proposed mechanism relies on a novel scheduling algorithm that enables each peer to select appropriate stream layers, along with appropriate peers to provide them. The presented mechanism makes efficient use of network resources and provides high system throughput. Second, we propose a bandwidth allocation model for P2 layered streaming systems based on auction mechanisms to optimize the allocation of sender peers' uploads bandwidth. The upstream peers organize auctions to "sell" theirs items (links' bandwidth) according to bids submitted by the downstream peers taking into consideration the peers priorities and the requested layers importance. The ultimate goal is to satisfy the quality level requirement for each peer, while reducing the overall streaming cost. Finally, we present a smoothing mechanism for layered streaming in P2P networks. The mechanism aims to reduce the number of layer changes under varying network conditions, and ensure a smooth playout for the end-user.

Keywords: P2P, H.264/SVC, Real-time Video Streaming, Bandwidth Allocation, Scheduling, Quality-Of-Service

Résumé de Thèse

Le but de cette thèse est de proposer des mécanismes efficaces pour l'ordonnancement des chunks et l'allocation de la bande passante dans le contexte de la transmission vidéo sur les réseaux P2P, afin d'offrir une meilleure qualité de service pour l'utilisateur final. Dans un premier temps nous avons proposé un mécanisme d'ordonnancement des chunks pour la transmission de vidéo multicouche dans les réseaux P2P. Le mécanisme proposé est basé sur une nouvelle technique qui permet de sélectionner les chunks adéquats et les demander des pairs les plus appropriés. Ensuite nous avons proposé un mécanisme d'allocation de la bande passante, toujours dans le cadre de transmission de vidéo multicouche dans les réseaux P2P. Le pair émetteur organise une enchère pour «vendre » sa bande passante. L'allocation tient en considération la priorité des pairs et l'importance des couches demandées. Finalement nous avons proposé un mécanisme d'adaptation lisse « smooth » d'une vidéo multicouche transportée sur un réseau P2P.

Après une introduction, nous présentons dans le chapitre 2 les motivations du travail le but du travail et les problèmes recherche qui demeurent. Dans ce chapitre nous présentons les composants des systèmes P2P et tout particulièrement la distribution et l'adaptation de contenus. Dans ce cadre, nous proposons une classification des applications de streaming vidéo P2P ainsi que des mécanismes d'allocation de bande passante et d'ordonnancement pour le streaming pair-à-pair. Nous nous intéressons également aux techniques d'adaptation de la qualité en se focalisant plus particulièrement sur la norme SVC (Scalable Video Coding).

Le chapitre 3 propose des mécanismes de priorisation pour la planification de streaming P2P multi-couches. Nous proposons une heuristique pour résoudre un problème général d'affectation généralisé (Generalized Assignment Problem – GAP). La solution présentée est ensuite adaptée au cas du streaming non multicouches. Les résultats issus des simulations montrent que les solutions proposées donnent de meilleurs résultats que les solutions traditionnelles.

Le chapitre 4 décrit un mécanisme d'allocation dynamique de la bande passante pour les réseaux de streaming P2P multicouches qui se base sur l'allocation d'une bande passante aux pairs tout en assurant un minimum de qualité de service à l'ensemble des pairs. Les bonnes performances des mécanismes proposés, qui sont détaillées à travers l'étude du ratio concernant l'utilisation de la bande passante ainsi que du niveau de satisfaction des pairs, montrent que ces derniers permettent d'obtenir une utilisation optimale de la bande passante.

Le chapitre 5 porte sur le lissage du streaming multicouches dans les réseaux P2P en se basant sur les métriques liées à la variation de la fréquence et de l'amplitude. Les mécanismes proposés ont été implémentés dans un banc d'essai réel et l'évaluation des performances montrent l'efficacité des mécanismes pour le lissage du streaming.

Dans le chapitre 6 (conclusion and perspectives), nous résumons les contributions proposées dans cette thèse ainsi qu'une ouverture sur les travaux futures

Mots clés: P2P, H.264/SVC, transmission vidéo temps réel, Allocation de bande passante, Ordonnancement, Qualité de service

Table of Contents

List of Figures

List of Tables

Acknowledgments

Tout d'abord je tiens à remercier mon directeur de thèse Toufik.

Un merci à mes amis et mes collègues.

Je dédie cette thèse à mon père Abdelkader, ma mère Fatma, mon frère Mohammed, ma sœur Amina et ma sœur Khadidja

Chapter 1

1. Introduction

1.1 Motivation

Future networked media environments will differ significantly from today's applications in two important dimensions. They will be high-quality, multi-sensory, multi-viewpoint and multi-streamed, relying on HD and 3D video which will place unprecedented demands on networks for high capacity, low-latency and low-loss communications paths. Advanced media applications will also be more interactive and distributed, putting the users at the center of a massively multi-participant communications environment where they can interact in real-time with other user and provider resources, to provide and access a seamless mixture of live, archived and background material.

High-definition, highly interactive networked media applications pose major challenges to network operators. Multi-sourced content means higher quantities of data throughout the network, putting additional pressure at the network edge for unprecedented upload capacity in access networks.

On 1st June 2011, the US company Cisco Systems published its white paper "Visual Networking Index: Forecast and Methodology, 2010 -2015" [1]. Cisco announced that the Internet traffic worldwide will increase fourfold by 2015 (Figure 1) driven by the rapid increase in video traffic generated by both consumers and industries, including TV, Video-on-Demand (VOD), Internet video and P2P video. Cisco forecasted, based on data from CAIDA [2], by 2012 Internet video will account over 50 percent of consumer Internet traffic. The amount of all forms of video (TV, VOD, Internet video, and P2P video) will surpass 90 percent of global consumer traffic by 2015. And video will over-take P2P traffic, which was the top traffic source in 2010. Furthermore, global mobile data traffic will growth 26 times between 2010 and 2015. Mobile video has the highest increase rate among any other application category measured in the Cisco VNI prediction report. Almost 54 percent of the world's mobile content traffic will be video by 2015.

There are abundance of multimedia streaming systems such as commercial ones such as Microsoft windows media player, Real player and QuickTime of Apple, non-commercial ones such as VLC, and streaming application systems such as Youtube. However, still a small portion of overall Internet traffic such as video over chat systems (Tencent QQ [3], Skype and video calling) are experiencing high growth. Skype hit 663 million users as of 2010 and controlled more than 13 percent of international phone communications in 2009 [4].

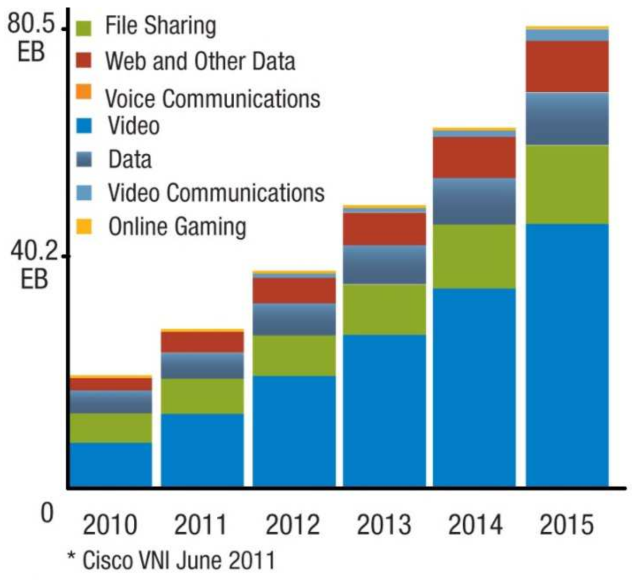


Figure 1: Internet traffic trend

Video streaming systems is getting more and more popular. The capacities and the access bandwidth of servers are bottlenecks of multimedia systems. As a consequence, more and more video streaming systems are using P2P streaming technology such as PPlive [5] and Zattoo [6]. Every user watching video using P2P streaming system uploads content at the same time when the video stream is being downloaded and watched. The user becomes a server without the cost of the traditional streaming systems. Hence, the more the number of online users, the faster is the content can be accessed. P2P systems also enable a client to automatically find other peers (other users) and get media video data from the appropriate peer (The peers with the close distance in terms of high bandwidth and low delay).

Research in the area of video streaming systems including for example Video-On-Demand (VOD), video sharing, and live streaming has been ongoing for many years. Due to the heterogeneity of networks and peers, the delivery of media streams is still facing many challenges. In particular, more and more multimedia streams have been delivered using application layers multicast and P2P networks. On the one hand, today's Internet only offers best-effort service, and it does not guaranty quality of service (QoS) for multimedia streaming applications. On the other hand, the capacity of devices in P2P overlay network varies from high performance workstations to low performance mobiles. In addition, with the large use of video streaming applications, resources like bandwidth, memory, storage are more and more rare.

In general, the effective delivery of time-sensitive and bandwidth demanding media video is still fronting many defies while video streaming is getting more and more popular.

1.2 Scope

P2P networks are distributed Internet systems in which peers cooperate with each other to achieve a desired service. The successful deployment of P2P technologies aims to overcome the limitation of client /server model by facilitating content sharing at large scale [7]. In such systems, there is no central entity to administer, organize and control the whole system. The P2P networks are participating, decentralized and self-organized networks. Indeed, in these networks, peers participate in resources distribution and consumption process. Resources can be physical resources such as network bandwidth, processing power or memory space, or logical resources such as services or different kind of information. In addition, in P2P networks there is no central entity to manage the entire network. But the network organization and content distribution process are handled by all peers in the network in collaborative manner with their neighbors without referring to a central entity. The main advantage of this characteristic is to avoid a single point of failures for the system. In P2P networks, participating peers self-organize themselves based on the information available in the neighborhood. They have to adapt to any upcoming event such as peers joining/leaving the system, bandwidth drop, etc.

In its infancy the P2P networks were exploited mainly by file sharing applications. However, in the past few years, audio and video streaming over P2P networks has emerged as a promising approach for distribution of multimedia content, instead of the traditional clients/server model [8]. While classic P2P file sharing applications are targeted for elastic data transfers, P2P streaming applications focus on the efficient delivery of the multimedia content under hard timing requirements.

The Internet Protocol (IP) is based on best effort packet switching and it does not present any guarantee in terms of Quality of Service (QoS) such as delay, jitter, packet loss, and bandwidth. Whereas, real-time video streaming applications require real-time performance guarantee in terms of restricted delay and jitter, bandwidth guarantee and low packet loss. Many parameters affect these performance metrics. Some of these parameters depends on the end systems such as video source load and others depend on the network conditions such as link capacity, intermediate routers congestion, etc. Besides, P2P networks are compound of heterogeneous networks and devices, which may have different characteristics to offer the same video quality to end users. Thus, a key goal of P2P video streaming system consists on the reliable delivery of high quality video over Internet while coping with unpredictable and dynamic issues of bandwidth, delay, jitter, packets loss [9].

In P2P live streaming systems, it is not uncommon to have hundreds of thousands of users trying to join a program in the first few minutes of a live broadcast, such as live football match, live concert, etc. This phenomenon, unique in live streaming systems, referred to as the flash crowd, poses

significant challenges in the system design to ensure the scalability of the system and timely content delivery (liveness).
The peer churn is another problem that the P2P systems suffer from. Indeed, the lifetime of each peer is subject to self-decisions of the users and is outside the control of the P2P designer. To successfully function, an overlay must compensate for the variability in peer membership and dynamic behavior of participating peers [10].

In addition to P2P development, new video coding standards have emerged in the last few years, which are allowing the deployment of real-time multimedia delivery over Internet with high QoS. Among these standards we can distinguish: MPEG-2, H.264 AVC and Scalable video coding (SVC) [11].
MPEG-2 standard is appropriate for most broadcasting systems such as DVB-T and DVB-S and data storage in DVD. MPEG-4 AVC (Advanced Video Coding), called also H.264, is a standard that enables high quality video encoding with error resilience carry. It is suitable for broadcasting and mobile communication. Scalable Video Coding (SVC) is as an extension to H.264 standard. It is considered as the most promising encoding standard. It is capable to produce video streams with variable bitrates, different temporal, spatial, and quality scalability levels. SVC is appropriate for delivering video content over heterogeneous networks such as Internet, mobile P2P and P2P TV. It seems to be the most attractive solution to the problems identified in modern video transmission, such as varying needs or preferences of end users as well as varying terminal capabilities or network conditions. Indeed, SVC has achieved significant improvements in coding efficiency with an increased degree of supported scalability relative to others video coding standards.

SVC over P2P networks received ample attention from the research community [12] [13][14][15] in order to take advantage of both, the scalability of P2P networks and dynamicity of SVC. Nevertheless there is still lack of comprehensive study in resource allocation and data chunks scheduling in such systems. This motivates us to propose bandwidth allocation and new scheduling mechanisms for smoothed delivery of layered streaming over P2P systems. In the following sub-section, we present the problem statement and our key contributions in this dissertation.

1.3 Research problem

Peer-to-peer (P2P) networks are getting increasingly popular for streaming video over the Internet. Due to peer dynamics, single-layer stream can neither match the overlay capacity changing, nor meet heterogeneous peer requirements. Layered streaming, such as Scalable Video Coding (SVC), provides a convenient way to perform video quality adaptation to adjust to the changing network conditions and receiver preferences. A layered streaming consists of a base layer and multiple enhancement layers. Receivers can adjust the video quality level to their capability by subscribing to different number of layers using pulling distribution approach. However in practice, the task of ensuring acceptable QoS/QoE (Quality of Experience) for real-time video streaming applications in the network is more and more challenging task, due to the heterogeneous nature of the internet and terminals and the characteristics of P2P networks, mostly the unpredictable behavior of peers in terms of joining/leaving the network and their contribution in the network. In this context, we need to address the problems related to:

– *Appropriate chunks selection*: In P2P video streaming systems, the content retrieval mechanism allows a user to receive streaming data blocks (chunks) from other peers using the constructed overlay. This mechanism plays a leading role in the video streaming process and its efficiency influences the global performance. The scheduling task is complicated in the context of video streaming since chunks received after their playback deadline are not played and considered as useless chunks. Moreover, in the context of layered video, the task becomes more and more complicated, since an additional constraint should be taken into consideration, namely the layers' dependency: chunks of higher layers received without their corresponding lower layers' chunks are not played and considered as useless chunks too. Hence, the scheduling mechanism should be aware of the chunks priority in terms of playback deadline and layer dependency. It should be also dynamic to adapt to the network conditions changes, and efficient to fully take advantage of the available bandwidth in the network.

– *Optimized mechanism for bandwidth allocation*: Efficient chunks scheduling and appropriate bandwidth allocation are the two significant challenges in real-time P2P streaming systems. The two components cannot be dissociated each from the other. A good exploitation of the sender peers' bandwidth cannot be reached without optimized scheduling scheme. On the other hand, an efficient scheduling algorithm cannot achieve a good throughput without appropriate sender peers' bandwidth allocation mechanism.

  While considering the dynamicity of peers in terms of joining/leaving the network, the heterogeneity of peers and their need for bandwidth, an efficient and dynamic mechanism for bandwidth allocation is required to ensure the timely availability of the streaming content and acceptable quality level for peers while fully taking advantage of the overall bandwidth in the network.

– *Smooth delivery of content:* The quality adaptation mechanism is required to handle the heterogeneities of client in P2P networks. However, the sudden variation in bandwidth may arise to sharp quality level variation which is annoying to the end user. A smooth mechanism is needed in order to ensure a stable, long-term smooth quality level for end users.

1.4 Contributions

In this dissertation, we focus on providing the quality-aware services for video streaming in P2P networks in order to enhance the peers quality level satisfaction, maximize the system throughput while reducing the useless chunks and maximizing the bandwidth utilization. Our contributions concern mainly the quality level adaptation, chunks scheduling and resources allocation in P2P layered streaming systems, for both live and on-demand systems.

We leverage the characteristics of efficient scheduling mechanisms based on optimization theory with smooth delivery algorithm, and economical model inspired bandwidth allocation scheme for layered streaming over P2P networks. The proposed streaming mechanisms consist of:

- Chunk scheduling mechanism: Our first contribution is a receiver side scheduler for video streaming over P2P systems. This scheduler assigns different chunks to sender peers on the basis of the chunks priority and playback deadlines. The ultimate goal of the proposed scheduling mechanisms is to fully take advantage of the available bandwidth in the neighborhood and reduce the useless chunks received after their playback deadline or without their corresponding lower layers' chunks. First, we theoretically study the scheduling problem in pull-based P2P video streaming and we model it as an assignment problem. Then, we propose AsSched, new scheduling algorithm for layered streaming, in order to optimize the throughput and the delivery ratio of the system. After that, we derive an optimal scheduling algorithm (NAsSched) for non-layered streaming.

- Bandwidth allocation mechanism: In order to better take advantage of the overall network bandwidth and reduce the useless data, due especially to the higher layers received without their corresponding lower layers, we propose an economical model based on auction mechanisms to optimize the allocation of sender peers' upload bandwidth in the context of P2P layered streaming. The upstream peers organize auctions to "sell" theirs items (upload bandwidth) according to bids submitted by the downstream peers taking into consideration the peers priorities and the requested layers importance. We first proof the Nash equilibrium of the system. Then, we perform extensive simulations to show the effectiveness of the proposed system in terms of bandwidth utilization and peers' quality level satisfaction.

- Smooth delivery for layered streaming: The layered encoding offers a fine grained control on video quality. However there are several complexities due to bandwidth variation and peer's unpredictable behavior. It is observed that the end user perceived quality oscillates very frequently within a short period of time. In this case, the accurate decision regarding the layers selection is very critical. For that purpose, we proposed a mechanism for smoothing of layered stream. The proposed mechanisms aim to reduce the frequency and the amplitude changes of layers. We showed that the number of layers change is drastically reduced using our mechanisms. The frequency reduction algorithms present better performance with large smoothing window, but to the detriment of the liveness of the stream and the effective utilization of the available bandwidth. As future work we plan to study techniques to achieve a trade-off between the smoothing quality and the liveness of the stream by acting on the smoothing window size.

1.5 Dissertation Structure

This dissertation is organized as follows:

Chapter 2 presents a state of the art of P2P streaming systems. We start first by presenting the architecture of P2P systems and a classification based on their decentralization degree. Then, we present the three components of P2P streaming systems namely the overlay construction, content delivery and content adaptation. The focus is given on the two last components which are more related to our contributions. In the content delivery component, we present first a classification for P2P video streaming applications based on the liveness of the stream. Then, we detail a state of the art of the scheduling and bandwidth allocation mechanisms in P2P streaming and we propose a classification for each technique. Moreover, we present the quality adaptation component of P2P streaming systems. Under this component, we present an overview on adaptation techniques and we focus on SVC adaptation in P2P systems.

Chapter 3 describes our first contribution for efficient scheduling for P2P layered streaming. We propose a chunk prioritization strategy. We then model chunk scheduling as an assignment problem and we propose a heuristic to solve it. The proposed solution takes into account both the available bandwidth and chunk availability for each peer. Moreover, we derive an optimal scheduling algorithm for non-layered streaming systems.

Chapter 4 describes in detail the proposed auction mechanism for bandwidth allocation in P2P layered streaming and the related theoretical study. In this work we dynamically allocate the bandwidth taking into consideration the peers request in terms of layers as well as their priority.

Chapter 5 deals with the problem of smoothing in P2P layered streaming systems. In this chapter, we consider two different metrics that are essential for understanding the problem of smoothing: the frequency and the amplitude quality level variation. We then propose different mechanisms to handle the smoothing problem considering the importance of each metrics.

Chapter 6 concludes this dissertation and addresses some of our future work.

Chapter 2

2 Related Work

2.1 P2P Networks

During the last decade, Peer-To-Peer (P2P) architecture has drawn remarkable attention from both academia and industry. The P2P experience has revolutionized the manner how people consume and interact with multimedia content. This exponential development is amplified by the availability of high speed Internet connections and the large adoption of portable media capturing devices, such as cell phones, cameras and other network-attached recording equipment.

The P2P networks are virtual networks built on top of the physical networks. They are made of heterogeneous interconnected peers that can have different capabilities and characteristics in terms of bandwidth, memory storage, processing power, etc. Milojicic et al. [7] define P2P networks as "a class of systems and applications that employ distributed resources to perform a function in a decentralized manner. The resources encompass computing power, data (storage and content), network bandwidth, and presence (computers, humans and other resources)". This definition brings out the concepts of decentralization, resource sharing and self-organization; characteristics that make this architecture more popular to provide efficient solutions. The P2P characteristics include:

- Decentralization: contrary to the client/server model, pure P2P networks do not have any central entity that has global knowledge/control of the system to organize and monitor the network.
- Resource sharing: P2P networks became popular since the emergence of file sharing systems, such as Napster [16] and Bittorrent [17]. Today, several other applications such as video streaming, telephony over IP and video conferencing are exploiting the P2P network architecture.
- Self-organization: In purely P2P networks, there is no global index of peers or resources, peers organize themselves into a network using a discovery process.
- Symmetric communication: The participating peers are extremely symmetric in P2P networks. There is no difference between client and server. These peers are identical in terms of roles and they collaborate with each other for the same objectives. All the communications between peers are also symmetric and peers join/leave P2P networks voluntarily.

- Scalability: In P2P networks identical resources may be available at different peers. The aggregation of peer's resources leads to high scalability with handy load-balancing between a large number of peers.
- Robustness: Contrary to the client/server model, P2P networks afford a robust solution in which peers do not suffer from a single point of failure problem. Every peer shares its resources with other peers and the failure of some or many peers might have no or small impact on the system functionality.

P2P systems guarantee the good utilization of resources with high consistency but on the other hand they present certain problems. Indeed, P2P networks incur from dynamicity and heterogeneity of its peers. In this regards, QoS guarantee for multimedia applications becomes an important issue. In addition, P2P networks are exposed to security, copyright violation and privacy issues that need to deal with for more efficient solutions. These latter challenges are out of scope of this thesis.
The following subsection summarizes some important aspects of P2P networks, such as their architecture and a topology-based classification.

2.1.1 Architecture

In Figure 2 we present a layered architecture of P2P networks. It consists of the following layers: the network communications layer, the routing layer, the control layer, the overlay components layer and the service layer. The layers are described in the following:

- The Network communications layer depicts the network characteristics of the participating peers in the overlay. Peers are machines connected over the Internet using the P2P overlay organization.
- In P2P networks, peers are very dynamic and they benefit from a significant autonomy. Mechanisms to search for resources and retrieve them from best peers are essential mechanisms that should be provided by a P2P system. Toward this, the routing layer ensures the peers management, including peers discovery and routing algorithms.
- The control layer deals with the P2P feature like reliability, fault resiliency, security and session management in order to maintain the robustness of P2P systems.
- The overlay component layer includes three modules: the overlay organization module that ensure the construction and the repair of the overlay network. The content delivery module ensures the distribution of the content including the scheduling and the resource management features. And finally the content adaptation module that ensures the adaptation of the content to the network condition and the end user context. These modules are presented in detail in section 2.2.

– The service layer is the top layer of this architecture. It includes the different services provided by the P2P network such as the file sharing, internet IPTV and video streaming (live and on demand). In the following section we develop the routing layer.

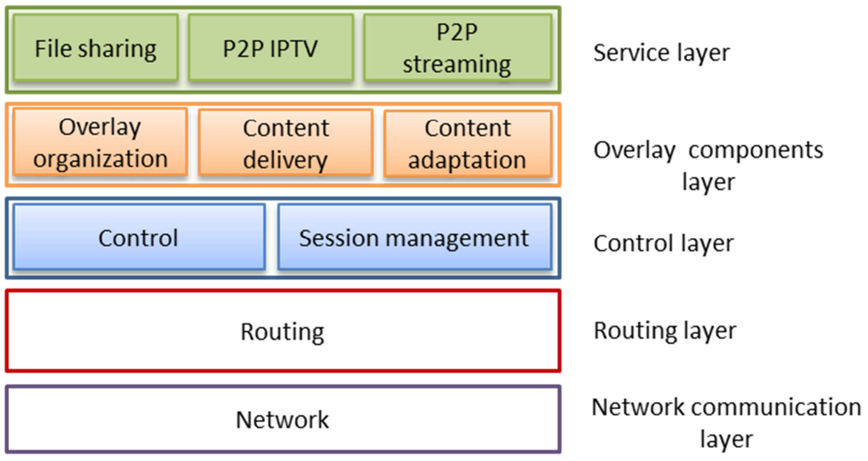


Figure 2: An Abstract P2P Overlay Network Architecture

2.1.2 Classification

Regarding the P2P overlay topology and the content discovery approach, the P2P systems can be classified into three classes: centralized overlays, decentralized structured overlays and decentralized unstructured overlays [18].

2.1.2.1 Centralized P2P systems

In the *centralized P2P systems*, a global index is maintained in a centralized server for all contents in a form of <content-key, peer address> items. Each peer that joins the overlay publishes the information about the content that it is willing to share in this server. Regarding the content discovery mechanism, the peer has just to retrieve from the centralized server the addresses of the peers that provide the requested content. This type of architecture is very simple and easy to set up. The scalability property is reached by the centralized discovery facility. Therefore, it does not require a lot of bandwidth for content discovery. Nevertheless, the central server represents a point of failure for the system. The pioneer P2P file sharing system, Napster [16], is an example of this architecture.

2.1.2.2 Decentralized structured P2P systems

In the *decentralized structured P2P systems*, the content discovery is rather distributed. In addition the overlay topology and content placement are strongly controlled in order to make the queries more efficient. These systems use Distributed Hash Table (DHT) as a support. For each content is affected a key, which is mapped to peers in a well-defined graph. These structured systems allow an efficient content discovery. Nevertheless, they do not deal with complex requests and need to store a reference to the content at the peer which is responsible for the associated key. CAN [19], Chord [20] and Tapestry [21] [22] are examples of decentralized structured P2P networks.

2.1.2.3 Decentralized unstructured P2P systems

The *decentralized unstructured P2P systems* rely on unstructured overlay networks in which the content discovery and download capabilities are distributed between peers. The overlay is organized as a random graph with some slack rules, where joining peers do not have any prior information about the topology of the overlay. They discover the contents using a flooding-based mechanism. To request content, a peer sends a flood query with a limited reach through the overlay. The query is executed hop-by-hop until its satisfaction, its failure or time-out. Although, the flooding-based mechanism used in this category of networks is resilient to peers dynamicity and efficient to locate extremely replicated resources. This architecture suffers from an additional load of requests replication which makes the numbers of requests in the network increase. In addition, this mechanism is not efficient for locating rare resources. Such systems are the most common in today internet such as Freenet [23] and Gnutella [24].

The P2P networks can be seen as the extreme of distribution which confer them an infinitely scaling. However the networks present also many weaknesses such as high peer churn, heterogeneous resources capabilities and that rapidly result on poor quality.

2.2 Video Streaming over P2P Networks

Video streaming denotes a delivery method by which a video stream is delivered from a source to clients in a continuous manner and consumed in real time by client application [25][26]. In P2P streaming systems, the source splits the media into pieces called chunks, defined as the small piece of stream that can be played independently. Each chunk is composed of one or several time stamped packets. The packets are reconstructed at the client side when they are being received. Streaming applications differs from file downloading applications in the sense, that in the former the media are consumed on arrival rather than downloading all the media first and consume it after. This is, indeed, the main important specificity of the media-streaming systems.

In P2P streaming systems, three essential components need to be considered: *overlay construction, content delivery* and *content adaptation.* The first two components are basic building blocks in most P2P systems, and as such have received ample attention from the research community. However, in order to tackle heterogeneity in terms of terminal capabilities, network conditions and user preferences, *content adaptation* is rapidly becoming a core component for such systems. Traditionally, the overlay construction component deals with the selection of appropriate neighbors for the content retrieval, and the content delivery component is responsible for the requesting and transport of content chunks from the chosen overlay neighbors.

2.2.1 Overlay construction

Like in the traditional P2P systems, peers in P2P streaming systems need to organize themselves in an overlay network in order to start the media streaming process. Peers must be able to explore the required content and the peers holding that content.

In practice, the P2P streaming systems rely on the following mechanisms to discover and locate the content: (1) tracker based method, (3) gossiping method and (3) Distributed Hash Tables (DHT).

Trackers are super-peers used to save information about "which peer holds which content". Each peer in the overlay starts consuming the content informs the tracker that it holds that content. Equally, when the peer no longer holds the content in its cache notifies its tracker. The consumer peer requests the tracker for a list of peers having the desired content. This later answers with all peers or an optimized subset of peers holding that content, and it is up to the requester peer to establish neighboring relation with these peers in order to start requesting the content.

Peers in the P2P network exchange periodically with their neighbors information about chunks that they hold, referred as *chunks Bitmap* or *buffer maps*. Based on this information, it selects which neighbor to provide the required chunks. The tracker is no longer solicited in this stage, and peers communicate directly between themselves, thus make the system more robust.

The second content retrieval mechanism relays on gossiping to find other peers consuming the same content [27]. A gossip protocol is inspired from gossip seen in the social network. Indeed, each peer relays information about chunks seen or heard about it in the neighborhood. This gossiping can be pull-based or push-based, depending upon whether the receiver peer request this information or peer actively advertise information to its neighbors. The term epidemic protocol is sometimes used instead of gossip protocol, because gossip spreads information in similar manner to the spread of a virus in a biological community. Coolstream [28] is a real-world implementation of gossip-based IPTV stream delivery system.

Another alternative for content retrieval in P2P streaming systems is to use a Distributed Hash Table (DHT) in order to store and retrieve content in the overlay. It provides a lookup service such as a hash table: (key, value) pairs and any participating peer can efficiently retrieve the content associated with a key. Maintaining the mapping of keys to contents is distributed among the overlay peers in such a way that the change in the set of participant peers doesn't seriously disrupt the system. This approach minimizes the dependency of the P2P network of a central entity such as tracker.

2.2.2 Content delivery

In addition to overlay construction component that organizes participating peers into an overlay construction on the basis on certain logic, the content delivery mechanism is another key component of the P2P system. It determines how the media content should be delivered to each participating peer using the overlay. Many researches [28][29][30][31] have talked the overlay constructions problem, but how to schedule media content to many neighbors while ensuring a good QoS (Quality of Service) level is still a big challenge for P2P streaming systems.
In this section we will introduce first the different media content delivery modes (based on the liveness of the content). Then, we will investigate the key scheduling strategies in P2P streaming systems; the empirical scheduling mechanisms and the non-empirical ones. Finally we address a state of the art on the bandwidth allocation problem in P2P streaming systems.

2.2.2.1 Media content delivery modes (Live, VOD, Time-shifted)

Based on its liveness, video streaming systems can be classified into three main categories: live streaming, on-demand streaming and time-shifted streaming systems.

Live streaming system distributes the fresh generated content to users as it is produced. The live streams are immediately played by the consumer peers. Broadcasting a live sport event directly to users is an example of live streaming. On the opposite, on-demand streaming allows to stream a fully or partially stored video to users on their demand. In this kind of service a user can select to play a video from any point, this is why only the part that the user is interested in is transmitted [32]. On-demand streaming permits the user the entire control of video by allowing the VCR operations like backward , forward, pause, resume, etc. Time shifted streaming permits the peer to access the past part of a live stream. Even the live stream has already started, the peers can start watching a stream from the beginning. The peers have total control on the past part of the stream and can join the live portion if chosen.

2.2.2.1.1 Live Streaming Systems

In live streaming systems [34], the delay between the creation of the media and the delivery to the end-users should be very short, ideally nil. In practice, the transmission over the Internet increases the delay. Issues such as non-deterministic join, peers churn, are common in live streaming. The algorithms designed for supporting live streaming should consider two important characteristics of the live content:

– A user joining an already started live streaming session gets the stream from the time of request. This allows a new peer to join an ongoing stream. Thus, supporting live streaming is relatively simpler since a request can be satisfied by any peer that is previously playing the stream.

- End-to-end delay from the time of media generation to its delivery is essential in live streaming. In these systems, the shorter the end-to-end delay is, more lively the stream is perceived by the end users (referred also as liveness). The algorithms designed for live streaming over P2P networks aim to reduce the end-to-end delay in order to ensure the liveness of the stream.

CoolStreaming/DONet [28] proposed by Zhang et al. is one of the first P2P live streaming systems. It is a framework for live media streaming which rely on data-driven overlay networks where peers exchange periodically with each other information about content availability. Thus, the availability of data decides its flow instead of a fixed structure. Contrary to Cool Streaming/DONet's, PRIME [35] builds a fixed mesh network that is randomly connected.

The most popular P2P IPTV service is PPLive [36]. It is a mesh-based P2P live streaming system based on the buffer-maps exchange for content location and gossiping for peers management and channels discovery.

2.2.2.1.2 On-Demand Streaming Systems

Contrary to live streaming systems, the on-demand streaming systems can distribute the content from any position in the video regardless the time of its request. In addition, the end-to-end delay in on-demand streaming systems is not very important comparing to live streaming systems. This is because the stream is pre-stored and not distributed at the production.

Even if the delay between the stream creation and distribution is not critical for on-demand streaming systems, the delay between the request and the reception by the end-user is very important. Along with the above pointed out differences, in an on-demand session, a peer can move to another position in the video or stop watching the video when the QoS degrades. Consequently, the P2P network becomes unstable if the number of such peers increases. This problem does not arise in the live streaming systems because the user it can't perform VCR (Video Cassette Recording) operation and does not have the choice of watching the video again in the future.

P2CAST [37] is an architecture for VOD services over P2P networks. The basic idea behind this architecture is that each peer acts as server and offers some pieces of data to other peers. This architecture used the P2P approach in order to overcome the problem of overload on a single streaming server.

Authors in [38] focus on enhancing the VOD user experience, such as small start-up delay and a suitable playback rate. They claim that the VOD users experience can be significantly improved using a good prefetching strategy.

2.2.2.1.3 Time Shifted Streaming

Regarding time shifted streaming technique, it permits users to voluntary pause a live video stream, rewind, and fast forward until current live position of the stream [39] [40]. Peers stock the received stream in their buffer. If a peer rewinds or fast-forwards to content missing in its buffer, the requested content is reclaimed from the neighbor peers. The peers buffering the required content provide it asynchronously regarding the live stream issued from the source of the stream. In the following paragraphs we describe briefly some time shifted systems.

Gallo et al. proposed in [41] an architecture that utilizes multicast, distributed caching and P2P distribution to provide time-shifted streaming services. The main idea behind this work is to leverage the live transmission to cache the stream in a distributed manner, avoiding retransmissions through the network when the same stream is demanded as time-shift service. The proposed solution seems suffer from a critical problem. The use of the hash functions, to retrieve the available chunks at different peers, appears irrelevant in this context since chunks are dynamically produced.

In [42] Liu et al. proposed a distributed time shifted streaming service. Fundamental challenges of time-shifted systems are tackled in this work. Especially, it allows the storage of all past chunks with a guaranteed quality of service. It ensures load-balancing between all peers in terms of chunks storage and delivering.

We note that in our contributions we are considering real-time streaming systems in general, both live and VOD streaming systems are supported by the scheduling, bandwidth allocation and smoothing mechanisms that we propose.

In the next section we continue with the content delivery component and we present a state of the art on scheduling in P2P streaming systems.

2.2.2.2 Scheduling in P2P video streaming

P2P streaming systems can be also categorized into two classes, namely tree-based and mesh-based. The tree-based systems, such as P2Cast [37] and ESM [43], have well-organized overlay and distribute video content by dynamically pushing data from a peer to its direct children. The vulnerability to peer churn is the major drawback of these systems. In the contrast, in the mesh-based P2P streaming systems, peers are not limited to a static topology. But, a peer dynamically connects to a subset of peers in the system. The stream is divided into small pieces called chunks, which follow independent paths to other peers. Many mesh-based P2P streaming systems have been proposed in the literature, such as CoolStreaming [44] and PRIME [45], etc.

In this section we focus on scheduling in mesh-based P2P streaming systems. We define first the scheduling problem in P2P mesh-based streaming systems. After that we go through the key scheduling mechanisms in the literature. Then, we give an overview on the main transmission strategies namely the push, pull and push-pull mechanisms.

2.2.2.2.1 Scheduling problem in P2P streaming systems

Blocks of a media stream, or chunks, do not contribute uniformly to the video quality at the receiver peer. A chunk is useful to the receiver peer only if: (a) it is received before its playback deadline, and (b) all the previous chunks needed for its correct decoding have been already received. The disparity in importance of chunks leads to the derivation of efficient chunk scheduling algorithms that decide which chunks should be requested at a given time instant and from which neighbor peer, in order to maximize the streaming quality [46].
Designing fully distributed video scheduling algorithms is a difficult task. Ideally, scheduling algorithms, executed independently on each peer, consistently select the set of video packets to be requested aligned with the upload and the download bandwidth of sender and receiver peers respectively. It is evident that the chunk scheduling mechanism should take into consideration the network bandwidth, processing power, memory size of peers, etc. Also it could include statistical information about historical data delivery in order to predict the willing and the reliability of peers to provide data. A scheduling scheme involves two main components: chunk selection mechanism and transmission scheduling mechanism. In the following sections we present these two components.

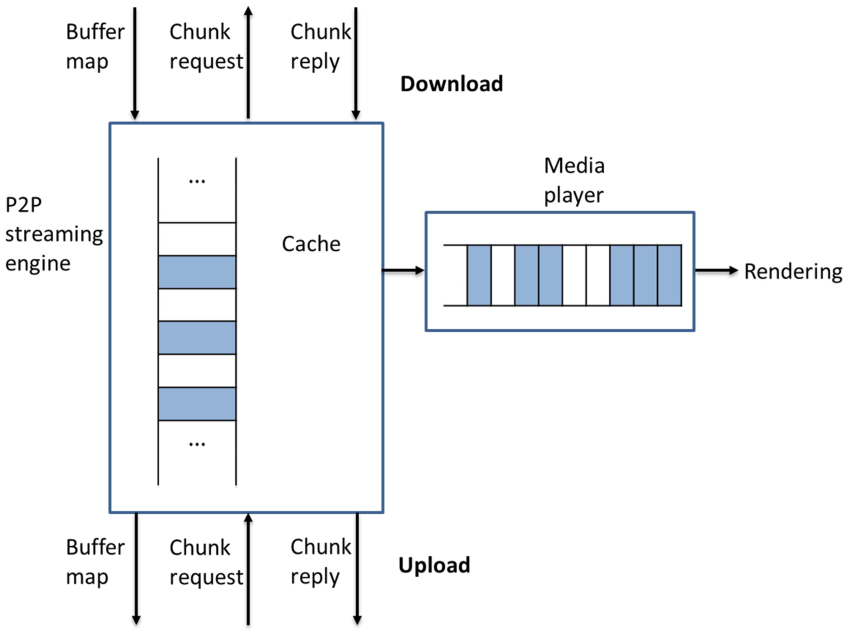


Figure 3: Scheduling in P2P streaming systems

2.2.2.2.2 Chunk selection strategy

Obviously, a chunk selection mechanism is a function that decides about the chunk to request and from which peer based on the information locally available at the peer.

The scheduling mechanisms proposed in the literature can be classified into two main categories: empirical-based scheduling and optimization-based scheduling mechanisms. The first category, as its name indicates, is based on empirical studies where justified or non-justified scheduling strategies are proposed such as the sequential, random, and the rarest first strategy. The second generation of scheduling algorithms is based on optimization theory. It relies on some mathematical modeling and tools to propose an optimized scheduling scheme. In this section, we present the key scheduling approaches proposed in the literature for each category and we discuss the strengths and the weaknesses of each approach.

2.2.2.2.2.1 Empirical-based scheduling approaches

A. Sequential

An intuitive chunk selection scheme is to select the chunk closest to the current playback position [47]. This strategy engenders, however, a big slant in the number of chunks replicas in the system. Especially, chunks near to the beginning of the video are much duplicated while chunks close to the end of the video are requested by only a few peers in a P2P live steaming system. This situation leads to a bottleneck in providing the tail chunks of the video stream. Even for VOD streaming systems it will be difficult, in this strategy, for two peers consuming different streams to request for each other.

B. Round-Robin Algorithm

While the sequential approach in [47] takes into consideration the upload capacity of the supplier peers to select the peer that will provide the chunk, the Round-Robin approach assigns each consecutive chunk to the next sender in a pre-formed sender list. Once the end of the sender list is reached, it starts again from the beginning of the list. It continues in this way until all chunks have been affected to the candidate sender peers. Every sender peer will have the same number of chunks to provide and no peer will be promoted based on its reliability, its estimated bandwidth, or any other parameter.

The Round-Robin approach, which assigns equal part of responsibility to all potential sender peers, is still widely adopted for its simplicity but is not appropriate for P2P streaming applications because of heterogeneity of peers especially in terms of upload bandwidth. It is suitable, however, as a basis for comparison with more advanced scheduling algorithms such as Rarest First [48]. Clearly, the main advantage of this approach is that it is very simple to implement and fast enough to be run in less powerful devices such as mobile phones. Its main drawback is that it does not consider the network conditions and peers capacities and that typically results in inefficient schedules.

C. Random scheduling

In this scheduling approach, every chunk is assigned to a random peer from the neighbor peer list. A perfect example of this approach is the Chainsaw scheduling algorithm proposed in [49]. Similarly to the Round Robin approach, no preference is given to any sender peer based on its upload capacity or its reliability, but they are chosen randomly. In most implementation, the random seed relies on the system time. This results on a pseudo-random schedule.

The main advantage of this approach is that there is a probability to lead to an efficient scheduling and the fluctuation in the network conditions usually does not have effect on the algorithm performance. The main drawback is that the chance that the resulting scheduling will be close to the optimal schedule is relatively low.

D. Rarest First

Another option for piece selection is the rarest first approach. The CoolStreaming algorithm implements the rarest first algorithm adopted by the CoolStreaming/DONet peer-to-peer system [48]. This heuristic algorithm assigns chunks of fewer suppliers with higher priority. It calculates the number of potential suppliers for each chunk (i.e., the neighbors containing the chunk in their buffers). It starts scheduling those having only one potential supplier, then those with two suppliers, and so on. If there is a chunk with different potential suppliers, the highest bandwidth supplier peer is chosen [48]. The idea behind this approach is that chunks with less potential suppliers have less chance to be provided before their playback deadline, so they need to be requested in priority.

The main advantage of this approach is that it tends to request more chunks from the fastest peers (that acquire chunks earlier) which is likely to be the most powerful peers. This helps speeding up the propagation of chunks in the network, hence enhancing the streaming quality. This approach improves the system scalability, as explained in [50].

The rarest first approach leads to an opposite effect to the one caused by the sequential approach. Indeed, the rarest first approach tries to increase the number of duplicated chunks by requesting those which are closer to the end of the video stream. Chunks near the playback position are ignored, which affects the playback continuity. In many situations this scheduler tends to stream from a single peer even if multiples neighbors having the content are available. This can induce the overload of the faster peers. In addition, this approach relies on the estimated bandwidth of a sender peer to determine if it has enough time to provide a chunk before its playback deadline. The problem when absolutely relying on the estimated bandwidth criteria is that sudden changes in bandwidth with the faster peer may causes missing of many chunks to their playback deadline.

E. Rarity-urgency Hybrid

Several other works proposed the urgency-rarity hybrid approach which not only relies on the piece rarity only but also selects chunks near to the playback position. Authors in [51] proposed a distorted random strategy, which gives preference to earlier chunks, while [52] [53] [54] [55] select chunks according to its priority. In PPLive [53], a mixed approach has been proposed. It gives the first priority to sequential, then the rarest-first. In order to achieve better streaming performance for the whole system, authors in [54] present a Mostwanted-Block-Download-First (MBDF) approach to replace the Rarest-Block-Download-First (RBDF) in Bittorent [ref] systems. In [55] Vlavianos et al. propose to subdivide the video buffer into two parts: the first part for high priority chunks and the second part for lower priority chunks. As the playback deadline of a low priority chunk approach, it becomes high priority. The rarest first approach is applied in the two parts of the buffer.

D. Anchor-based

In Video-On-Demand (VOD) streaming systems, users may jump forward or backward to a random position in the video. In order to support such VCR functionalities, a number of video anchor points are selected in the video [56]. Anchors are chunks each of a fixed time length distributed among the video with fixed interval. When a user tries to jump to particular position in the video, if the chunk for that location is missing, the closest anchor (ahead or behind) is used as an alternative. The anchor based approach give anchor chunks more priority. This make the VCR operation suitable, in the other hand it can affect the playback continuity of the stream because of giving more effort to anchor chunks rather than to urgent chunks for example. In practice, users do not jump from a position in the video to another very often. And by enhancing the scheduling algorithm, the initial buffering time after a VCR operation can be reduced to satisfactory level without implementing anchoring mechanism. This is why the anchoring approach is still under revision for future utilization.

2.2.2.2.2.2 Optimization-based scheduling approaches

Computing the optimal chunks schedules to maximize the video quality is computationally complex [57]. In last section we have presented the pioneer empirical-based scheduling strategies in P2P streaming systems; that resort to simple heuristics for chunk scheduling. Several other strategies have been derived from these works such as [58], which describes a weighted round-robin algorithm based on senders' upload bandwidth capacity, in addition to the work of Kowalski and Hefeeda [59] which proposes to assign each chunk to the sender that will provide it the first. However, all these heuristics still not sufficient to provide performance guarantees on perceived video quality and to perform well in streaming systems [38].

One way to deal with the complexity of the chunk scheduling problem is to state a utility function for the perceived video quality and try to find a scheduling algorithm that optimize this function [60][61].

Authors in [60] design a utility function for each chunk. The proposed function take into consideration the rarity of the chunk, which is defined as the number of potential senders of this chunk, and its urgency, which is the time difference between the current time and the playback deadline of that chunk. They then modeled the chunks scheduling problem into a complicated min-cost flow problem and propose a heuristic to resolve it. We note that even if min-cost flow problem can be optimally resolved, it is computationally expensive in terms of CPU resources consuming. In addition this scheduling algorithm is not suitable for the layered streaming.

Authors in [61] formulate an optimization problem to maximize the perceived video quality. They propose a chunk utility function driven by the chunk importance in the video decoding process at the receiving peer as well as the popularity of the chunk within the peer neighborhood. Authors propose to solve this optimization problem using an iterative descent algorithm named Iterative Sensitivity Adjustment (ISA). Nevertheless, this algorithm is computationally expensive and cannot be adopted in real-time systems. They propose instead to simplify the original formulation by an ad-hoc utility function for each chunk, which expresses its utility as the multiplication of its rate-distortion (R-D) efficiency, rarity, and urgency. They then use the greedy algorithm to schedule the chunks, i.e., they schedule the chunks with higher utility value first then the lower utility ones. This greedy algorithm does not provide optimal schedules nor any guaranteed performance.

Many other works on the chunk scheduling problem have been proposed but they do not directly resolve it. Authors in [62] propose a P2P scheduling strategy based on measuring the time required to download the entire video from a number of peers to select sender peers from a large group of candidate senders.

2.2.2.2.3 Transmission Scheduling Strategy

The scheduling algorithm has the task to select a number of video pieces to download or to upload. The question that arises is where the decision to request/distribute the chunks is taken, at the receiver peer level? At the sender peer level? And how the content is diffused: pushed by the sender peer or pulled by the receiver peer? All these tasks are performed by the transmission scheduling algorithm. There are two objectives in designing the transmission algorithm: (1) make best use of downloading bandwidth; (2) reduce the overheads. In this sub-section we give a brief outline of the main transmission scheduling mechanisms.

2.2.2.2.3.1 Push-based

In push-based strategies (Figure 4), scheduling decisions are taken at the sender peers level. Multimedia content in a tree overlay is often pushed from the source to other peers, such as ESM, which distribute video content by dynamically pushing data from a peer to its children. In addition, Guo et al [37] propose a covering scheme for such service. However, the tree structure brings

variation and it is difficult to conserve. In the meantime, the push-based method in adopted in mesh-based system also. To be efficient, each received block is uploaded to the maximum number of neighbors peers, trying to serve each piece to at least one neighbor peer [37]. The sender peers maintain or reduce the number of neighbors' peers to which it is sending the pieces concurrently depending on the block transfer time. In addition, the sending peer favors the neighbors' peers that reduce the waste of its upload bandwidth to the other peers. Push-based strategies are more appropriate for upload constrained systems, like those containing an immense part of peers connected through ADSL, since the distribution of blocks is then controlled by the sender peer.
Nevertheless, pushing blocks would invoke substantial overhead and it cannot assurance high playback continuity, as a peer may receive the same block recurrently from different neighbor peers.

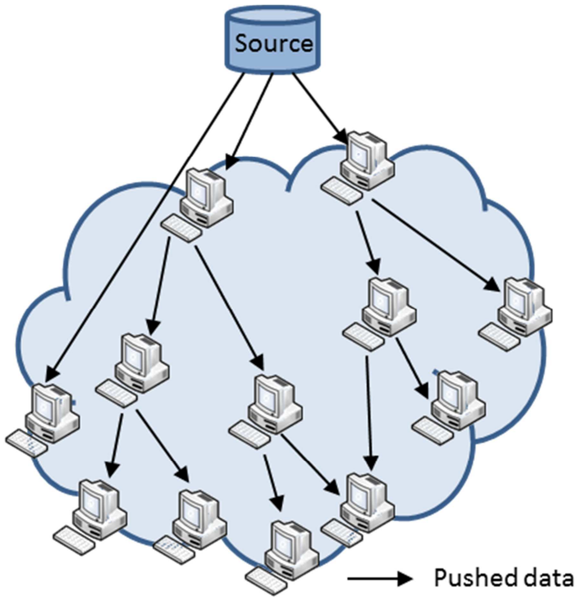


Figure 4: push-based transmission strategy

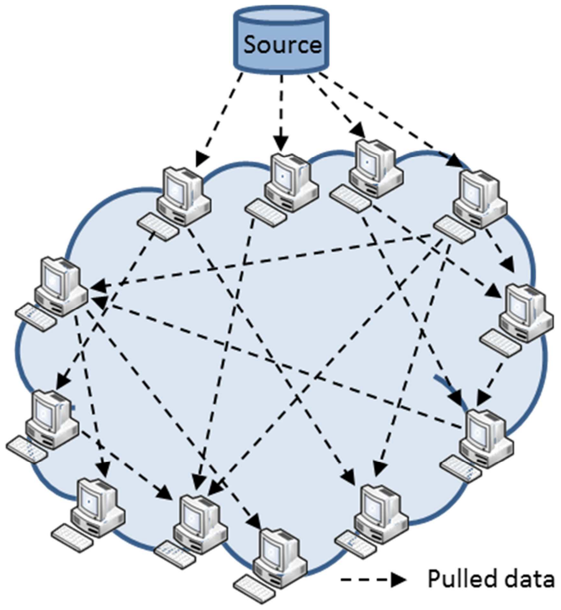


Figure 5: pull-based transmission strategy

2.2.2.2.3.2 Pull-based

In the pull-based strategies (Figure 5), scheduling choices are taken at receiver peer; a peer pulls the video content only if the content is of concern. This category of pull-based systems is natural when the peers are independent and self-interested. A pull-based scheme proposed by Chainsaw [49] is based on random scheduling approach. For each missing block, each peer randomly selects a peer from different neighbor peers who hold the block, and then ask it from the selected peer. Nevertheless, because content distribution still accepts ample randomness and uncertainty, it is very uncertain to guarantee the dissemination of each data block to all the peers before they playback deadline. According to round-robin approach [63], all the requested chunks are scheduled to one peer uniformly in a fixed order. This approach can reach good equilibrium of load, but it is adopted only in P2P live streaming systems to be in need most peers having desired content.

Most proposed mesh-based P2P streaming systems, like PeerStreaming, CoolStreaming [65] and AnySee [66], assume an intelligent pull gossiping algorithm: every peer exchanges content availability information periodically with its neighbors and then reclaim required chunks from a subset of peers. Specially, in CoolStreaming, the algorithm computes the number of potential providers for each chunk and, starting from the chunk with only one potential provider, it selects the provider with the highest upload bandwidth and sufficient available time in case of multiple provider. Though such a pull-based approach is more robust to peer crescendos than push-based approach, it unavoidably growths the delay of content transmission from source to all peers, because of delays produced by periodic exchanges of chunks disposal.

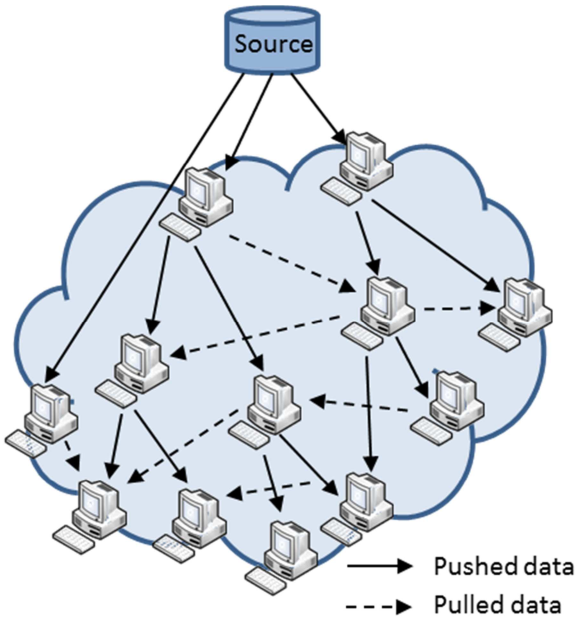


Figure 6: push-pull transmission strategy

2.2.2.2.3.3 Push-pull Based

While push-based solution causes an important overhead, as a peer may get the same chunk many times, a pull-based strategy causes intolerable latencies. Therefore, there is a trade-off between efficacy and overhead. So push-pull based approaches (Figure 6) are proposed in [45] [67] [68] [69]. Zhang et al. [67] propose a mechanism named GridMedia. It is based on a push-pull strategy that resides in demanding chunks in pull manner at the beginning and requesting nodes relaying chunks in push mode in the close following stage. Authors in [68] proposed a Push-to-Peer strategy; it is composed of two phases periodically: push stage and pull stage. PRIME proposes the two stages to the transmission phase and the swarming stage, where the new content is rapidly transmitted to the whole system in the first stage, and peers exchange their buffer maps in the second phase. Propositions in [69] differ from all these strategies in such a way that it uses a structured overlay, based on a prefix-routing partner selection approach. Their strategy further uses new push algorithm personalized precisely for prefix-routing-based architectures to rapidly distribute the video content, thus significantly minimizing the end-to-end delay observed in other pull-based mechanisms. Therefore, the push-pull based mechanisms have the advantages of the pull-based mechanisms and in addition have the efficiency of push-based mechanisms.

2.2.2.2.3.4 Coding for Distributed Transmission

Many other mechanisms for chunk scheduling are proposed by chunk coding for distributed transmission. Some contributions recently proposed approaches leverage network coding to enhance the bandwidth utilization [70] [71] [72] [73] [74]. Using network coding, each peer makes a linear combination on available blocks and transmits the combined blocks to its neighbor peers. When a peer receives sufficient linearly independent combinations, the original video can be rebuilt. [70] takes advantage of the random network coding in P2P data delivery. In [71] and [72] P2P live streaming systems adopt random linear network coding. Since chunks have a playback deadline, rather than encoding all available chunks, the Deadlineaware Network Coding strategy suggested in [73] regulates the coding window size for each peer on the basis of its network conditions and playback deadline in such a way to avoid chunk miss to address the challenge that some chunks may miss the play deadline before being transmitted to the decoder. Authors in [75] in addition using network coding to solve scheduling problem within a chunk, they also resolve the problem of scheduling across chunks. In addition to network coding, many other coding strategies were also proposed in P2P streaming, like MDC (Multiple Description Coding), FEC (Forward Error Correction), etc.

2.2.2.2.4 Discussion

Some of the scheduling mechanisms presented in this section are currently part of the most protuberant implemented P2P streaming systems. Table 1 summarizes of the main characteristics of these mechanisms.

P2P System	Topology	Chunks selection	Scheduling strategy
GridMedia	mesh	rarity urgency	push-pull
CoolStreaming	mesh	rarest first	pull-based
P2Cast	tree	sequential	push-based
GridCast	mesh	anchor-based	pull-based
PPLive	mesh	rarity urgency	pull-based

Table 1: P2p Streaming Systems Characteristics

Nevertheless, there are still many concerns that need to be fixed in order to make of P2P streaming the first media distribution solution. In specific, the cooperation strategies that are well studied for file sharing systems but not applicable to live streaming system because of the playback deadline. Consequently, these strategies should be adjusted to P2P streaming systems rather than simply use older approaches. Some strategies use chunks of equal size in scheduling algorithms, in which chunks limits do not match GoP (Group of Pictures) boundaries. Only GoPs that are delimited by a piece border can be played. Simple mapping between chunks and GoPs should be established. In addition, most of the studied mechanisms don't consider the specificity of layered streaming mainly the layers dependency. Indeed, chunks of higher layers received without their corresponding lower layers chunk are useless even received before their playback deadline. Therefore, there is a need for a mechanism to ensure receiving the lower layers chunks earlier in such a way to minimize the useless received chunks, and ensure the playback continuity even with minimum available quality.

Finally, it should be pointed out that the ultimate goal of chunks scheduling algorithms is to fully utilize the bandwidth allocated among peers. Consequently, the scheduling algorithm performance depends closely on how to use peer's upload bandwidth and how to distribute it among the requester peers. Hence the chunk scheduling mechanism should be prior-strengthened by an efficient bandwidth allocation mechanism.
In the next section we present the bandwidth allocation problem in P2P streaming system and the key solutions proposed in the literature.

2.2.2.3 Bandwidth allocation in P2P video streaming

P2P overlays are distributed systems where autonomous peers contribute a portion of their resources to cooperatively distribute data in an efficient and scalable manner. In P2P architectures, peers' upload bandwidths are the most important resource. How to allocate upload bandwidth among neighbor peers is an important issue. Nevertheless, in P2P streaming architectures, it is more challenging to design an efficient bandwidth allocation scheme. First, nodes have different uploading capacities and they are dynamically joining and leaving the overlay, which causes the system capacities to be disturbed and unstable. Secondly, nodes are strategic and selfish. This is one of the main challenges for the design of incentive mechanisms, which look for align the utility of the system with the utility of individual nodes.

The first bandwidth allocation systems for P2P streaming systems have been addressed mainly from the fairness viewpoint. The main objective was to equitably allocate bandwidth to peers without taking into consideration neither the malicious behavior of some peers nor the heterogeneity of peers. This class of bandwidth allocation strategies referred to as *Fairness-based bandwidth allocation.* In the other hand, several recent bandwidth allocation mechanisms look for aligning the utility of the system with the utility of individual peers (referred to as *Incentive-based bandwidth allocation*). In the following section we present the pioneer works from both classes in the context of P2P streaming systems.

2.2.2.3.1 Fairness-based bandwidth allocation mechanisms

While many P2P streaming systems have been deployed in practice and lots of research has been actively conducted on various issues in these systems, resource allocation and utilization have mainly been tackled from the fairness viewpoint. These strategies themselves have different resource allocation goals and enhance the available resources utilization mainly through admission control. The admission control mechanisms can be classified into four categories: Best-fit, Preemption-based, Random-selection and Reputation-based strategies. The Best-fit strategies, adopted in P2Cast [37] and ESM [76] select several existing peers with the largest available bandwidth for the new joining peer. Once joining the overlay, each peer selects nodes with the largest available bandwidth to maximize its download bandwidth. Upon the arrival of a new node, the random selection algorithm often selects a number of peers that can satisfy the bandwidth requisite of the new node. In [77], [78], [79], a partial list of existing nodes may be constructed from a centralized node (tracker) and the new node selects the downstream peers randomly from the list. Preemption-based strategies [80], [76] first select a number of nodes as the downloading nodes of the joining node. If these downstream peers do not have the required bandwidth and the bandwidth supply of the joining peer is bigger than some existing peers, some of the existing peers in the network will be prevented and their preempted bandwidth will be reallocated to this newly joining peers. Reputation-based mechanisms [81] [82] deliver different amount of bandwidth to existing nodes depending on their rank. As all nodes' rank is decided over history, it takes long time to reach the equilibrium and precise rank and their rankings are depending on the dynamicity of the system. Whereas these mechanisms have significantly simplified the bandwidth allocation problems and system implementation, an important part of available bandwidth have potentially been lost.

2.2.2.3.2 Incentive-based bandwidth allocation mechanisms

The normal behavior of a P2P streaming system depends on the cooperation degree between participants' nodes. Without adopting any incentive mechanism, rational peers may try to increase their utility by forbearing to announce their content to their neighbors or directly refusing blocks requests when their neighbors' peers request media blocks to them. Even when P2P streaming systems use their own protocols, it is still potential for peers to bound their upload bandwidth by, such as PeerGuardian [83]. Therefore, it is essential to integrate incentive mechanisms into P2P streaming systems to encourage peer cooperation and penalize free riders peers. In comparison with incentive mechanisms in P2P file sharing architectures, building incentive mechanisms for P2P streaming applications face two main defies: (a) strict playback deadline and (b) high bandwidth need.

There has been intensive research work in designing incentive mechanisms for P2P file-sharing architectures [84] [85] [86]. Nevertheless, only a few works focus on the incentive mechanisms for

P2P live streaming systems. Nowadays, there are mainly four types of incentive mechanisms for P2P streaming architectures: reciprocity, micropayment, reputation and taxation-based mechanisms.

In the reciprocity-based approach [87] peers can exchange their content only when both of them have the content interested by their corresponding peer. However such approach reduces the chance of content exchange in the system. Authors in [88] propose a mechanism where peers determine if their neighbors are free riders or not by examining exchanged control messages. This strategy can solve the problem where a peer with small upload bandwidth could be considered as a free rider, but it will increase the bandwidth consumption in the network. In [89] [87] authors consider the design of tit-for-tat incentive mechanism for MDC P2P streaming architecture. With MDC, even peers with low upload bandwidth can contribute their bandwidth, and experiment high QoS if they contribute more.

The other strategy to provide incentive in the P2P streaming architectures is micropayment-based strategy. [90] [97] [98] [99] propose the notion of virtual currency. It can be exchanged between peers and enhance the chance of content distribution. The system requires a central entity to manage and track the virtual currency, and therefore it cannot scale for a large number of users. In [100] exploits a detached self-managed currency to ensure transactions among nodes. All nodes can buy the currency from the tracker. Trades can be achieved between peers by trading currency. In [101] authors propose to incorporate proxies into the architecture, which can buy virtual currency from the tracker and then allocate to peers. Such architecture can help to minimize the tracker workload.

The third strategy is based on reputation mechanisms. The idea behind reputation-based strategies is that a node can value its neighbors by a metric named reputation score. This metric can be computed using the history of interaction between peers. Once a peer receives requests from its neighbor peers, it can decide which peer it will serve on the basis of their reputation. Author in [102] proposed a new approach where a peer receiving resources from its neighbors, it sends them messages about services that they realized. In addition, a peer checks its received messages to decide if its neighbors are free riders. In [90] authors proposed an incentive mechanism based on online time and effective upload bandwidth.

The fourth incentive mechanism is taxation-based strategy. In the above-mentioned strategies, low-bandwidth nodes suffer from low quality due to their limited contribution in the network. By adopting different tax rates for different nodes, high-bandwidth nodes should contribute more bandwidth and help to enhance the performance of low capacity peers. Such category of redistribution enhances the performance of the whole system.

2.2.2.3.3 Conclusion

In this section, we have presented a brief state of art for the classification of bandwidth allocation mechanisms in P2P networks along with some of the pioneer mechanism in the literature. We propose to classify the bandwidth allocation mechanism in this context to fairness-based mechanism which considers all peers are of equal importance in the network without considering neither heterogeneity nor the malicious behavior of some peers. The second category is incentive-based. The aim goal of this class is to combat the free riders peers that try to enjoy the resources of the peer-to-peer system without giving anything in return, or giving the minimum. In this perspective we can see clearly the difference between the two classes. The first one tries to manage and deal with the available bandwidth in the system, and try to best take advantage of it. Whereas the second classes tries to incite peers to contribute in the system and by the way increase the available bandwidth in the system. In our contribution we aim to allocate the available bandwidth in the network while taking into consideration the heterogeneity of peers and nature of the layered streaming in P2P systems. The ultimate goal is to best take advantage of the available bandwidth and enhances the system streaming quality. The incentive mechanisms are out of scope of our proposal.
In the next section we tackle the third component of the P2P streaming systems namely the content adaptation.

2.2.3 Content adaptation

Today's Internet is connecting millions of clients using heterogeneous terminals and through heterogeneous networks. The convergence between existing and emerging technologies such as broadband, mobile, and broadcast networks is considered as a new challenge to overcome for creating a new, open and flexible platform for the delivery of media over all type of networks. In such pervasive media environments, content consumers are demanding their content to be accessible from any available home or portable devices (PC, TV, Notebook, PDA, Cellular phone, etc.) connected through different heterogeneous networks. However, to deliver the multimedia content in accordance to characteristics of distinct consumers connecting through different networks is challenging due to different constraints including available bandwidth, display capability, CPU speed, battery constraints, user preferences, etc.

Content Adaptation [91][92][93][94] is the key technique that is widely used to address the above issues. Content Adaptation is a set of technologies that can be grouped under the umbrella of the Universal Multimedia Access (UMA) concept. This refers to the capability of accessing to rich multimedia content through any client terminal and network. Generally, the ability to customize/personalize any requested media content in real-time is called "adaptation". The objective of "adaptation" is to encode/modify the original video content in such a customized way that can be used "anytime" from "anywhere" (using any access network) and by "anyone" (using any terminal capability).

The multimedia content delivery over heterogeneous networks suffers from different challenges [95] [96] that affect directly the perceived Quality of Service (QoS). The channel bandwidth is the first problem that impacts directly the perceived QoS. The available channel bandwidth between the receiver terminal and the content server is generally unknown and has a time-varying characteristic.

The second problem is packet loss which occurs, in general, in network element such as routers or in the access network due to channel interference and fading problems. The router queue can be overloaded by short term burst traffic leading to packet drop. Transport-level congestion control mechanism can alleviate this problem but does not avoid it. To deal with packet loss issues, content delivery applications must be designed with error control capabilities in mind.

However, all these problems can be tackled efficiently with dynamic content adaptation mechanisms. The adaptation of the multimedia content according to changing usage environments during the service delivery is becoming more and more important. That is, the characteristics of the environment where the actual multimedia content is consumed (e.g., network, terminal, and user characteristics) are varying during the consumption of the (multimedia) service. Thus, immediate actions are needed to be performed in order to enable a unique, worthwhile, and seamless multimedia experience during the entire session lifetime. These kinds of actions are generally referred to as dynamic multimedia content adaptation [110] [111].

In yet another scenario where different end users are interested in the same multimedia content but with different usage environments, optimal resource utilization is achieved by transmitting the multimedia content to an intermediate peer, between the source peer and the consumer peer such that the offered service satisfies a set of usage environment constraints common to all consumer peers. On receipt of the multimedia content, the intermediate peer adapts and forwards the multimedia content satisfying the individual usage environment constraints of each end user (receiver peer). This kind of approach where multiple adaptation steps are successively performed within the delivery path is referred to as distributed multimedia content adaptation [112].

The content adaptation can be performed at different epochs and at different levels of service lifetime. Regarding the epoch, content adaptation can be performed during:

- The service invocation phase along with service personalization for example used in Video on Demand (VoD) media streaming. This phase takes into consideration user profile, terminal profile, static network conditions, etc.

- The service delivery phase based on dynamic network conditions and/or feedbacks coming from the network, from the content distribution block or from the terminal (e.g. perceived quality feedback)

Regarding the levels, content adaptation can be performed as:

- Application-level adaptation (e.g. transcoding, transrating, scalable encoding, etc.) and protocol adaptation (streaming over RTP/UDP, MPEG-2 or HTTP)
- Network-level adaptation (e.g. DiffServ packet marking, re-routing etc.) which is provided by the content distribution
- Cross-Layer adaptation which performs adaptation at different level and using different parameters

In general, adaptation is necessary in the case of shortage of resources and can be performed at different levels based on the estimated end-to-end constraints. Such estimations are based on the network feedback and/or end-to-end feedback. For example, the content server (or an Adaptation Gateway) adapts its sending rate according to the available estimated bandwidth. In fact, to deal with the long-term bandwidth variation, it is necessary to choose the best codec that can generate packet at certain target rate. Short-term bandwidth variation can be managed with some specific content adaptation mechanism (e.g. transrating, transcoding, etc.). Transcoding is the conversion of media content from one digital format to another format, for example from MPEG-2 to H.264.

Transrating is changing the bitrate of video stream to meet the requirements of the network or end-user device. The downgrading of the resolution, for example conversion of media content from high definition (HD) to standard definition (SD) is an example for such techniques.

Whatever the adaptation is, its main goal is to enhance the delivered video quality and to enable the terminal to access the content which was initially not designed for.

This section is organized as follow. We start first by presenting the different video coding standards. The focus will be on the scalable video coding (SVC) that we adopt in our work. Then we give an overview on three key questions that should be tackled in any P2P streaming adaptation architecture, namely where, where and when to adapt? Finally we give an overview on the SVC layer selection strategies.

2.2.3.1 Video Coding Standards

P2P networks are expanding quickly as a heterogeneous communication networks. The number of users is growing exponentially and they are using services that need high bandwidth requirements such as media streaming. These heterogeneous clients have variant uplink and downlink bandwidth capabilities. Furthermore, high popularity of video sharing across the networks arise the issue of networks congestion. There are a number of problems that affect video streaming. Current Internet infrastructure (IP-Internet Protocol) provides best effort services that do not offer quality of service. These problems result into (1) lower throughput (bandwidth management), (2) higher and random packet losses, (3) high transfer delay, and (4) delay variation (jitter). These parameters are unpredictable and never acceptable for real-time applications. Thus, we need to design solutions for efficient video streaming over P2P networks that can address the abovementioned issues.

The selection of an appropriate codec is vital and plays an important role to deal with the problem of bandwidth management. The best codec can produce video at certain rate and when the channel conditions change, it applies an adaptive behavior.

There exist a number of audio/video coding standards for video content delivery over IP networks. The ITU-T and the ISO/IEC JTC1 are the two organizations that develop video coding standards. The ITU-T video coding standards are denoted with H.26X (e.g. H.261, H262, H.263 and H.264). The ISO/IEC standards are denoted with MPEG-x (e.g. MPEG-1, MPEG-2 and MPEG-4).

The ITU-T standards have been designed essentially for real-time applications, such as video conferencing, while the MPEG [113] [114] standards have been designed mostly to address the needs of video storage (DVD), broadcast video and video streaming applications. For the most part, the two standardizations committees have worked independently on the different standards. The exceptions are H.262/MPEG-2, completed in 1994, and H.264 (also called MPEG-4 Part 10 [115][116] or MPEG-4 AVC) finalized in 2003 and H.264/SVC completed in 2007 as shown in Figure 7.

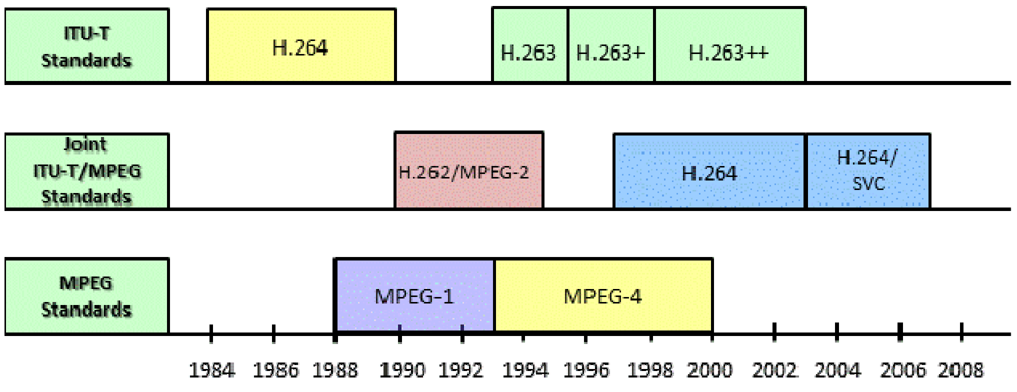


Figure 7: Progression of the ITU-T Recommendations and MPEG standards

In the next sub-section, we will present in more details the H.264/SVC encoding standard.

2.2.3.1.1 Scalable Video Coding (SVC): H.264/SVC

Scalable video coding (H.264 SVC) is a set of new scalable extensions for H.264 standard that is considered the most promising video format for media streaming over heterogeneous networks [115][118][119]. Specified in Annex G of H.264/AVC, SVC allows the construction of bitstreams that contain sub-bitstreams that can be consumed by heterogeneous clients. A scalable video coding is capable to produce highly compressed bitstreams, in order to create a wide variety of bitrates.

In SVC encoding scheme, each video stream is encoded in multiple video quality layers. Each layer can be decoded to provide different video characteristics. The first layer provides the basic quality of the video is called "Base Layer" while other layers, which are used to enhance the overall video quality of the base layer, are called "Enhancement Layers" [120] [122][123].

An original SVC stream can be truncated to produce video of different qualities, sizes, and frame rates, i.e. in SNR (Signal to Noise Ratio), spatial and temporal dimensions. This scalability makes SVC bitstreams suitable for heterogeneous networks and terminals to meet the QoS requirements restrictions often encountered by the streaming applications. In SVC stream, the base layer is encoded using a fully standard compatible H.264 AVC (Advanced Video Coding). Then enhancement layers can be added, each providing temporal, spatial, or SNR scalability. The SVC format has the ability to provide the decoder with different enhancement layers depending on reception of the base layer and lower enhancement layers. Thus a transmission scheme should ensure that these layers are transmitted such that packet loss is kept as low as possible even for high overall transport packet loss rates. Besides the ability to adapt to different heterogeneous networks and terminals, SVC can also achieve graceful degradation of video quality in case of packet loss and high end-to-end delay, as the decoder will successfully decode the stream even in the absence or late arrival of some layers.

The objective of the SVC standardization has been to enable the encoding of a high-quality video bitstream that contains one or more subset bitstreams. Those sub-streams can themselves be decoded with a complexity and reconstruction quality similar to that achieved using the existing H.264/AVC design with the same quantity of data as in the subset bitstream. Hence, it enables the transmission and decoding of partial bitstreams to provide video services with lower temporal or spatial resolutions or reduced fidelity.

The term "scalability" refers to the removal of parts of the video bitstream in order to adapt it to the various needs or preferences of end users as well as to varying terminal capabilities or network conditions. Apart from the required support of all common types of scalability, the most important design criteria for a successful scalable video coding standard are coding efficiency and complexity. Since SVC was developed as an extension of H.264/AVC with its entire well-designed core coding tools being inherited, one of the design principles of SVC was that new tools should only be added if necessary for efficiently supporting the required types of scalability.

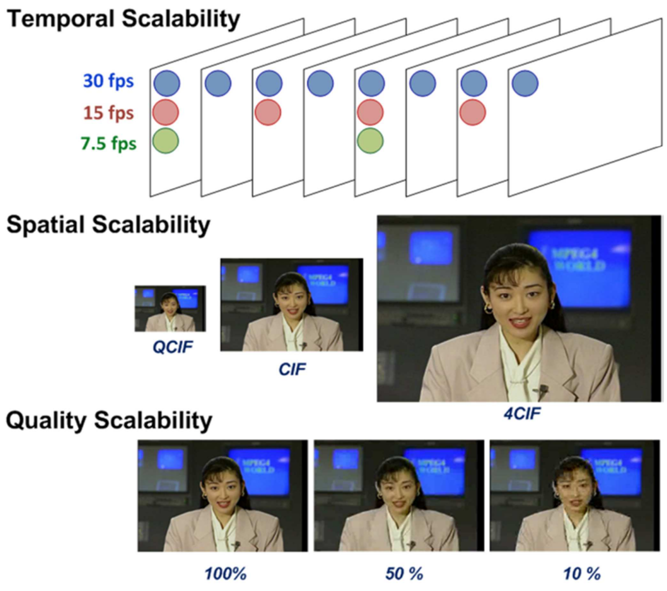


Figure 8: The basic types of scalability in video coding

Spatial scalability and temporal scalability describe scalability approaches in which subsets of the bit stream represent the source content with a reduced picture size (spatial resolution) or frame rate (temporal resolution), respectively. With quality scalability, the sub-stream provides the same spatio-temporal resolution as the complete bit stream, but with a lower fidelity, where fidelity is often informally referred to as signal-to-noise ratio (SNR). Quality scalability is also commonly referred to as fidelity or SNR scalability. The different types of scalability can also be combined, so that a multitude of representations with different spatio-temporal resolutions and bitrates can be supported within a single scalable bit stream.

After presenting the Scalable Video Coding (SVC) standard that will be adopted along with our three contributions as the layered streaming encoding system, we present in the next section the description of the adaptation process in the P2P streaming systems, by asking the three following questions: where? When? And how to adapt the content?

2.2.3.2 Where to Adapt the Content?

An important question raised in the content adaptation in P2P streaming systems is where to adapt the content in case of distributed and heterogeneous overlay organization [124] in which different peers act as client consuming the content or servers providing the content. Some of them may also be just simple nodes relaying the content to other. In this case, the adaptation process can be performed at different levels: at the content source, at the content consumer or at a relaying node level. In this sub section we will present briefly the different alternatives.

2.2.3.2.1 Adaptation at Original Content-Source Level

In this approach, the adaptation is performed at the content provider (Figure 9) which has significant knowledge about network and terminal capabilities in order to decide and execute efficient adaptation. This approach provides more control over the multimedia content by limiting the alteration of the content owner. However, this approach is not scalable because the content provider takes into consideration of both the content provision and content adaptation. In addition, the adaptation of content is resources consuming (CPU, memory, etc.) which limits the number of consumers that can be supported by content provider. Also, the original quality (without adaptation) cannot be delivered as it to some capable receivers if an adaptation is applied.

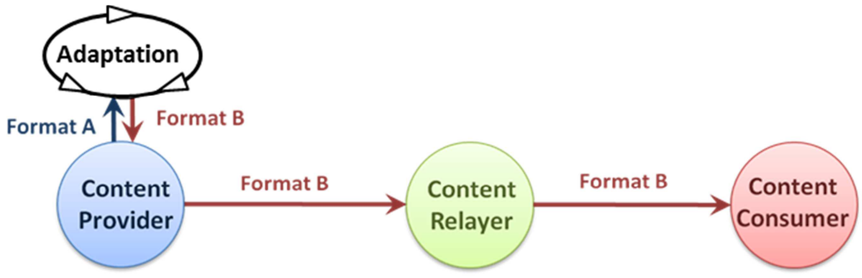


Figure 9: Adaptation at Original Content-Source Level

2.2.3.2.2 Adaptation at Consumer Level

In this approach, the adaptation is carried out by the content consumers (Figure 10) on the basis of some capabilities. The advantage of this approach is that the capabilities of the terminal are well known to the adaptation engine. The proposed smoothing mechanism (section 5) falls into this category of adaptation mechanisms.

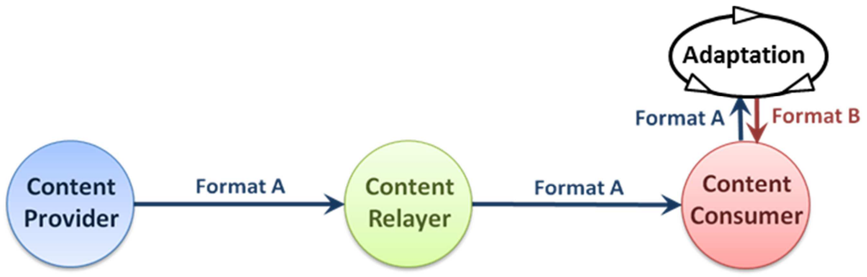


Figure 10: Adaptation at Consumer Level

2.2.3.2.3 Adaptation at Gateway Level or Intermediate Node

In this approach, the adaptation is performed at a dedicated intermediate peer inside the network that is located between the source peer and the consumer peer (Figure 11). This approach reduces the adaptation burden on content server and facilitates the content adaptation following the network characteristics. However, this approach adds significant overhead while performing adaptation of secure media content because it requires decrypting the content before adaptation and then the resulting adapted content should be encrypted again to be delivered to the consumer.

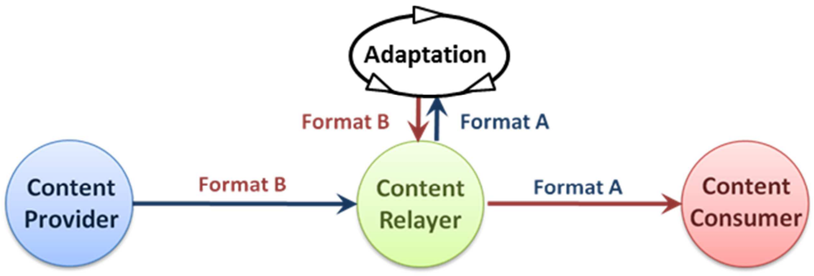


Figure 11: Adaptation at Gateway Level or Intermediate Node

2.2.3.3 When to Adapt the Content?

The epochs of service are also another important aspect to consider in the adaptation process. This is about the timescale related to different stage of the service, i.e. service invocation and service delivery/consumption. The adaptation scale should be accurate to reflect any change in the user context such as degradation of network conditions, changing in usage environment, etc. Adapting the content too often may not be a good solution for providing a good quality of experience as the quality may vary considerably over the time. Whereas, adapting the content irregularly (for example too late after detecting the problem) may affect considerably the received quality. So timescale has to be carefully taken into account to efficiently deliver a smooth [125], stable over the time, and acceptable level of quality of service/experience.

2.2.3.4 How to Adapt the Content?

The last question related to content adaptation concerns the techniques that may be used for content adaptation. It can be achieved using transcoding, transrating, and/or SNR adaptation techniques [126][127]. However, emerging of new video encoding standards as SVC (2.2.3.1.1) has opened up new horizons for adaptation which consist of simply dropping of enhancement layer. Yet which layers to drop is considered as a challenging problem and need to be investigated further as many possibilities exist when layer are dropped.

2.2.3.4.1 Codec Adaptation (Transcoding)

This adaptation function allows the transcoding from a certain format to another one. For example, an original MPEG-2 content can be transcoded to SVC to enable scalable content distribution for heterogeneous consumers with different profiles. Similarly, SVC content can be transcoded to an MPEG-2 in order to serve a legacy terminal which accepts only MPEG-2 format.

2.2.3.4.2 Protocol Adaptation

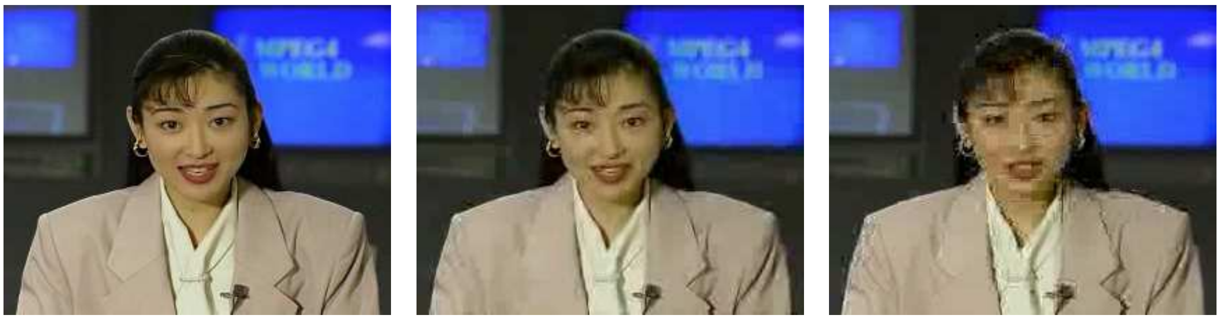
Different content may be transmitted via different protocols. The adaptation service can include a protocol adaptation which allows clients that do not talk a particular protocol to access the service. For example, an end user behind a firewall would not be able to receive a content using RTP with RTSP signaling as the necessary ports may be blocked by the firewall. Thus, an intermediate node (i.e. service node performing protocol adaptation) involved in a distribution of a content (that is pushed using RTP by the content provider) may be switched to HTTP delivery in order to serve this user which is behind a firewall. A signaling mechanism is very important as the service may be delivered in a non-transparent manner.

2.2.3.4.3 Bitrate Adaptation

The bitrate adaptation in video streaming context is the modification of the stream bitrate in order to meet the network bandwidth requirements or the receiver device capability/preferences, while keeping a QoS/QoE above a certain threshold. This adaptation can be performed by reducing or enhancing the fine-grained video quality (Figure 12). It is also known as the SNR (Signal to Noise Ratio) adaptation. The bitrate adaptation can also be performed by reducing/increasing the spatial resolution of the video, i.e. spatial adaptation (Figure 13). Finally, the bitrate adaptation can be performed by deleting some frames from the video (Figure 14), and consequently reducing the framerate, defined as the number of frames played per time unit. This is called temporal adaptation. In the majority of the cases, SNR adaptation is more suitable since it allow to easily targeting the desired bitrate while maintaining an acceptable level of user experience. Also, reducing frame rate of some video content (football match, action movies, etc.) may affect considerably the user experience. In this case, it is preferable to have a lower SNR than lower frame rate.

2.2.3.4.3.1 Quality (SNR) Adaptation

The quality adaptation (Figure 12) or Signal to Noise Ratio adaptation process relies on the principle of image compression techniques. An example of quality adaptation is given in Figure 12.



100% Quality **50% Quality** **10% Quality**

Figure 12: Quality (SNR) Adaptation

2.2.3.4.3.2 Spatial Adaptation

The spatial adaptation (Figure 13) is based on image scaling (upscaling/downscaling). An example of special adapataion is given in Figure 13. This process relies on some techniques of interpolation such as Nearest Neighbor Interpolation, Bilinear Interpolation and BiCubic Interpolation.

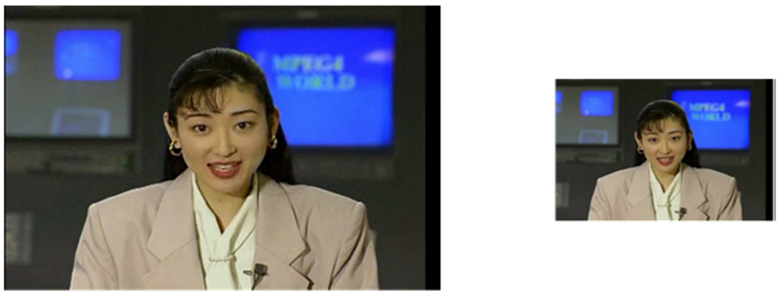


Figure 13: Spatial adaptation

2.2.3.4.3.3 Temporal Adaptation

The temporal adaptation (Figure 14) of a video content is performed by dropping some intermediate frames. The selection of frames to be dropped should be achieved taking in consideration the frame importance in the video.

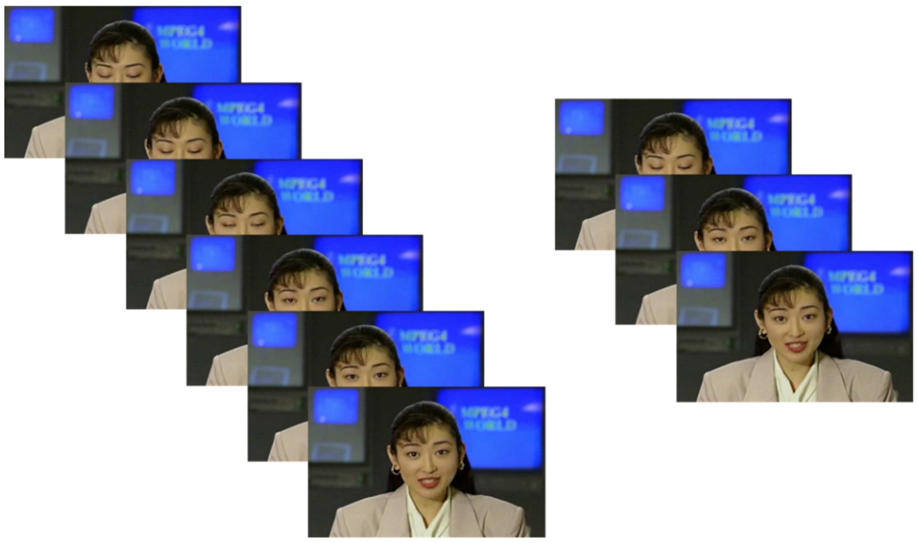


Figure 14: Temporal adaptation

2.2.3.4.4 SVC-based Layer Selection Adaptation

In non-layered streaming, maximizing the peer delivery ratio is almost equal to maximizing the throughput, which is not the case in layered streaming. In layered streaming, subscribing-to many layers can result in poor video quality due to the layer dependency. For example, if some packets of lower layer are missed, the depending packets in the upper layer cannot be correctly decoded and thus become useless. Therefore, the selection of layers is an important consideration for achieving higher video quality.

The selection of SVC layers can be carried out at three different levels. The layer selection can be done at Source, Network or/and Receiver level. The source node can select appropriate SVC layers based on the user context (User preference, Profile, Bandwidth availability, etc.).

At network level, the higher SVC layers may be dropped due to limited bandwidth or network congestion. The network entity can drop higher layer in order to ensure service continuity. Furthermore, the SVC layers can be transmitted with different priorities by the content source. These priorities are mapped to the different classes of service in the network. For example, packets of the base layer are marked to the higher class of service compared to enhancement layers.

On the receiver side, high layer delivery ratio and low useless packets ratio can be achieved by properly selecting the layers (pull-bases scenario). The decision for the selection of appropriate layers at the receiver level depends highly on available bandwidth. Once the layers are selected, the missing data units in subscribed layers should be fetched in a reasonable way when they are close to their playback time.

In order to provide the best quality to the user, a trade-off between the scalability of the three dimensions should be considered. For example, if bandwidth shortage occurs during the streaming session the user can switch to a lower bitrate by lowering the quality level. If another decrease in bandwidth occurs, the user may decrease the temporal scalability and so on. These steps define an adaptation trajectory also called an adaptation path [121]. Examples of an adaptation trajectory are shown in Figure 15.

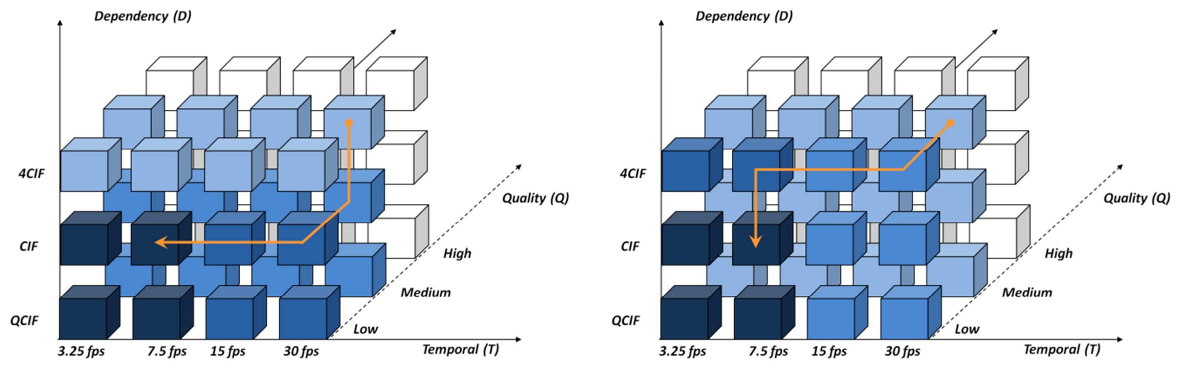


Figure 15: Examples of SVC Adaptation Trajectory

The adaptation problem that we are tackling in this dissertation is rather the fluctuation of the quality level due to the fluctuation in the download bandwidth of the peer and its impact of the QoE perceived by the end-user. The SVC adaptation path is out of scope of this work.

2.2.4 Conclusion

In this chapter, we have presented a brief state of art for the three components of video streaming systems over P2P networks, namely the overlay construction, content delivery and content adaptation. We focused on the two later components. In the content delivery component we have presented three content delivery modes: live, on-demand and time-shifted, then we went through the scheduling and bandwidth allocation mechanisms in P2P streaming systems. We classified the scheduling mechanisms into two categories. The empirical-based and the optimization-based mechanisms. In the first category, algorithms are simple to deploy even in low capacity devices however their performances are limited and sometimes random. On the opposite, the optimization-based scheduling mechanisms guarantee a certain level of performance, to the detriment of important requirement in terms of memory and CPU resources consuming. In our contribution related to scheduling, we propose a tradeoff between the two categories, an optimization-based scheduling algorithm with reasonable resource consuming.

After that, we classified the bandwidth allocation mechanisms proposed in the literature into fairness-based mechanisms where all peers are considered equal regardless their performance and capacities, and without taking into consideration their contribution in the network. In the second category (incentive-based), peers are served regarding their involvement in the network. The second key difference between the two categories is that the fairness-based deals with the available bandwidth in the network. However the incentive based aims to increase the overall bandwidth in the network. The incentive mechanism is out of scope of our proposed bandwidth allocation mechanism. Nevertheless, we take into consideration the capacity of the peers in terms of download and their priority in the process of allocation. The priority of peer can be eventually determined by an incentive mechanism.

Finally we presented an overview on content adaptation in P2P streaming, mainly we investigated when, where and how to perform the adaptation. The focus is given on the adaptation using scalable video coding (SVC) that we adopt in our works on scheduling, bandwidth allocation and smoothing. This choice is justified by the promising characteristics of SVC and real-time content adaptation support for the heterogeneous networks and terminals.

The proposed scheduling mechanism for real-time streaming is described in following chapter.

Chapter 3

3 Efficient Scheduling Mechanism for Layered and Non-Layered Streaming in P2P Networks

3.1 Introduction

In P2P video streaming systems, the content retrieval mechanism allows a user to receive streaming data blocks (chunks) from other nodes using the constructed overlay. This mechanism plays a leading role in the video streaming process and its efficiency influences the global performance. Two main approaches have been proposed: the pull and the push mechanisms. The pull mechanism is based on the chunks availability at peers: what chunks are available from which neighbor? Thus, a receiver node has to locate the missing chunks and to request them from the appropriate nodes. On the other hand, in the push mechanism, it is the sender nodes which crowd the chunks to the receiver node without any action from this later.

The pull mechanism is considered as very simple and suitable approach as it allows the receiver to cope with two main challenges: eliminating chunks redundancy and recovering from chunks loss. However, it adds complexity to the receiver side because it is responsible for designing the appropriate chunks to be selected from the appropriate neighbor.

The scheduling task is complicated in the context of video streaming since chunks received after their playback deadline are not played and considered as useless chunks. Moreover, in the context of layered video, the task becomes more and more complicated, since an additional constraint should be taken into consideration, namely the layers' dependency. Hence, in layered video coding, video is encoded into a base layer and several enhancement layers, where a higher layer can be decoded only if all related lower layers are available. This is what we call the layers' dependency.

In the literature many works are based on empirical studies for specific policies and heuristics (ref. section 2.2.2.2.2). Examples of this include a pure random strategy [49], Local Rarest First (LRF) [50] and Round Robin (RR) [128]. Apart from empirical studies, some works use queuing models for scheduling [129]. The algorithm proposed in [130] minimizes the base layer losses, but it assumes equal rates for the base and enhancement layers. This model of video is rather ideal and can be approximated only by fine grained scalability (FGS). Furthermore, a few theoretical studies tackle the optimal stream scheduling. Most of these works are under restrictive hypothesis or computationally expensive. In [131] a scheduler has been proposed to maximize the video quality by prioritizing the most important chunks. This strategy is particularly suited for push-based, tree-structured overlays. The scheduling mechanism proposed in LayerP2P [15] is able to save base layer losses to the detriment of the enhancement layers. Authors propose to categorize chunks request into two types: regular requests and probing requests. The regular request concerns the requests of layers lower than or equal to a threshold l_n, which are firstly assigned to different suppliers based on random scheduling algorithm (that authors believe it achieves a high system throughput) without any prioritization among different layers. Secondly, the probing requests (layer greater than l_n) are sent to the suppliers layer by layer, in ascending manner. The quality threshold l_n and the maximum quality level to be requested are decided based of the available download capacity on the receiver peer, by consequence it follows the bandwidth fluctuation without any smoothing mechanism.

The authors in [132] propose an optimal scheduling strategy to minimize the overall video distortion, but the approach is strongly related to the Multiple Description (MD) coding, which is less efficient compared with layered coding [133]. Zhang et al. [134] have discussed the scheduling problem in data-driven streaming systems. They define a utility for each chunk as a function of its rarity, which is the number of potential senders of this chunk, and its urgency, which is the time difference between the current time and the deadline of this chunk. They then use this model to transform the chunk scheduling problem into a min-cost flow problem. This algorithm, however, is computationally expensive and may not be feasible for live video streaming systems subject to strict deadlines on computationally-constrained devices.

Szkaliczki et al. [135] also address the chunk selection problem in streaming layered video content over peer-to-peer networks. The authors present a number of theoretical solutions to maximize the utility function of chunks that exist in the literature. However, their proposed solutions rely on the definition of chunk utility functions whose objective definition may be difficult in real-life scenarios.

In this chapter we present a new analytical model and its corresponding algorithms to deal with the chunks scheduling problem in Pull-based P2P video streaming, both in case of layered and non-layered video streaming. First, we propose a chunks prioritization strategy in order to represent the urgency of chunks and its layers dependency. Then, we model the problem as an assignment problem and we propose new algorithms to resolve it in order to fully take advantage of bandwidth capacity of the network and to meet the availability of chunks in neighborhood. The rest of this chapter is organized as follows: section 3.2 formulates the scheduling problem in P2P video streaming, section 3.3 models and presents the solution that we propose, section 3.4 presents and discusses the performance evaluation results, and finally, section 0 concludes the chapter by highlighting the key contributions.

3.2 Chunks Scheduling: Problem Statement and Formulation

The basic idea in Pull-based P2P video streaming is that the overlay is constructed in such a way to optimize some parameters such as the delay, the bandwidth, etc. Each node in the overlay is connected to a set of neighbors but it is up to the receiver node to ask the chunks from its neighbors.

To better explain the problem of scheduling in P2P layered streaming, we assume a mesh-based pull approach in which the receiver side buffer is organized into a sliding window (Figure 16) containing chunks of different layers. The chunks beyond the playhead position are denoted as the *exchanging window*; only these chunks are requested if they have not been received yet (the chunks whose deadline has passed will not be requested). Each peer periodically announces the chunks that it holds to all its neighbors by sending a *buffer map* (Figure 17), a bit vector in which each bit represents the availability of a chunk in the sliding window. Periodically, each peer sends requests to its neighbors for the missed chunks in its exchanging window. As long as its request remains in the exchanging window, chunks are re-requested if not received.

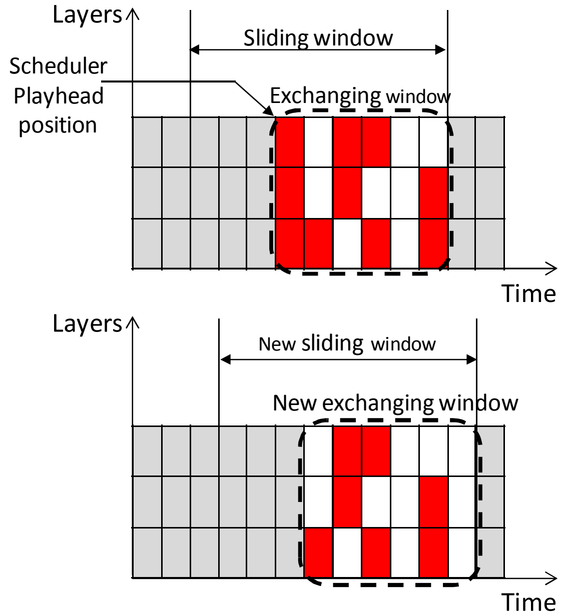


Figure 16: Sliding window mechanism

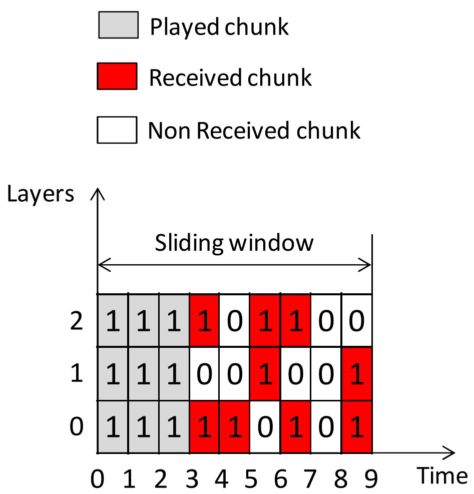


Figure 17: Buffer map structure

Of course, upper layer chunks received without the corresponding lower layer chunks are not decodable (and are considered useless, as described earlier). Thus, the chunks having time stamp T = 5 in Figure 17 are not played, because the base layer was not received.

In order to increase the throughput of the system, our approach aims to take full advantage of the download bandwidth of peers by maximizing the number of chunks that are requested within each scheduling period. Figure 18 illustrates an example of the optimal scheduling problem in terms of bandwidth utilization. For simplicity, in this example we consider a single-layer stream. Peer 1 is the receiving node, and it requests missing chunks from its neighbor peers 2, 3 and 4. Each neighbor advertises the chunks that it holds using a buffer-map. The numbers on the arcs denote the units of bandwidth that the neighbor peer is willing to provide to the receiver node (peer 1) in terms of chunks per unit time. An optimal scheduling scheme for this example is represented in Figure 19, where rows represent the peers and the columns represent the chunks. Chunk 1 is requested from peer 4, chunks 2 and 3 from peer 2, and chunks 4 and 5 from peer 3. This strategy takes full advantage of the available bandwidth of the network. In Figure 20, we represent the result of Round Robin scheduling strategy [128] applied to the same example. In this case, only 4 chunks out of the total of 5 can be requested in a single time unit.

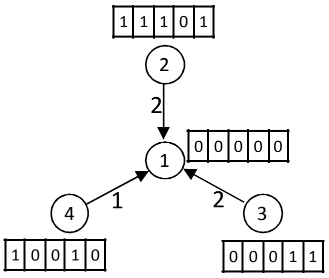


Figure 18: Example of the optimal chunk scheduling problem

Node \ Chunk	1	2	3	4	5
2	1	1	1	0	1
3	0	0	0	1	1
4	1	0	0	1	0

Figure 19: Optimal Chunk scheduling example

Node \ Chunk	1	2	3	4	5
2	1	1	1	0	1
3	0	0	0	1	1
4	1	0	0	1	0

Figure 20: Round robin scheduling example

The basic idea of our proposed mechanism is to select the appropriate enhancement layers based on the current quality level and an estimation of the available bandwidth for next time period. In addition, for each chunk belonging to a selected layer, we will define a *priority* on the basis of its playback deadline and its dependencies with other layers. This priority will then be used to guide chunk scheduling by requesting those chunks with higher priority first.

3.3 Model and Solution

The main goal of our scheduling approach is to efficiently request the missing chunks in the exchanging window of the receiver peer. This can be achieved by requesting the higher priority chunks before the lower priority chunks while at the same time taking full advantage of the available network capacity. Since, this scheme will closely depend on the definitions of these priorities, we now explain how they are calculated.

Intuitively, it seems clear that since chunks are useless if they are not decoded by their playback deadline, the priority of each chunk should be closely related to how close they are to it. Another issue to consider is the dependency between layers; a higher layer chunk received without its corresponding lower layer chunks will not be decoded. To factor these two variables into our priority model, we will define two functions. The first one, the *emergency priority* P_E, is a function of how close a chunk is to its playback deadline; the second one, the *layer priority* P_L, is a function of how many underlying layers are necessary to decode a particular chunk. Using these two functions, we can define our priority function P_{ij} as:

$$P_{ij} = P_E\left(T_i - D_j^i\right) + \theta P_L\left(L_j\right) \qquad \text{Eq. 3-1}$$

where T_i denotes the current time in the peer *i*, D_j^i denotes the playback deadline of chunk *j* in peer *i*, *L_j* denotes the stream layer to which chunk *j* belongs, and θ is a parameter that can be adjusted for different layers prioritization strategies. Hence, $P_E\left(T_i - D_j^i\right)$ evaluates P_E at a time interval equal to the remaining time that chunk *j* has until its playback deadline at peer *i*, and $P_L\left(L_j\right)$ evaluates P_L at an integer proportional to the number of underlying layers needed to decode chunk *j*.

Different values of θ can be used to implement different protocol behaviors. A small θ leads to the prioritization scheme represented in Figure 21(a), also called conservative chunk scheduling, where the receiver always requests chunks of lower layers first; a large θ leads to the aggressive chunk scheduling scheme represented in Figure 21(b), where chunks are requested on the basis of their timestamp only. Intermediate values of θ lead to tradeoffs between these two extremes; a particular example of this is shown in Figure 21(c).

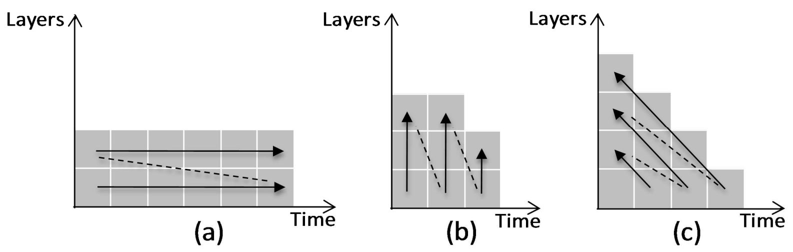


Figure 21: Scheduling strategies in case of layered streaming

We continue the presentation of our scheduling algorithm by defining R_{ij}^k, a Boolean variable that indicates whether the peer *i* requests the chunk *j* from the neighbor *k*:

$$R_{ij}^k = \begin{cases} 1, & \text{if peer } i \text{ requests chunk } j \text{ from neighbor } k, \\ 0, & \text{otherwise.} \end{cases}$$

We now present the core of our chunk scheduling heuristic. Using P_{ij} as defined in Eq. 3-1, we propose the *aggregate priority* Π_i of peer *i* as a figure of merit for our scheduling algorithm:

$$\Pi_i = \sum_{\substack{j \in M_i \\ k \in N_i}} P_{ij} R_{ij}^k \qquad \text{Eq. 3-2}$$

where M_i denotes the set of chunks that peer *i* requires from the overlay, and N_i denotes the overlay neighbors of peer *i*. Using this figure of merit, our scheduling problem can be formulated for each peer *i* as:

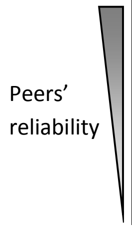
$$\text{Maximize:} \quad \Pi_i$$

$$\text{Subject to} \quad \sum_{j \in M_i} R_{ij}^k \leq C_i^k \qquad \text{Eq. 3-3}$$

$$\sum_{k \in N_i} R_{ij}^k \leq 1 \qquad \text{Eq. 3-4}$$

Where C_i^k denotes the download capacity of the link between the receiving peer *i* and its neighbor *k*. Our scheduling mechanism will therefore maximize a figure of merit that trades off chunk urgency with stream quality, subject to a constraint on total link capacity (Eq. 3-3). Equation (Eq. 3-4) ensures that a receiver peer does not request a chunk *j* from more than one neighbor. That means a chunk is not requested more than once. Usually, each chunk is requested from a single neighbor, unless it is not available in the neighborhood. In this case it cannot be requested.

This optimization problem can be naturally transformed into an Assignment Problem (AP) [136] where a set of missed chunks b$\in M_i$ in peer i are to be assigned to a set N_i of its neighbors. The assignment itself is captured with R_{ij}^k, the cost function for this assignment is $-\Pi_i$, and the feasibility conditions of the problem are (Eq. 3-3) and (Eq. 3-4). Therefore, in this case the set of chunks can be understood as a set of tasks which should be assigned to a set of agents (neighbors peers) while optimizing the overall cost, which refers to the priority sum of the chunks. In its original version, the AP involves assigning each task to a different agent, with each agent being assigned at most one task, i.e. *one-to-one assignment*. Since we want to assign one or more chunks to each neighbor, we will use an alternative formulation, the Generalized Assignment Problem (GAP) [137], that considers *one-to-many assignment* (multiple tasks can be assigned to the same agent). We therefore model the scheduling problem in layered streaming as a GAP, scheduling *m* chunks to *n* nodes (*m* ≥ *n*). This can be represented by the assignment matrix shown in Figure 22.



Chunk / Node	1	2	…	m-1	m
1	P_{i1}	P_{i2}	…	$P_{i(m-1)}$	P_{im}
2	P_{i1}	P_{i2}	…	$P_{i(m-1)}$	P_{im}
…	…	…	…	…	…
n-1	P_{i1}	P_{i2}	…	$P_{i(m-1)}$	P_{im}
n	P_{i1}	P_{i2}	…	$P_{i(m-1)}$	P_{im}

Figure 22: Assignment matrix-GAP

The GAP is known to be NP-hard problem [137]. In the following section, we propose a novel heuristic to approximate its solution, and use it to perform chunk scheduling in Pull-based P2P streaming systems.

3.3.1 Algorithm

In order to construct a solution for the scheduling problem in layered streaming, modeled as GAP, we consider an arbitrary algorithm (let say algorithm A) to provide solutions (approximate or otherwise) for small versions of the knapsack problem (e.g. the Harmony-search algorithm [138]). As a first step, we reorganize the rows of the assignment matrix based on neighbors' reliability (Figure 22) in order to assign chunks to the more reliable nodes first. Then, we perform the recursive procedure shown below. Of course, *j* is initialized to 0 as part of the algorithm setup. In this algorithm, N_i denotes the list of peer *i's* neighbors.

```
Assignment_Matrix_Line_Processing

1. Run the Algorithm A on the row j with respect to the capacity $C_i^k$ of node k and chunk size
   r. This will give the set of chunks from the most reliable peer that has not been considered
   yet, and which maximize aggregate priority. Let $S_j$ be this solution, i.e. set of selected chunks
   returned.
2. if $j < |N_i|$ (Termination condition)
        - $j = j + 1$. (Increment the indicator variable j, so that recursive calls to this function will
          consider the next row of A).
        - Perform Assignment_Matrix_Line_Processing(j) and let $S'$ be the returned chunks
          list. Return $S_j \cup S'$
   else
        Return $S_j$
```

Figure 23: Assignment matrix line processing algorithm

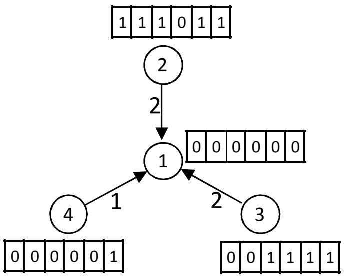
3.3.2 Processing overhead

The scheduling algorithm that we propose is based on a powerful knapsack algorithm, mainly the Harmony search algorithm. The results obtained using the HS algorithm may yield better solutions than those obtained using current algorithms [139], such as conventional mathematical optimization algorithms or genetic algorithm based approaches. The study performed in [139] suggests that the HS algorithm is potentially a powerful search and optimization technique for solving complex engineering optimization problems.

Finally, the amount of data processed in each scheduling period is very low. Indeed, the exchanging window size is very low, and the maximum number of neighbors considered in the simulation is not more than 30 neighbors, consequently the scheduling matrix size is small, and can be proceeded in very short time.

3.3.3 Special case: Non-layered streaming

In this section we propose to adapt and to simplify the solution presented in the last section to the non-layered video streaming. Initially, the priority function P_{ij} (Eq. 3-1) is simplified to the emergency priority P_E. In addition, we assume that the non-layered video is subdivided into chunks of equal size. It is hard to consider this assumption in the case of layered video, especially in the case of SVC [11] where the video stream is subdivided into NALs (Network Abstraction Layer) of different sizes. Consequently, the scheduling problem in layered video streaming can be modeled as one-to-one assignment problem, more especially as *m*-cardinality assignment problem [136], defined as the assignment of *m* jobs (chunks) among *n* to *m* agents (nodes). To do that, each neighbor node is represented in the assignment matrix of the receiver node *i* by $b_{i,j}$ rows (Figure 24), where $b_{i,j}$ represent the capacity between the receiver **peer** *i* and its neighbor *j* (in terms of chunks per time unit).



Chunks / Nodes	1	2	3	4	5	6
2	$\boxed{P_{i1}}$	P_{i2}	P_{i3}	-M	P_{i5}	P_{i6}
2	P_{i1}	$\boxed{P_{i2}}$	P_{i3}	-M	P_{i5}	P_{i6}
3	-M	-M	$\boxed{P_{i3}}$	P_{i4}	P_{i5}	P_{i6}
3	-M	-M	P_{i3}	$\boxed{P_{i4}}$	P_{i5}	P_{i6}
4	-M	-M	-M	-M	-M	$\boxed{P_{i6}}$

Figure 24: 5-cardinality assignment matrix example

In order to solve this problem we propose to transform it, first to a one-to-one classic assignment problem (i.e. transform the assignment matrix in Figure 25 to a square matrix: Figure 26), and then apply the Hungarian algorithm [140] to get the optimal scheduling. The Hungarian algorithm is a powerful combinatorial optimization algorithm, which solves a classical assignment problem in polynomial time. It is applicable, exclusively, to square assignment matrix.

Chunks / "Nodes"	1	2	…	*n*
1	P_{i1}	P_{i2}	…	P_{il}
2	P_{i1}	P_{i2}	…	P_{il}
…	…	…	…	…
m	P_{i1}	P_{i2}	…	P_{il}

Figure 25: l-cardinality assignment matrix

Chunks / "Nodes"	1	2	…	*n*
1	P_{i1}	P_{i2}	…	P_{il}
2	P_{i1}	P_{i2}	…	P_{il}
…	…	…	…	…
m	P_{i1}	P_{i2}	…	P_{il}
…	*L*	*L*	*L*	*L*
n	*L*	*L*	*L*	*L*

Figure 26: Transformed m-cardinality assignment matrix

3.3.3.1 Transformation rules

The following steps are performed to build the new square assignment matrix (Figure 26) of a receiver node *i*:

a) For each nodes j∈N_i add $b_{i,j}$ rows to the matrix, (matrix of *m* rows, where $m = \sum_{j\in N_i} b_{i,j}$)

b) For each missed chunk in node *i* add a column to the matrix (matrix of *n* columns)

c) The value of *Cell* (*k*,*j*) is the chunk *i*'s priority, i.e. *Cell* (*k*,*j*) =P_{ij}, if the node *k* holds the chunk *j*, L otherwise (-L is a big positive number).

d) If the matrix is not square, i.e. *n* > *m* : append x = *n* - *m* virtual nodes to the assignment matrix. Set the *Cell* (*k*,*j*) value to L for each row *k*∈ {*n*-*m*+1, *n*-1}, where L is a big negative number (Figure 26).

After applying these rules, we transform the formulation (Eq. 3-2) into its corresponding assignment problem represented by a square matrix (*n* X *n*) composed of *n* chunks to be assigned to *n* "nodes". Hence, the Hungarian algorithm can be applied to get the optimal chunk scheduling.

Formally, the assignment problem (Eq. 3-2) can be rewritten into the following assignment problem:

$$Max\left(\sum_{k=1}^{k=m}\sum_{j=1}^{j=n} P_{ij}^{k} R_{ij}^{k} + \sum_{k=m+1}^{k=n}\sum_{j=1}^{j=n} LR_{ij}^{k}\right) \quad (i\in N) \qquad \text{Eq. 3-5}$$

Subject to

$$\sum_{k=1}^{k=n} R_{ij}^{k} = 1 \quad (1\le j\le n), \sum_{j=1}^{j=l} R_{ij}^{k} = 1 \quad (1\le i\le n) \qquad \text{Eq. 3-6}$$

Theorem: Let $\overline{R}_{ij}^{k}$ $(1 \le\ j,k\ \le l)$ be an optimal solution to the assignment problem (Eq. 3-5), then $\overline{R}_{ij}^{k}$ is an optimal solution to the m-cardinality assignment problem (2).

Proof.

Suppose that $\overline{\mathrm{R}}_{ij}^{k}$ $(1\le j,k\le n)$ is not an optimal solution to the *m*-assignment problem (Eq. 3-2), then there exists a feasible solution $\hat{R}_{ij}^{k}$ $(1\le j\le n,\ 1\le k\le m)$ of (Eq. 3-2) which verifies:

$$Max\left(\sum_{k=1}^{k=m}\sum_{j=1}^{j=n} P_{ij}\hat{R}_{ij}^{k}\right) > Max\left(\sum_{k=1}^{k=m}\sum_{j=1}^{j=n} P_{ij}^{k}\overline{R}_{ij}^{k}\right)$$

Without loss of generality, we assume that

$$\sum_{k=n}^{k=m}\sum_{j=1}^{j=m}\hat{R}_{ij}^{k} = m$$

And

$$\hat{R}^k_{(m+1)(m+1)} = \hat{R}^k_{(m+2)(m+2)} = \ldots = \hat{R}^k_{nn} = 1$$

Since the cost $P_{ij} = L$ for $m+1 \leq k \leq n$ and $1 \leq j \leq n$, we have:

$$Max\left(\sum_{k=1}^{k=m}\sum_{j=1}^{j=n} P^i_j \hat{\mathrm{R}}^{\mathrm{k}}_{\mathrm{ij}} + (n-m)L\right) > Max\left(\sum_{k=1}^{k=m}\sum_{j=1}^{j=n} P^k_{ij} \overline{\mathrm{R}}^{\mathrm{k}}_{\mathrm{ij}} + (n-m)L\right)$$

Which is in contradiction with that $\overline{R}^k_{ij}$ $(1 \leq i,j \leq n)$ is an optimal solution to the assignment problem (Eq. 3-5).

3.4 Performance Evaluation

As abovementioned, there are three main steps for building streaming applications in overlay networks. In this paper we focus on the streaming scheduling step. For that reason we use in all our simulation a simple algorithm for overlay construction: each node randomly selects its neighbors so that a random graph is constructed. The overlay is composed of 500 nodes and each node has a certain number of neighbors. The simulations were performed using Matlab-Simulink [141].

We perform extensive simulations using an overlay in which each peer has varying number of neighbors. We compare the performance of our algorithm with three classic scheduling methods described earlier, namely random strategy (RND), Local Rarest First (LRF) and Round Robin (RR). We consider three categories of peers: 40% of users with 512Kbps, 30% with 1Mbps and 30% with 2Mbps, and for all users, the upload bandwidth capacity is half of the download bandwidth. We set the emergency priority defined in (5) as $P_E(T_i - D^i_j) = 10^{(T_i - D^i_j)}$ and we set the layer priority as $P_L(L_j) = 10^{(L-L_j)}$ in order to ensure that the lower layers have much larger priority than the upper layers. For the four methods, we adopt the conservative approach described in section 3.3. That's why we set the parameter θ to a very low value $\theta = 10^{-L}$.

3.4.1 Results and Discussion

In Figure 27, we study the performance of our mechanism (referred as AsSched) in terms of bandwidth utilization and we examine the effect of the neighbor density. It is observed that our proposed mechanism outperforms the three other scheduling schemes and ensures more than 90% of bandwidth utilization, while the LRF scheme allows the maximum bandwidth utilization of 88%. The RND method shows the worst result (up to 71%). On the other hand, we note that the bandwidth utilization for all the schemes increases with the increase in the number of neighbors (for maximum 15 neighbors). This is due to the increase of the chunks availability in the neighborhood. The stable behavior is observed when the number of neighbors for a peer is greater than 15. We conclude that, for the considered overlay, the average of 15 neighbors per peer is enough to ensure the chunks availability.

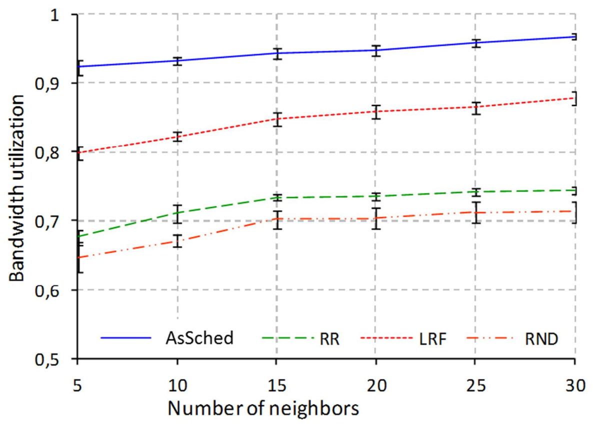


Figure 27: Bandwidth utilization Vs. Neighbor density

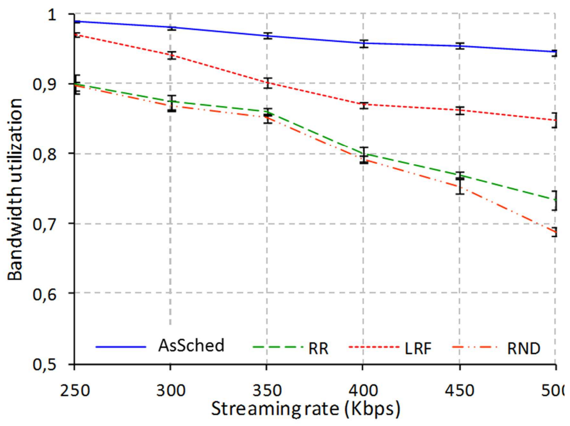


Figure 28: Bandwidth utilization Vs. Streaming rate

Moreover, we are interested on the impact of the streaming rate on the bandwidth utilization. The results are presented in Figure 28. We note that the general trend is the decreases in bandwidth utilization with the increasing streaming rate. This can be explained by the fact that when the streaming rate is high, a missed chunk has less chance to be rescheduled before its playback deadline. Nevertheless, our proposed mechanism outperform the three others mechanisms and allows a gain up to 25%. This is due to the fact that our mechanism tries to fully take advantage of the available bandwidth in the neighbor, using the assignment mechanism.

In Figure 29 and Figure 30, we evaluate the useless chunks ratio according to the neighbor's density and the streaming rate respectively. It is observed in Figure 29 that the useless chunks ratio decreases with the increasing number of neighbors. This decrease of useless chunk ratio is due to the higher probability of getting the requested chunks from the large pool of neighboring peers. Moreover, our mechanism tries to find the good tradeoff between the chunks availability in order to request the right chunk from the right neighbor. That's why the useless chunks ratio is low in proposed mechanism compared to the others systems.

In Figure 30, we evaluate the useless chunks ratio with the varying streaming rate. It is found that the useless chunks increase with the increase of the streaming rate. However, the proposed mechanism still outperforms the existing systems.

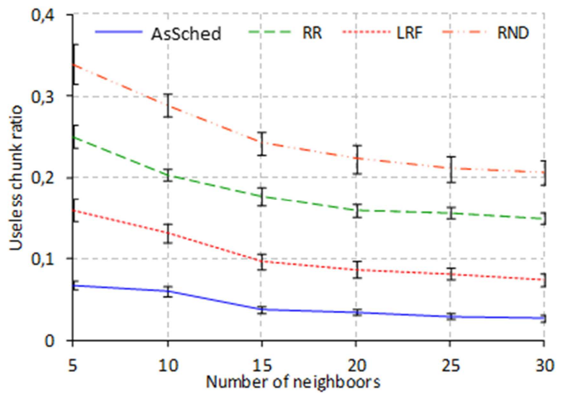


Figure 29: Useless chunks ration Vs. Neighbor density

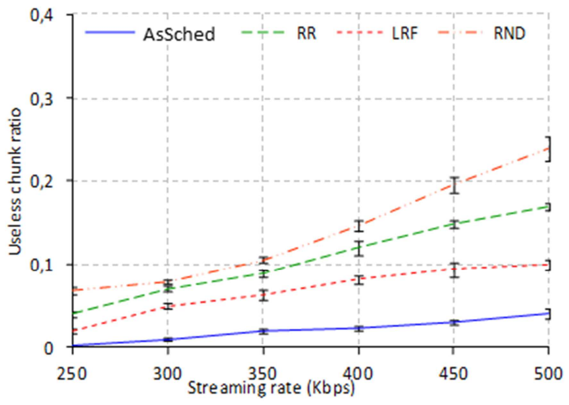


Figure 30: Useless chunks ration Vs. Streaming rate

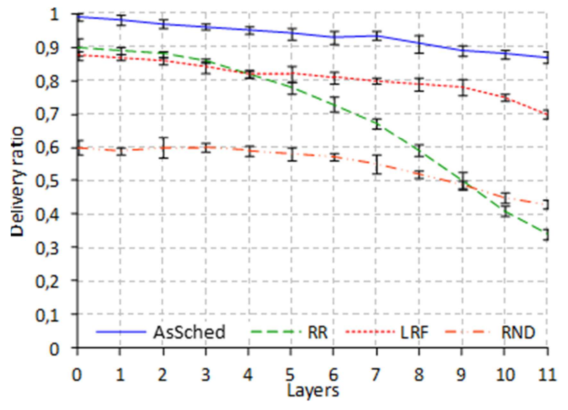


Figure 31: Delivery ratio (12 layers)

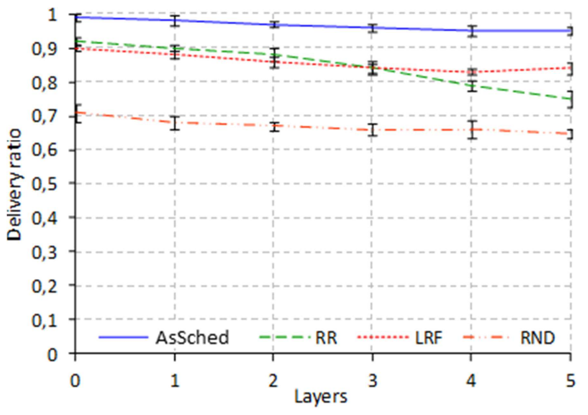


Figure 32: Delivery ratio (6 layers)

Figure 31 describes the delivery ratio at each layer. We encode the video into 12 layers and set the rate of each layer at 100 Kbps. We note that our proposed mechanism is fairly good. In lower layers, delivery ratio is approximately 1 and in higher layers it is also above 0.9. The RR has much more better delivery ratio at lower layers than higher layers but, the delivery ratio for all layers is not as good as the proposed mechanism. We note that the LRF strategy has even higher delivery ratio than the RR strategy. Finally, the random strategy has the poorest performance. As shown in Figure 31, our proposed mechanism outperforms other strategies with a gain of 10%-50% in most layers.

In order to show the importance of varying number of layers, we encode the video into 6 layers. In Figure 32, we note that the delivery ratio of each layer is nearly similar to that in 12 layers encoding scenario. Our mechanism is still the best among all the three others methods. However, we note that the delivery ratio of all the methods is little higher than in the case of 12 layers. This is due to the fact that encoding the video into six layers allow peers to allocate all their bandwidth to lower layers, however in the second case, some bandwidth will be dedicated to the higher layers (higher than 6).

3.5 Conclusion

In this chapter, we tackled the optimal scheduling problem in pull-based real-time streaming systems in multilayer streaming scenarios. We modeled the problem as a Generalized Assignment Problem (GAP) and we proposed a heuristic to resolve it. Then, in second time, we adapted the solution to non-layered streaming and we modeled it as *m*-cardinality assignment problem that we proposed a new solution for it. The simulation results show that the proposed solutions outperform the traditional strategies by about 15 to 60 percent, mainly in terms of bandwidth utilization; the key bottleneck in future networks media environments.

Chapter 4

4 Dynamic Bandwidth Allocation in P2P Layered Streaming

4.1 Introduction

In a heterogeneous P2P streaming system, there are a number of peers demanding as well as providing services. The best quality requested could be different from a peer to another. Hence the need of peers in terms of bandwidth differs also. From the system's perspective, the problem we consider is how the system should allocate resources such that the least quality perceived by the peers is maximized. The importance of this criteria lies that the least quality perceived by the peers reflects the fairness among peers.

Layered streaming, such as Scalable Video Coding (SVC) [142], provides a convenient way to perform video quality adaptation to adjust to the end user heterogeneity and changing network conditions. A layered streaming consists of a base layer and multiple enhancement layers. Receivers can adjust the video quality level to their capability by subscribing to different number of layers using pulling distribution approach. In a P2P network, it is natural to request the layer from different peers (upstream peers). Thus, each upstream peer shares its upload bandwidth among different peers (downstream peers) to serve different layers. How to resolve bandwidth conflicts among peers in order to maximize benefits of both upstream and downstream peers while respecting the layers importance, their dependencies and the peers' priorities, is highly challenging issue. Moreover, the selfishness of peers in P2P networks is inevitable [81]. As a result, the most important question to investigate, while designing resource allocation mechanisms in P2P layered streaming systems are "How to exploit and manage the selfishness of peers in order to reduce the global streaming cost while satisfying the peers quality level requirements and priorities?"

Recently there have been significant research attentions towards designing P2P overlays by exploiting the selfishness of peers. These works can be categorized into two main approaches: "non-strategic behavior approach" such as [81][143][144] and "strategic behavior approach" such as [145][146]. In the former, each peer is considered as a potential game player which envision maximizing its utility regardless the behavior of the other peers, while in the later each peer envision maximizing its utility taking into consideration the actions of the other peers. In this paper, we propose to design a simple, strategic and differentiated QoS model to efficiently manage the bandwidth allocation problem in P2P layered streaming networks based on microeconomic models, specifically the auction

mechanism. Our model make benefit of auctions mechanism dynamicity to handle the varying network conditions in terms of peers churn and consequently the available bandwidth.
The rest of this chapter is organized as follows. Section 4.2 introduces the background and related work. Section 4.3 presents the system model for layered P2P streaming and problem formulation. Section 4.4 presents our proposed bandwidth allocation and discusses the Nash equilibrium of the proposed system. Section 0 discusses our comparative studies and the results. At last, section 4.6 briefly concludes this chapter.

4.2 Related Work

The rapid growth in users' demand and network bandwidth saturation is challenging the current Internet infrastructures in performing high quality multimedia services. An efficient bandwidth allocation mechanism is required to ensure good profit of the available bandwidth in the network and ensure acceptable multimedia quality level for the end users.

The bandwidth allocation in P2P streaming applications has gained researchers attention recently. Many studies have been performed to investigate the different bandwidth allocation strategies and their impact on both network performance and end users satisfaction. Authors in [147][148][149][150][151]study the impact of peers' upload bandwidth and conclude conditions for universal streaming for churnless systems. Zhang et al. [152] studied the bandwidth influence on chunk (data block) scheduling algorithms using extensive packet-based simulations. They prove that random chunk scheduling can achieve near-optimal streaming quality if the overall upload bandwidth is at least 1.2 times of needed bandwidth. Furthermore, measurement studies and implementations [153][154] confirm that bandwidth has a big impact on streaming quality for P2P streaming systems. Authors in [154] studied the new Coolstreaming system by exposing its design options and the logic behind them. They demonstrate that there is a highly distorted resource distribution in such systems and the performance is mostly affected by the system dynamics. Recently, researchers have studied the bandwidth allocation for improving streaming quality in more challenging P2P networks such as multi-overlay, multi-sources and multi-Swarm P2P streaming systems. Wu et al. [155] studied the bandwidth contest among coexisting overlays and propose a solution based on auction. Liang and Liang et al. [156] studied the optimal bandwidth sharing in multiple video conferencing swarms systems. They dynamically share a pool of helpers between swarms to address the bandwidth shortage intra and inter-swarms. However, none of these works have taken into consideration the layered streaming properties, and its benefits for providing personalized and customized video quality while coping with bandwidth fluctuation problem.

Many recent works, such as [157][158], leverage the characteristics of SVC and P2P networks and propose adaptive video streaming mechanisms. In [157], authors proposed an optimization technique based on harmony search algorithm in order to increase the delivery ratio for the most important layers, while reducing the overhead and ensuring load balancing in the overlay. Authors in [158] proposed taxation based P2P layered streaming designs, including layer subscription policy, chunk scheduling strategy and mesh topology adaptation. Wang and all [159] proposed inter-swarm collaboration scheme in order to improve content availability and optimize resource utilization in the entire system. They show that the server cost can be significantly reduced while high streaming qualities are guaranteed in the entire system, even during extreme scenarios such as unexpected flash crowds. However, no one of these works has tackled the upstream peer bandwidth allocation problem in layered P2P systems. Indeed, a little literature has studied bandwidth conflicts for layered streaming. To our knowledge, the most closely related work is presented by Wu et al [155]. They coordinate multiple streams as an auction game, where each peer participates in media distribution by bidding for and selling bandwidth. Their strategy is not compatible with scalable streams as each stream is considered as an isolated stream without any relationship with any other streams. In contrast, there exists inherent content priority among layers. Moreover, authors only consider a scenario where the upload bandwidth sum in the network is always sufficient to support all the peers' requirement in all the overlays.

In this chapter, we look for resolving bandwidth allocation conflicts in P2P layered streaming. Our main contributions are as follows: we model the bandwidth allocation process in P2P layered streaming as a series of auction games [160] in which peers bid for and sell the upload bandwidth to maximize their benefits. In order to resolve the problem of layers dependency, we set up auction game to allocate bandwidth for each layer. We start by allocating the bandwidth for the lower layers then for the upper's ones. Then we detail our proposal with the necessary demonstrations, and we perform more extensive simulations to show the effectiveness of our model. The comparison is performed with four other strategies: Layers fair strategy, adjusted layers fair strategy, weighted strategy and downstream fair strategy described in details in section 0.

4.3 System Model and Problem Formulation

In this section, we first model the problem, we present the considered assumptions then we develop the proposed bandwidth allocation mechanism in P2P layered streaming systems.

4.3.1 Network model and assumptions

The scalable video codec, such as SVC [142], allows a video to be encoded into a base layer and several enhancement layers. The scalability of SVC is three dimensional: quality, special and temporal.

The quality scalability determines the video SNR (Signal to Noise Ratio), the spatial resolution determines the resolution of the user's screen and the temporal scalability determines the frame rate of the video. In our model we assume that the layered stream is encoded into M layers at the source peer $\{l_0, l_1 \dots l_{M-1}\}$, with l_0 representing the base layer and $l_1 \dots l_{M-1}$ representing the enhancement layers: l_1 is the first enhancement layer, l_2 is the second enhancement layer, etc. defined by an adaptation path [121] which is out of scope of this paper.

We assume that each video layer l_i is distributed with a transmission rate of b_i. Thus, each peer can subscribe to a particular video layer depending on its download capacity and other parameters such as its processing capability, preferences, etc. Without loss of generality, we assume that peers decide their quality level only depending on their available download bandwidth.

We consider an overlay network composed of *n* peers relayed by a set of application-layer links $k_{i,j}$ (link between downstream peer *i* and its upstream peer *j*). So, the topology of the overlay can be modeled as a directed weighted graph G= (*S*, *L*, *K*) where *S* denotes the set of upstream peers (called in some architecture Seeders), *L* the set of downstream peers (called in some architecture Lechers) and *K* the set of links.

To represent a practical network setting, we limit the upload and download capacity of any peer by u_i and d_i respectively. Therefore, each peer can only provide limited service for its downstream peers, and make a limited layer subscription as well. We assume also that the downstream peers have different levels of priorities $P=\{pr_1, pr_2, \dots, pr_q\}$ where $pr_1 > pr_2 > \dots > pr_q$. So, for each downstream peer $i \in L$ is assigned a level of priority pr_i. This can be mapped in real world P2P systems, for example, to an incentive mechanism where peers contributing in the overlay (sharing more upload bandwidth) are promoted to upper priority classes, and peers less contributing are demoted to lower priority classes.

For the reader's convenience, Table 2 summarizes the notations used throughout this chapter.

Table 2: Notation Table

$\boldsymbol{S}$	Set of upstream peers
$\boldsymbol{L}$	Set of downstream peers
$\boldsymbol{K}$	Set of application links
$\boldsymbol{b_i}$	Bit rate of the layer l_i
$\boldsymbol{pr_j}$	Priority of downstreal peer *j*
$\boldsymbol{u_i}$	Upload bandwidth capacity of peer *i*
$\boldsymbol{d_i}$	Download bandwidth capacity of peer *i*
$\boldsymbol{l_i}$	Video layer *i*, $i \in \{0, 1, 2\dots, M-1\}$, the stream is encoded into M layers.
$\boldsymbol{d_i^j}$	Bandwidth allocated to the downstream peer *i* by its upstream peer *j*

$b_{i,j}^k$	Bandwidth requested by the downstream peer *i* to its upstream peer *j* to acquire layer *k*
$p_{i,j}^k$	Unit price that the downstream peer *i* is willing to pay for the upstream peer *j* to acquire bandwidth for layer *k*.
l_M^i	Maximum layer available at the peer *i*
u_i^k	Peer *i'* upload bandwidth allocated to layer *k*
$\mathbf{L}_j$	Set of downstream peers connected to the upstream peer *j*
$\mathbf{S}_i$	Set of upstream peers of the peer *i*
T_i	Initial budget allocated to peer *i*
$\mathbf{Q}_j$	Quality level *j*, i.e layers: l_0, l_1, …, l_j

4.3.2 Problem statement

Through the following example we illustrate the problem of bandwidth allocation in P2P layered streaming. Figure 33 represents part of an overlay composed of three upstream peers S1, S2 and S3 with upload capacity of 100, 150 and 120 kbps respectively, and four downstream peers L1, L2, L3 and L4 with download capacity di of 150, 150, 100 and 100 kbps respectively.

Suppose each upstream peer allocates its upload bandwidth fairly between its direct downstream peers. In Figure 33 , the numbers on the arcs represent the distribution of the upstream upload bandwidth on their respective downstream peers by adopting this strategy. Peers L_1, L_2, L_3 and L_4 can get 100, 140, 90 and 40 kbps respectively. Suppose that the original stream is encoded into one base layer and one enhancement layer, with each layer rate of 50 kbps (i.e. $b_0=b_1=\ldots= b_M$ = 50kbps). Clearly, downstream peers L_1 and L_2 can access base layer and enhancement layer l_1, while downstream peer L_3 can get only the base layer and the L_4 can't play any layer. In this case the useless bandwidth *w* is:

$$w = (u_1 + u_2 + u_3) - (l_0 + l_1 + l_0 + l_1 + l_0) * b_0 = 120 \text{ kbps}$$

An optimized allocation strategy is represented in Figure 34, where downstream peers L_1, L_2 and L_3 can get layers l_0 and l_1 and peer L_4 can get the base layer l_0. The useless bandwidth in this case is 20kbps instead of 120 kbps in the first strategy.

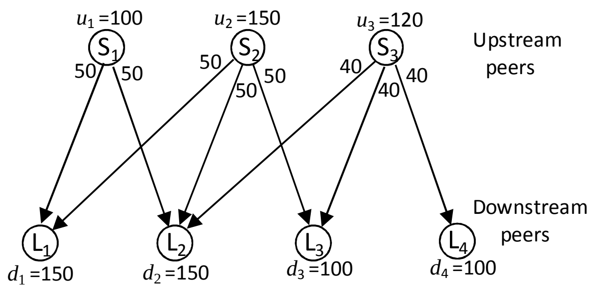


Figure 33: Example of native bandwidth distribution in P2P layered streaming

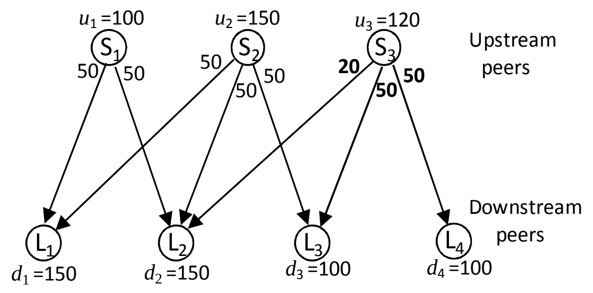


Figure 34: Example of optimized bandwidth distribution in P2P layered streaming

Following the above definitions and assumptions, we now formulate the problem as follows.

Given a set of upstream peers S willing to share a total upload bandwidth $= \sum_{i \in S} u_i$, and a set of downstream peers requesting each a bandwidth d_i corresponding to quality level Q_j (i.e. requesting layers: l_0, l_1, …, l_j), we seek the best possible plan solution which allocates the upload bandwidth *U* to better take advantage of the available bandwidth and maximize the total system utility in terms of downstream peers' quality level satisfaction. Better taking advantage of the available bandwidth implies reducing the useless data which is due either to over-allocated bandwidth to peers or receiving layers without their corresponding lower layers. The system-level utility optimization can be defined as:

$$min\left(\sum_{i \in S} u_i - \sum_{j \in L} \sum_{m=0}^{m=M^j} (l_m * b_m)\right) \qquad \text{Eq. 4-1}$$

Subject to

$$\sum_{m=0}^{m=M^j} (l_m * b_m) \leq d_j \qquad \text{Eq. 4-2}$$

Where M^j represents the maximum layer played by the peer j. Eq. 4-2 ensures that the download capacity of a peer *j* is not violated.

4.4 The Proposed Allocation Mechanism

The process of bandwidth allocation that we propose is modeled as a set of dynamic auctions organized by upstream peers in order to give rise to competition on its upload bandwidth u_j. The players in these auction games are the downstream peers. Indeed, each downstream peer, having an initial budget T_i, submits bids to all its downstream peers in order to purchase bandwidth. Consequently, each downstream peer can participate in different independent parallel auction games A_j. The upstream peers allocate their upload bandwidth depending on bids received from downstream peers.

4.4.1 Bandwidth allocation mechanism: a conservative approach

In order to guarantee a minimum quality level for all downstream peers and respect the layers dependency, every upstream should start by allocating bandwidth for the base layer, then for the enhancement layers in an ascending manner. In our model, every upstream peer organizes an auction to distribute bandwidth needed for the base layer. Then, if there is still remaining bandwidth, it organizes another auction to sell bandwidth for the first enhancement layer, the second enhancement layer, and so on. In the receivers' side, the peers participate in auctions depending on the quality level that they decide. A peer which has decided a quality level 2 will participate in a set of auctions organized by its upstream peers to distribute bandwidth for base layer (l_0), for the first enhancement layer (l_1) and for the second enhancement layer (l_2). We note that an upstream peer does not start to allocate bandwidth for an upper layer, until there is no request for the current layer from their downstream peers (i.e. until the downstream peers' requests are satisfied or their budget is exhausted).

The auction game to distribute bandwidth for certain layer li is illustrated in Figure 35. In this figure, three parallel auctions are organized by upstream peers S_1, S_2 and S_3 in order to allocate their upload bandwidths u_1, u_2 and u_3 respectively for layer l_i. The downstream peer L_1, connected to upstream peers S_1, S_2 and S_3, participates in auctions A_1, A_2 and A_3 organized by the three upstream peers. While the downstream peer L_4 participates only in auction A_3 organized by S_3 (since L_4 is connected only to S_3).

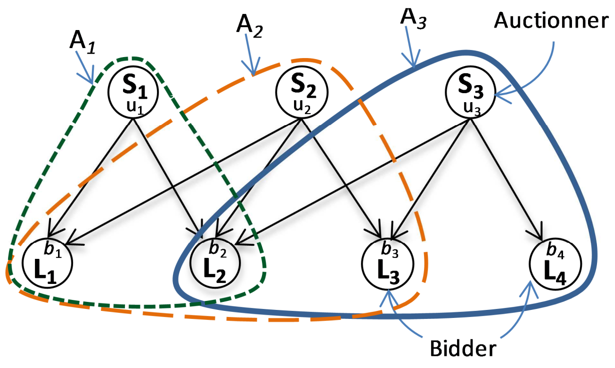


Figure 35: Auctions example

Let $b_{i,j}^{k}$ be the bandwidth requested by the downstream peer *i* to its upstream peer *j* to acquire bandwidth for layer k and $p_{i,j}^{k}$ be the unit price that the downstream peer is willing to pay for that bandwidth. The bid of the downstream peer *i* can be expressed by the pair $B_{i,j}^{k} = (b_{i,j}^{k}, p_{i,j}^{k})$

After modeling the bandwidth allocation problem in P2P layered streaming as a set of auction games, where items to sell are the upload bandwidth, and where the sellers are the upstream peers and the buyers are the downstream peers, we discuss in the following the allocation strategies of the upstream peers and the bidding strategies followed by the downstream peers.

4.4.1.1 Upstream peer's side: bandwidth allocation strategy

The upstream peer starts allocating bandwidth foremost for the lowers layers then the upper layers in ascending manner. It executes the algorithm presented in Table 3.

Table 3: Allocation strategy-global algorithm

```
$l_k = l_0$
While $u_i > 0$ and $l_k \le l_M^i$
    Auction ($l_k$)
    $u_i = u_i - u_i^k$
    $l_k = l_{k+1}$
End while
```

Where u_i^k represents the total bandwidth allocated to the layer k for all peer *i* ' downstream peers and l_M^i is the maximum layer available in the peer *i*.

In the following, we detail the strategy of the upstream peer within the auction game to allocate bandwidth for a layer *k*.

In auction A_j^k, organized by the peer *j* to allocate bandwidth for the layer *k*, the seller *j* aims to maximize its revenue by selling its bandwidth at the best price. Given the downstream peers' bids $B_{i,j}^k = (b_{i,j}^k, p_{i,j}^k)$, the upstream peer j aims to maximize:

$$max \sum_{i \in L_j} (b_{i,j}^k * p_{i,j}^k) \qquad \text{Eq. 4-3}$$

Subject to

$$\sum_{i \in L_j} b_{i,j}^k \le u_j \qquad \text{Eq. 4-4}$$

Where L_j denotes the set of downstream peers connected to the upstream peer j.

In order to maximize its revenue, the upstream peer adopts the best offer auction strategy: it starts first by serving the downstream peer, willing to pay the highest price. Once it is served and if there is still remaining bandwidth, the downstream peer proposing the second highest price will be served and so on. The allocation strategy of the bandwidth for a layer *k* is performed in many rounds as shown in Table 4.

Table 4: Per layer bandwidth allocation strategy

1) Receive bids from downstream peers
2) Allocate bandwidth to downstream peer willing to pay the highest price
3) While there is still remaining bandwidth serve the downstream peer willing to pay the next highest price
4) Notify the allocated bandwidth to all the downstream peers involved in the auction
5) Receive new bids from downstream peers (having sufficient budget) whose bandwidth request is not satisfied. Go to 2

4.4.1.2 Downstream side: bidding strategy

The question to deal with is how much bandwidth should the downstream peer request from each of its upstream peer $b_{i,j}^{k}$ and at what price $p_{i,j}^{k}$.

The ultimate goal of the downstream peer is to minimize the bidding cost as well as the streaming cost. The bidding cost can be mapped in real world's auction to the items' purchase price which express the competition degree on these items. Indeed, we believe that requesting stream from less loaded upstream peers, allows reducing the delay, since it reduces the congestion in the concerned peers. In addition it allows a good load balancing of the stream through the overlay, which can be benefic in the case of peers churn. The streaming cost can be seen as the transport cost of purchased items. In the context of P2P streaming, reducing the streaming cost is equivalent to get the stream from best links. That means links with lowest delay, lowest bit error rate, etc. So, the goal of the downstream peer is to get bandwidth from less loaded upstream peers (low price) and via best links (low streaming cost).

In our system, the downstream peer starts first by requesting bandwidth for the lower layers, the enhancement layers incrementally, by joining the corresponding auction organized by the upstream peers in this order. This strategy allows requesting primarily the bandwidth for the lower layers from the best links and then the enhancement layers from the other links. Hence, lower layers have more chance to be received by the downstream peers, and consequently more chance to be decoded properly. In the opposite, if the upper layers are promoted, the decoding of the stream could not be possible in the case of the corresponding lower layers are missing and by consequence, the throughput of the system will degrade.

Formally, in each auction games organized by an upstream peer *j* to allocate bandwidth for the layer *k*, the downstream peer aims to minimize the bidding cost. That means:

$$min \sum_{j \in s_i} \left(b_{i,j}^k * p_{i,j}^k \right)$$ Eq. 4-5

Subject to:

$$\sum_{j \in s_i} b_{i,j}^k \geq b_k$$ Eq. 4-6

Where Si denotes the set of upstream peers of the peer *i*. Condition (Eq. 4-6) ensures that the downstream peer *i* get the necessary bandwidth to play the layer *k*.

In addition to the bidding cost, the downstream peer aims also to minimize the streaming cost from each of its upstream peer *j*, denoted as $E_{i,j}(b_{i,j}^k)$. Therefore, the bidding strategy of downstream peer *i*, in each auction game A_i^k to acquire bandwidth for layer *k*, can be seen as an optimization problem of the overall cost:

$$min \sum_{j \in s_i} \left(b_{i,j}^k * p_{i,j}^k + E_{i,j}\left(b_{i,j}^k\right)\right)$$ Eq. 4-7

Subject to

$$\sum_{j \in s_i} b_{i,j}^k \geq b_k$$ Eq. 4-8

$$b_{i,j}^k \geq 0$$ Eq. 4-9

In practice, we consider the streaming cost function $E_{i,j}$ as non-decreasing function depending on $b_{i,j}^k$, strictly convex and twice derivable.

In the following we present the bidding strategy of the peer to set the requested bandwidth ($b_{i,j}^k$) and the bidding unit price ($p_{i,j}^k$).

4.4.1.2.1 Peer's strategy to set the requested bandwidth ($b_{i,j}^k$)

Given the bid price p_j^k in an auction organized by the upstream peer *j* to allocate bandwidth for layer k, the downstream peer *i* aims to optimize the overall cost by adjusting the requested bandwidth $b_{i,j}^k$ from each downstream peer. So, the goal of the downstream peer is to minimize the global marginal cost M*i* defined as the change in total cost that arises when the quantity produced changes by one unit [161].

Let c_i be the overall cost at the downstream peer i, i.e.

$$c_i = \sum_{j \in s_i} \left(b_{i,j}^k * p_{i,j}^k + E_{i,j}\left(b_{i,j}^k\right) \right) \qquad \text{Eq. 4-10}$$

The corresponding marginal cost is:

$$M_i = \frac{dc_i}{db_{i,j}^k} = \sum_{j \in s_i} p_{i,j}^k + \sum_{j \in s_i} \frac{dE_{i,j}\left(b_{i,j}^k\right)}{db_{i,j}^k} \qquad \text{Eq. 4-11}$$

Since the streaming cost function $E_{i,j}$ is strictly convex, the second derivative $\frac{dM_i}{db_{i,j}^k} = \frac{d''E_{i,j}\left(b_{i,j}^k\right)}{db_{i,j}^k}$ is strictly positive. Consequently, the marginal cost M*i* increases with the increase of the bandwidth request $b_{i,j}^k$. To solve efficiently this optimization problem we consider the water filling algorithm [162].

To set the bandwidth quantity (to request from an upstream peer *j*), the downstream peer *i* - applying the water filling algorithm - set $b_{i,j}^k$ to 0 for all $j \in \mathrm{S}_i$, then it identifies the upstream peer j_0 having the lowest marginal cost M_{i,j_0} and increases its demand b_{i,j_0}^k until the marginal cost becomes equal to the next highest marginal cost M_{i,j_1}, corresponding to the upstream peer *j1*. The downstream peer *i* increases then fairly b_{i,j_0}^k and b_{i,j_1}^k until their corresponding marginal cost M_{i,j_0} and M_{i,j_1} meet the next highest marginal cost M_{i,j_1} corresponding to the upstream peer *j2*, and so on. The downstream peer carries out this mechanism with all its upstream peers until it obtains the bandwidth that it requests for the layer *k* (b_k).

4.4.1.2.2 Peer's Downstream peer's strategy to set the bidding unit price ($P_{i,j}^k$)

After defining the bandwidth request strategy of the downstream peer, the next question to deal with is how the downstream peer set the unit price that it will announce to the upstream peer j?

In the bootstrap stage of each auction, the downstream peer is provided with an initial budget T_i^k for each layer *k*, which it spends to acquire bandwidth for the layer *k*. This budget is relative to the priority of the peer. The peer with higher priority receives larger budget, and vice versa. The budget T_i^k of the downstream peer *i* is defined by the formula:

$$T_i^k = \tilde{p}_i^k * B_k \qquad \text{Eq. 4-12}$$

Where $\tilde{\mathrm{p}}_i^k$ denotes the reference unit price assigned to the downstream peer *i*.

When the downstream peer *i* joins the auction organized by an upstream peer *j*, first, it sets its price bid to one unit (i.e. $p_{i,j}^k = 1$). Using the water filling algorithm described earlier, it computes the optimal quantity of bandwidth $b_{i,j}^k$ to request from each upstream peer *j*, and submits bids consequently. After the upstream peers allocate their upload bandwidth using the strategy described above, it proposes the bandwidth $a_{i,j}^{k}$ to the corresponding downstream peers. On receiving the proposed bandwidth, the downstream peer determines its behavior in the next round of auctions, using the following algorithm:

Table 5: Bidding algorithm

```
1) Receive bandwidth allocation and current prices from upstream peer.
2) For each upstream peer:
    If requested bandwidth $b_{i,j}^{k}$ from an upstream peer j is not satisfied:
        a. Increase the price $p_{i,j}^{k}$ by one unite within the reference price $\tilde{p}_{i}^{k}$
        b. Using the water filling algorithm, decide the quantity of bandwidth $b_{i,j}^{k}$ to request from $j$
3) Send new bids ($b_{i,j}^{k}, p_{i,j}^{k}$) to the upstream peer
```

It is clear in this algorithm that the reference price $\tilde{p}_{i}^{k}$ assigned to the downstream peer allows differentiating the downstream peers in accordance with their priorities.

4.4.2 Convergence to Nash equilibrium

Our mechanism for bandwidth allocation can be modeled as a non-cooperative game where the players are the set of downstream peers L, the strategies are the set of bids ($b_{i,j}^{k}, p_{i,j}^{k}$) and the cost function of a player *i* is the overall streaming and bidding cost C_i. Formally, we consider the finite game Γ = < L, D, C> where:

- L denotes the set of players (downstream peers)
- D denotes the set of strategies, i.e. D = (D_1, D_2 … D_i), where D_i = ($B_{i,1}^{k}$, $B_{i,2}^{k}$…$B_{i,j}^{k}$) is the tuple of bids submitted by the player to its upstream peers.
- C denotes the set of costs: C = (c_1, c_2 … c_i), where ci is the overall cost at the player *i* as defined in (Eq. 4-10).

Theorem

The auction game for bandwidth allocation in layered P2P leads to a Nash equilibrium.

Proof

We start first by proofing the Nash equilibrium in each auction game A_{i}^{k} organized by a peer *i* to allocate bandwidth for layer *k*. Then we derive the Nash equilibrium of the whole system.

The proof of the Nash equilibrium in A_i^k can be reduced to the proof of the existence of a fixed point for a certain function $\varphi: B \rightarrow B$ where B designs the bidding configuration of our game in a certain round of the auction.

According to Brouewer fixed point theorem [162], a fixed point exists for a continuous function φ of a convex compact set to itself.

From the definition of the upstream peer's bidding strategy (cf. section 4.4.1.1), we derive that the set of possible strategies B_i^k for a player i , to allocate bandwidth for layer *k*, is a finite set since the price is bounded by $\tilde{p}_i^k$ and the requested bandwidth is bounded by $b_{i,j}^k$. From this we derive that the set of strategies *B* is compact … (a)

Each binding strategy $B_{i,j}^k \in B$ is defined in [0, B_i^k] x [0, $\tilde{p}_i^k$], so the set of strategies *B* is convex … (b)

From (a) and (b) we conclude that the set of strategies *B* is non-empty convex and compact set. Let's now define the function φ and proofing its continuity.

The price adjustment algorithm described in 4.4.1.2.2 defines a mapping function $f: P^k \rightarrow P_i^k$ where P^k denotes the matrix of bid price sent by downstream peers to their upstream peers and P_i^k denotes the vector of price bids of each downstream peer *i* to its upstream peers: $P_i^k = \{P_{i,j}^k, \forall j \in \mathrm{S}_i\}$

The water filling algorithm mechanism described earlier defines a mapping function $\mathrm{g}: P_i^k \rightarrow \beta_i^k$ where β_i^k denotes the set of downstream peer's i bandwidth request to its upstream peers: $\beta_i^k = \{b_{i,j}^k,\ \forall j \in \mathrm{S}_i\}$

We define the mapping function: $\varphi: (\beta, P) \rightarrow (\beta, P)$

Where β denotes the matrix of downstream peer's bandwidth request to its upstream peers and $P = \{P_i^k, \forall i \in L_i\}$. This function maps the bidding configuration of the auction from one round to the following round.

We start by showing the continuity of the function f. Based on the water filling mechanism the optimal marginal cost is continuous on P, i.e. for each $P_i^k \in P$ the water filling algorithm associates an optimal marginal cost. In addition the inverse streaming cost function $E_{i,j}^{-1}$ is continuous on P since E is convex and twice derivable. We conclude that the function f is continuous.

Let now showing the continuity of the function g. We can see clearly in the price adjustment strategy of the downstream peer described in Table 5 that the two possible actions for the downstream peer is that for each system bidding configuration either it increases the price $P_{i,j}^{k}$ or freeze it. From that we derive the continuity of the function f. So we show the continuity of f and g, consequently the continuity of φ.

φ is a continuous function from a convex compact set B to itself, so it has a fixe point as given by the Brouewer fixed point theorem. Thus, we show the existence of the Nash equilibrium for our auction mechanism of allocation bandwidth for a layer *k*.

Since our system is composed of a set of auctions A_{i}^{k} to allocates bandwidth for a layer *k*. And we proved the Nash equilibrium of any auction, we can conclude that our system presents a Nash equilibrium point, which is the set of Nash equilibrium point of all the auctions.

4.5 Performance Evaluation

After a theoretical study of our mechanism in terms of system convergence to Nash equilibrium, we will study through simulation the performance of the proposed mechanism. The simulations were performed using Simulink-Matlab [141].

4.5.1 Simulation set up

The performance of our system is carried for mesh based P2P network of different size to measure the performance of our mechanism in terms of:

- Peers quality level satisfaction, defined as the ratio of peers getting the quality level *k* that they request.
- Useless chunks ratio, defined as the ratio of chunks received without their corresponding lower layers.
- Downstream peers' average streaming cost.
- Peers priority distribution over the network.

We generate diverse topologies of different sizes, different upstream peers' connectivity degree (defined as the number of downstream peers connected to the same upstream peer), and different stream rate. Each network includes three classes of downstream peers: Q_1, Q_2 and Q_3 with priorities pr_1, pr_2 and pr_3 respectively.

The upload bandwidth of each upstream peer varies from 256 kbps to 2 Mbps and it is uniformly distributed throughout the network. Similarly, the download bandwidth of each downstream peer is also uniformly distributed between 256 kbps and 2 Mbps. If not specified otherwise, the stream is composed of 6 layers.

We use the following function to measure the streaming cost:

$$E_{i,j}(b_{i,j}^k) = \frac{b_{i,j}^k}{x_{i,j} - b_{i,j}^k}$$

Where $x_{i,j}$ is the available bandwidth between the downstream peer i and its upstream peer j. This function expresses the ratio of the peer's i requested bandwidth from the peer j, to the remained free bandwidth in the link $m_{i,j}$, i.e. the utilization ratio of the link. Link having low utilization ratio, is considered as good link because it presents low delay and low bit error rate since the intermediate routers' queues are less occupied. We note that we choose this streaming cost function as an example to perform our simulation. Any other function to evaluate the streaming cost in a link, satisfying the conditions of convexity and derivability mentioned in 4.4.1.2, can be used.

First we compare the performance of our proposed mechanism with a set of other strategies described in the following paragraphs. Then, we will study in depth the performance of our mechanism in terms of number of auction rounds to reach the equilibrium and the appropriate network configuration.

4.5.2 Strategies of comparison

The performance of our conservative approach will be compared with the following bandwidth allocation strategies.

1. Layers fair strategy

   The most intuitive strategy is to allocate the upload bandwidth fairly between the available layers, i.e. for each available layer l_i: $l_0 = l_1 = \ldots = l_{n-1} = u_i/n$. The idea behind this strategy is to not promote a layer to the detriment of another layer.

2. Adjusted layers fair strategy

   This strategy is similar to the layer fair strategy. However the upload bandwidth of an upstream peer is fairly split between the requested layers. There is no bandwidth allocated to unsolicited layers from the downstream. The advantage of this strategy comparing to the previous is to minimize the useless bandwidth allocated to unsolicited layers.

3. Weighted strategy

   In Layered streaming layers have different importance and different distribution in the network. Base layer (Layer 0) for example is widely requested since it is mandatory to decode a stream of any quality level. This is why in this strategy we give more importance for this layer, and we propose the following distribution of the upload bandwidth over the layers:

l_0	l_1	l_2	l_3	l_4	l_5
30%	20%	15%	15%	10%	10%

4. Downstream fair strategy

   In this strategy the upload bandwidth is allocated fairly between the downstream peers. That means each upstream peer splits its bandwidth fairly between all the downstream peers connected to it, i.e.

$$d_j = \frac{u_i}{S_i}$$

Where d_j the bandwidth is allocated to the downstream peer *j* and S_i is the set of the peer *i'* downstream peers.

4.5.3 Results and discussions

4.5.3.1 Bandwidth utilization Vs neighbor density

Firstly, we study the impact of the neighbor density on the upload bandwidth utilization. We mean by the upload bandwidth utilization, the ratio of the requested bandwidth among the overall available upload bandwidth. Indeed, the bandwidth allocation mechanisms allocate the upstream peers' bandwidth, however it's up to the downstream peers to request the appropriate chunks. In all compared mechanisms, the same scheduling algorithm that we proposed in [163] is used. Figure 36 represents the variation of bandwidth utilization while increasing the neighbor density from 5 to 30 neighbors per peer in an overlay composed of 1000 peers, under a streaming rate of 1.5 Mbps. The general trend is the increase of the upload bandwidth utilization with the increase of the neighbor density. This is because low neighbor density leads to low probability to get the appropriate chunks, mostly for the upper layers that are infrequent in the network due to the layered stream nature. Conversely, increasing the neighbor density leads to increase the chance to get the appropriate chunks, and consequently the increase of the bandwidth utilization.

We note that the layered fair approach presents the worst performance among all the studied strategies, because it allocates the bandwidth fairly between all layers regardless the downstream peers' needs. Consequently, a large part of the allocated bandwidth is not requested. As expected, the adjusted faire layers presents better performance, because the upload bandwidth is shared between the potential requested layers, but the distribution of the bandwidth between layers is not proportional to the requests on these layers. Our proposed mechanism (referred to as conservative approach) presents the best bandwidth utilization, since the bandwidth allocation is performed while taking into consideration the downstream peers needs in terms of layers. The upload bandwidth is not beforehand dedicated to a layer, but it is dynamically allocated, starting by the most important layers, then the less important ones. The weighted approach presents also a bad performance. Although, more importance is given to the lower layers, but the proposed configuration is not appropriate for all upstream peers to meet the demand of the downstream peers. This is why, in our mechanism the bandwidth is allocated dynamically, depending on the peers' request on layers.

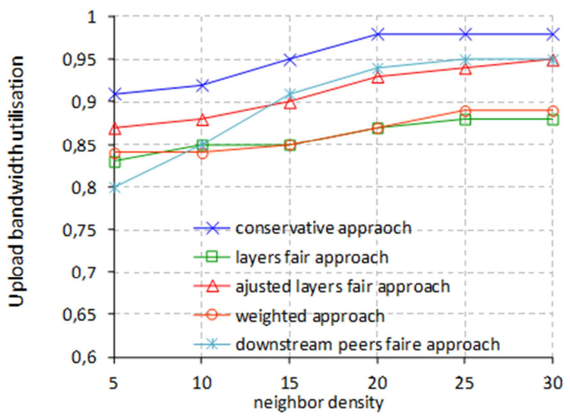


Figure 36: Bandwidth utilization Vs Neighbor density

4.5.3.2 Quality level satisfaction

Although the first studied parameter "bandwidth utilization", as defined, represents the proportion of the requested bandwidth among the all the allocated bandwidth, gives a general idea about the effectiveness of the allocation strategies. But it does not reflect the good utilization of the requested bandwidth. For example a peer receiving upper layers without their corresponding layers does not really take advantage of the bandwidth allocated to him. This is why we are interested, in second time, on new metric namely the "peers' quality level satisfaction". For each requested quality level we measure the portion of peers effectively receiving this quality level among all peers that request it.

4.5.3.2.1 Quality level satisfaction Vs Streaming rate

Firstly we measure the quality level satisfaction while varying the neighbor density in the network from 5 to 30 neighbors per peer.

In Figure 37 we measure the quality level satisfaction while varying the neighbor density. The first remark is that peers requesting low level quality are always more satisfied. In all neighbor density configurations, peers requesting quality level Q_0 and Q_1 are satisfied. This is was expected since in our mechanism we start first by allocating bandwidth, in each upstream peer, to the lower layers then if there is remaining bandwidth, we allocate for the upper layers. The second remark is that, increasing the neighbor density leads to increase the quality level satisfaction, because increasing the number of neighbors leads to increase the probability to get the requested layers, mostly for the upper layers. We note that a neighbor density of 20 neighbors provide a good quality level satisfaction. Indeed, more than 97% of peers level quality is satisfied, all categories included.

We note in Figure 38 that the quality level satisfaction in layers fair strategy is lower than in the proposed mechanism. We note that in the overlay of neighbors' density equal to 5, only 55% of peers requesting the quality level L_5 are satisfied. This is due, firstly to the lake of lower layers in the neighborhood, mainly the base layer which is required by all peers' categories (l_0, l_1… l_5), while the layers fair strategy split fairly the bandwidth between all layers. We note also that quality level satisfaction increase also by increasing the neighbor density, in the layers fair strategy.

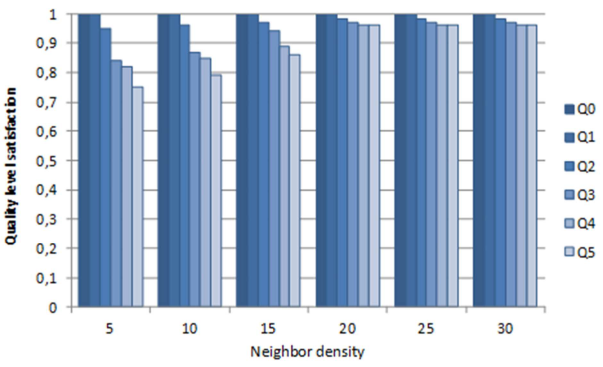


Figure 37: Quality level satisfaction Vs neighbor density (Conservative approach)

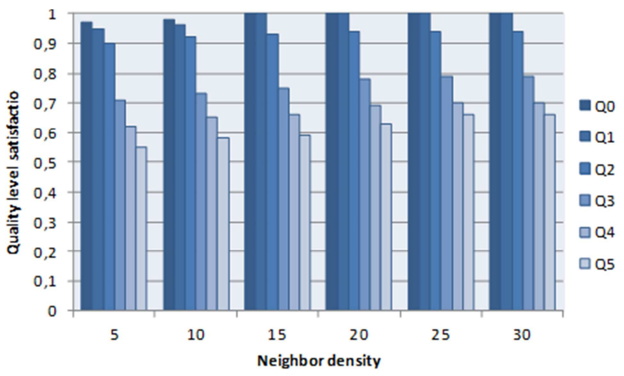


Figure 38: Quality level satisfaction Vs neighbor density (layers fair strategy)

In accordance with the previous study on the bandwidth utilization, the adjusted fair strategy (Figure 39), presents better performance than the fair strategy, for the same reason, because in this strategy we allocate bandwidth fairly between requested layers, never for the non-solicited layers.

In the weighted strategy (Figure 40), we note that the peers' quality level satisfaction is higher than in the adjusted fair strategy, for all peers' category. This confirms the first statement in the study of the bandwidth utilization parameter, the bandwidth allocation mechanism should give more importance for the lowers layers in the bandwidth allocation process.

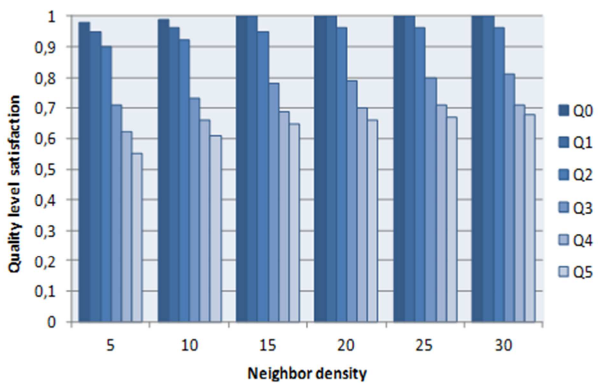


Figure 39: Quality level satisfaction Vs neighbor density (Adjusted layers fair strategy)

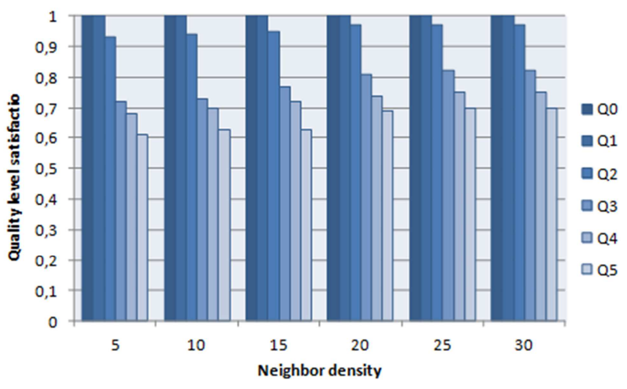


Figure 40: Quality level satisfaction Vs neighbor density (Weighted strategy)

We note also a good performance of the downstream peers' fair strategy (Figure 41). We believe that these results are due to the good scheduling algorithm associated with this bandwidth allocation strategy. Because in this strategy the bandwidth is subdivided fairly between the downstream peers, which is not always relevant due to the peers heterogeneity. In addition the problem in this case is pushed on the scheduling algorithm. So this strategy is not aware of the layered nature of the stream.

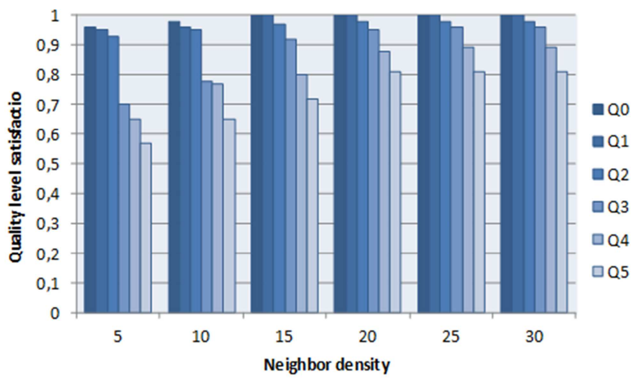


Figure 41: Quality level satisfaction Vs neighbor density (Downstream peers fair strategy)

4.5.3.2.2 Quality level satisfaction Vs neighbor density

We performed another set of tests to measure the quality level satisfaction under different streaming rate conditions varying from 300kbps to 1800kbps. For different value of streaming rate, we measure the quality level satisfaction of each peer's category.

We remember that the peers download bandwidth is uniformly distributed between 256 kbps and 2 Mbps. This is why all peers can request at least the quality 4 for the lowest value of the streaming rate, i.e. 300kbps (all layer have the same size of 50kbps). We note that for a streaming rate of 300kbps, all peers requesting quality level 4 and 5 are fully satisfied. This is due to peers' overcapacity comparing to the streaming rate. However, when the streaming rate increases, we witness a drop in the quality level satisfaction. This is valid for all peers' categories and for all the studied strategies. Increase of the streaming rate leads to reduce the ratio of the high quality level peers in the network and consequently the high layers become increasingly scarce. Consequently the probability to get these layers decrease, thus reduce the quality level satisfaction for the higher quality level peers.

Comparing the different bandwidth allocation strategies we note that the proposed strategy presents the best quality level satisfaction among the other strategies, since it allocates the bandwidth not only following the peers' needs in terms of bandwidth but also respecting the layers dependency. The most important layers first then the less important. In addition it allocates the best links to the most important layers. In this way the useless chunks are reduced and by the way increase the peers' quality level satisfaction.

At streaming rate of 600kbps, the layer size is 100kbps, and the peers quality level requested varies from 1 to 5. We note that the peers requesting the low quality level are satisfied (100% for Q_1 and Q_2), then the degree of satisfaction decrease with the increase of the streaming rate. This is due to the rareness of the higher layers in the network. The quality level satisfaction of the higher capacity peers decrease with the increase of the streaming rate. Only 59% of L_5 peers are satisfied when the streaming rate is 1800 kbps.

Among the studied strategies, we note that the layer fair strategy (Figure 43) presents the worst performance. Since the bandwidth is fairly is split between the received layers regardless the downstream peers needs and regardless the layers importance. At streaming rate of 900kbps, 70% of L_5 are satisfied, and only 50% of them are satisfied when the streaming rate is 1800kbps.

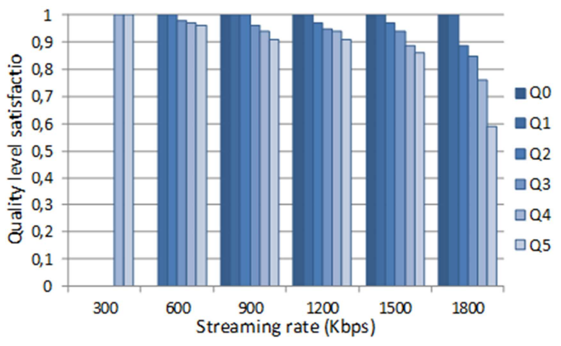


Figure 42: Quality level satisfaction Vs streaming rate (Conservative approach)

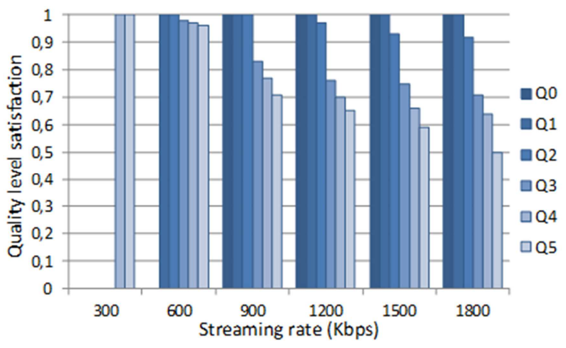


Figure 43: Quality level satisfaction Vs streaming rate (Layers fair strategy)

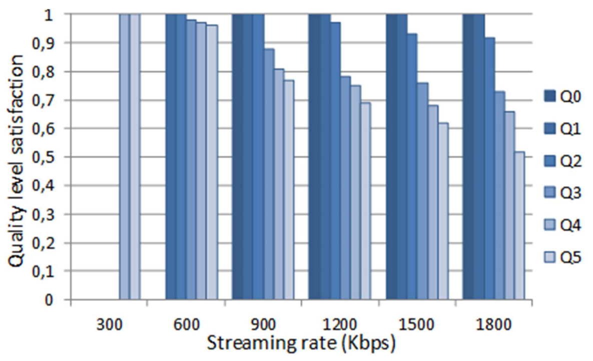


Figure 44: Quality level satisfaction Vs streaming rate (Adjusted layer fair strategy)

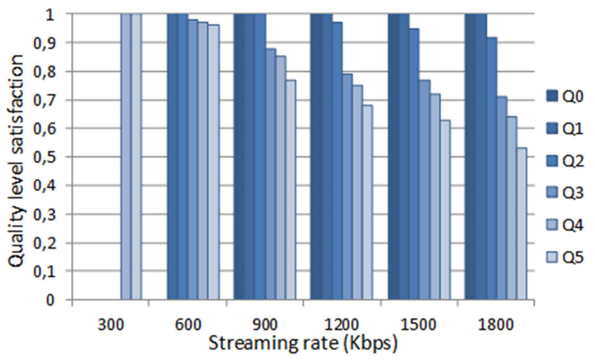


Figure 45: Quality level satisfaction Vs streaming rate (Weighted strategy)

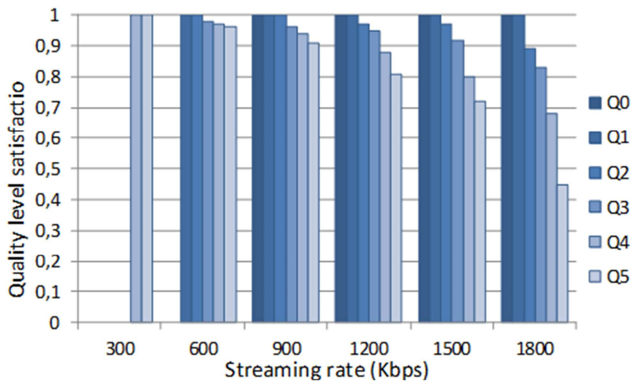


Figure 46: Quality level satisfaction Vs streaming rate (Downstream peers fair strategy)

We note also that the Weighted strategy presents the worst quality level satisfaction. We believe that this is due to fixed bandwidth portion allocated to layers. Indeed, even we allocate more bandwidth for base layer (30%) comparing to the layers fair strategy, in some cases it is extraneous with the need of upstream peers, at the expense of the upper layers. This confirms the effectiveness of our approach based on dynamic allocation of the bandwidth without any waste, starting by the lower layers then the uppers ones.

4.5.3.2.3 Quality level satisfaction Vs peers capacity (3 layers)

After a detailed study of the neighbor density and the streaming rate impact on the peers' quality level satisfaction, we study now the impact of the network configuration on this parameter. For that reason we vary the portion of the high capacity peers, having the capability to get l_2, from 5 to 50%. Through this study we look for investigating the impact of the high capacity users in the network and comparing the performance of the five considered strategies and identifying the best network configuration in terms of peers' capacity in order to ensure a good quality of service for the end users.

In order to simplify this study, the video is encoded into 3 layers: a base layer l_0 and two enhancement layers l_1 and l_2; each of them streamed at 300kbps (global streaming rate is 900kbps). The network is composed of 1000 peers; each of them is connected to 10 neighbors.

In Figure 47, we note that the peers requesting quality level 0 and 1 are satisfied regardless the network configuration. However the satisfaction of peers requesting the quality level 2 is correlated with the portion these later in the network. At 5% of high capacity peers in the network, only 25% of these peers are satisfied. This percentage is acceptable (up then 90%) only when the portion of these peers is up or equal to 20%. This can be naturally explained by the presence of the higher layers in the neighborhood. A low portion of high capacity peers causes a scarce presence of the upper layers in the neighborhood, and consequently the propagation of these later until the high capacity peers.

When observing Figure 47, Figure 48, Figure 49, Figure 50 and Figure 51 we note that in the studied P2P network 25% of high capacity peers in the network is sufficient to ensure a good propagation of the higher layers. Indeed, at this ratio, up to 95% of high capacity peers are satisfied in the conservative approach that we propose. Up to 90% in the case of Layers fair strategy and Downstream peers' fair strategy, and about 60% for the adjusted layers fair strategy and the weighted strategy.

In Figure 48, we note that initially the quality level satisfaction of Q_0 and Q_1 peers decreases with the increase of the l_2 peers' portion when the ratio of these later is less than 20%. We explain this by the fact that increase of the l_2 peers leads to increase of the bandwidth portion reserved to the layer 2, which is considered as useless bandwidth since few peers in the neighborhood request it. More l_2 peers in the network leads to reduce the portion of l_0 and l_1 peers in the network and consequently their satisfaction which returns to its normal proportion. At the opposite, in Figure 50 we note that increasing the number of l_2 peers leads to increase the satisfaction degree of these peers until a certain threshold (20%) where the fixed ratio for layer 2 bandwidth became insufficient to meet the needs of all l_2 peers, whence degradation of the quality level satisfaction of l_2 peers.

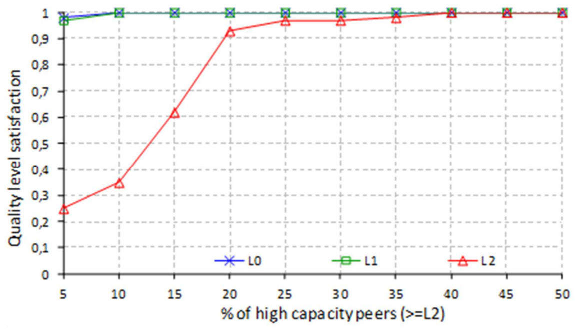


Figure 47: Quality level satisfaction Vs peers capacity (Conservative approach)

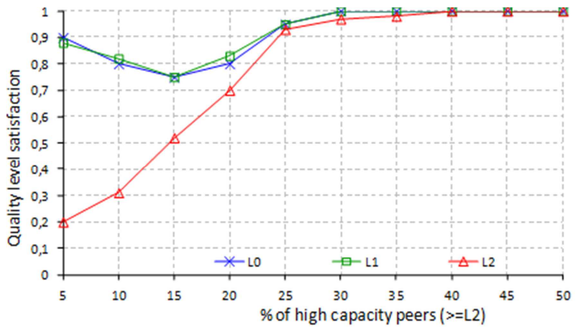


Figure 48: Quality level satisfaction Vs peers capacity (Layers fair strategy)

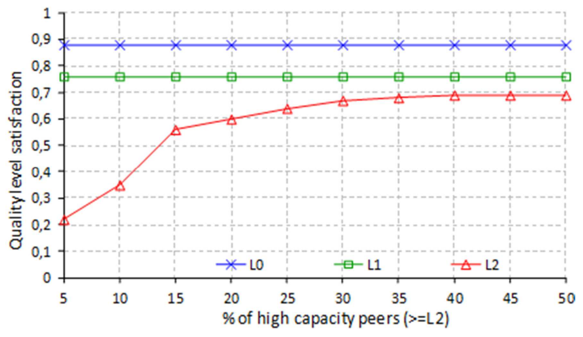


Figure 49: Quality level satisfaction Vs peers capacity (Adjusted layer fair strategy)

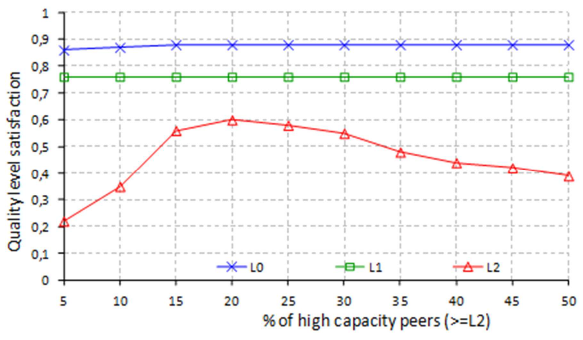


Figure 50: Quality level satisfaction Vs peers capacity (Weighted strategy)

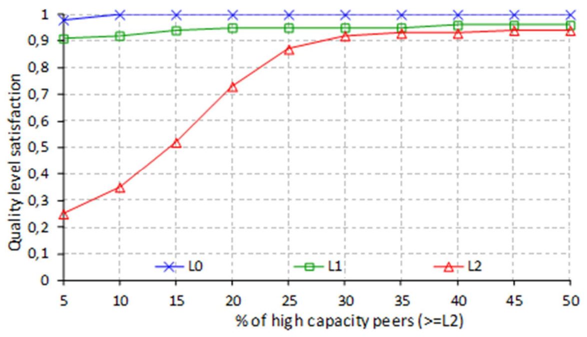


Figure 51: Quality level satisfaction Vs peers capacity (Downstream peers fair strategy)

4.5.3.3 Conservative approach performance study

Figure 52 illustrates the evolution of the average streaming cost experienced by peers, by varying the size of each class of peers. The network is composed of 500 peers.

First of all, we observe that the streaming cost in downstream peers of Q_1 is smaller than in downstream peers of Q_2 which is smaller than in downstream peers of Q_3. This shows that the bandwidth allocation mechanism in our system respects the priorities of peers.

We observe also that with the increase in downstream peers of Q_1, the average streaming cost experienced by peers of this class increases. This is due to the increase in the competition on the good quality links (links with low streaming cost). As a result, more and more downstream peers of Q_1 get connected with links having higher streaming cost. On the opposite, the streaming cost of downstream peers with Q_3 decreases with the increase of their number in the network. This can be explained by the decrease of the number of Q_1 and Q_2 in the network, consequently the competition on the best links reduces.

Figure 53 illustrates the evolution of the number of rounds necessary to reach the equilibrium of the system while varying the size of the class Q_1. We observe that the number of rounds to reach the equilibrium increases with the increase of size of the network, which can be explained by the increase of the competition on the bandwidth. Moreover, when increasing the number of downstream peers of Q_1 from 10% to 50%, the number of rounds necessary to allocate the bandwidth increases by at least 10 rounds.

From Figure 52 and Figure 53, we conclude that increasing the number of downstream peers of Q_1 leads to worst streaming const, i.e. deterioration in the QoS' level experienced by the downstream peers of this class. In addition, more rounds are necessary to reach the equilibrium of the system. We note also that a network composed of 10% to 20% of Q_1, ensure a good quality level to downstream peers of this quality as well as less rounds to reach the equilibrium.

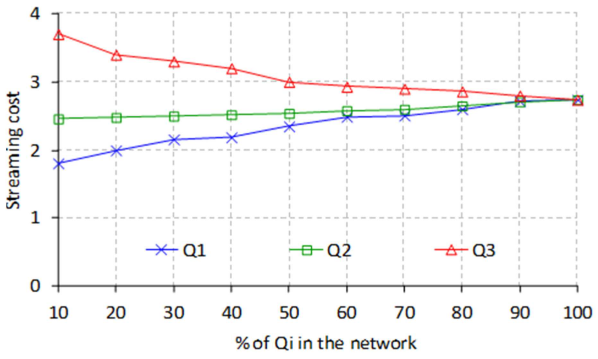


Figure 52: streaming cost Vs network configuration

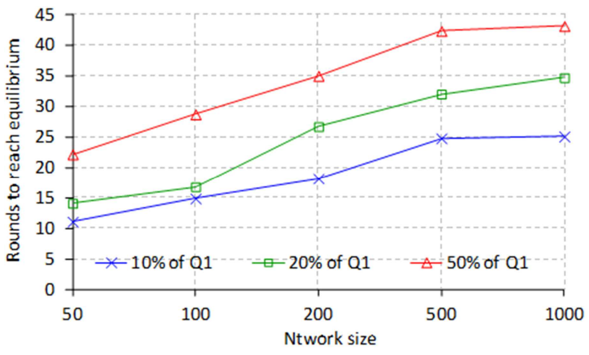


Figure 53: Number of rounds to reach equilibrium Vs Network size/configuration

4.6 Conclusion

In this chapter, we proposed a bandwidth allocation mechanism for layered streaming in P2P network that allocates appropriate bandwidth to the appropriate peers while ensuring a minimum quality level to all peers. Each upstream peer organizes a set of auctions to sell its bandwidth, an auction for each layer, starting by the lower layers. In this manner the lower layers are transmitted via the best links, consequently increasing the system throughput.

To study the effectiveness of the proposed mechanism, we performed extensive simulations and we compare the performance our mechanism with four others strategies. We studied different metrics that are essential in determining the performance of our proposed mechanism. We studied first the bandwidth utilization ratio. The results shows that the proposed mechanism outperform the others mechanism and ensure up to 90% of bandwidth use. Then we studied in second time the quality level satisfaction of peers which is the ultimate goal of such mechanism in the context of P2P streaming. The obtained results confirm those found for the bandwidth utilization.

We conclude that the bandwidth allocation mechanism in the P2P streaming systems should be dynamic and aware of the neighbors' needs, not only in terms download bandwidth but also in terms of layers, and should be aware also of the layers importance, where most important layers should be conveyed by the most reliable links.

Chapter 5

5 Smoothing of Layered Streaming in P2P Networks

5.1 Introduction

Layered video streaming in peer-to-peer (P2P) networks, especially Scalable Video Coding (SVC) (cf. 2.2.3.1.1), has drawn great interest, since it can not only accommodate large numbers of users, but also handle peer heterogeneity in terms of download bandwidth, terminal capabilities and user preferences. With layered video coding, the original video is partitioned into multiple layers, which are then transmitted independently. This allows peers with high capacity to receive all layers of the video and enjoy maximum quality, while peers with lower capacity receive a subset of layers and experience reduced quality.

Three essential components need to be considered in layered video streaming over P2P networks: overlay construction, content retrieval and content adaptation. The overlay construction component refers to the selection of appropriate neighbors for the retrieval of content as discussed in section 2.2.1. The content adaptation and content retrieval are responsible for the selection of appropriate layers, and requesting those layers from different overlay neighbors. The design of algorithms that optimally perform these tasks is non-trivial, especially when uniform high quality playback is desired. Of the many engineering challenges posed by such design, one of the most important is the fluctuation in available bandwidth between peers. On the one hand, the delivery of maximal video quality to the user provides a rationale for the algorithm to aggressively select higher quality layers when sufficient bandwidth becomes available. On the other hand, if this bandwidth is only available for a brief period, the algorithm will soon be forced to fall back to selecting lower quality layers, leading to an undesirable fluctuation in user QoE (Quality of Experience). Therefore, a playout smoothing mechanism must balance the aggressiveness with which it uses bandwidth when it becomes available, and the conservativeness with which it maintains a stable user QoE.

In non-layered streaming, there is almost no difference between high delivery ratio and high throughput, because there is no inter-layer dependency. This is not the case in layered streaming, because enhancement layers are only useful if lower layers are present. Losses at the lower layers can severely degrade system performance and lead to wasted system resources if enhancement layers are selected for which the corresponding lower layers are missing. We will refer as useless chunks to these upper layer chunks that cannot be correctly decoded because some lower layer chunks are missing or the chunks received after their playback deadline.

In this work, we present a quality-aware smoothing mechanism which aims to design a quality adaptation mechanism that control the level of smoothness of the stream by switching gracefully from one quality level to another, while trying to maximizing the video quality level.

The rest of this chapter is organized as follows. First, we present some of the related work on layered streaming in P2P networks (section 5.2). We then discuss different metrics to be considered for smoothing of layered streaming (5.3). We present the mechanism for smooth selection of appropriate layers considering the available bandwidth fluctuations in the network (section 5.4). In the end, we studied the performance evaluation of our proposed smoothing mechanism (section 5.5).

5.2 Related Work

There have been numerous efforts in the design and evaluation of layered video streaming systems in the last decade. In addition to its promise in handling client heterogeneity, layered video has recently received attention as an application layer solution to the limited deployment of IP-multicast. We now present some of these works, and contrast them with the one presented in this chapter.

To handle changing networks dynamics, several layered streaming systems have been proposed [164][165][166][167][168]. Saparilla and Ross [164] were among the earliest contributions who investigated two-layer coded video streaming over the Internet. This study focuses on the dynamic allocation of available bandwidth to these two layers in order to reduce client starvation. However, the selection of appropriate layers under the varying network conditions was not discussed.

Recently, the authors in [169] focused on combining the benefits of network coding with layered streaming to mitigate the inherent challenges in unstructured P2P systems. The work focuses on the average quality satisfaction of the peer, but does not consider the degradation in user's quality of experience (QoE) due to variation in quality levels. In [170] authors proposed a taxation-based P2P layered streaming design including layer subscription strategy and mesh topology adaptation. The taxation mechanism is devised to strike the right balance between social welfare and that of individual peers. Although the work considers the strategy for layer selection, it mainly focuses on fairness in P2P systems.

In [171], Nguyen et al. demonstrate the importance of neighbor selection in layered streaming and identify the unique challenges of neighbor selection for system performance. In addition, the authors propose a new neighbor selection technique that can offer good performance and scalability under network fluctuations. The core of the technique is a preemption rule that biases neighbor selection policies by taking into account peer capacity. The work focuses on achieving high quality by providing each peer with a set of neighbors having higher perceived quality. Our work moves a step further by not only selecting the appropriate layers according to available local resources to ensure smoothing, but also by ensuring the effective utilization of the available overlay capacity.

The next section will explain the problem of layered smoothing in detail along with the important metrics which should be considered while designing a smooth layered streaming system for P2P networks.

5.3 Problem Statement

One of the problems in assessing the performance of a video delivery scheme is the lack of a good metric that captures the user's perception of video quality. It is generally observed that it is visually more pleasing to watch a video with consistent, lower quality than one with high but varying quality [172]. However, reducing the quality to a bare minimum by following a strictly conservative approach is undesirable, as it fails to adequately take advantage of available overlay resources. The objective of the layer selection mechanism presented in next sections is to optimize the perceived video quality, while at the same time ensuring the smooth delivery of the layered stream. To explain our smoothness criterion, we direct the reader to Figure 54 which exemplifies two possible approaches to stream smoothing for a given available bandwidth profile.

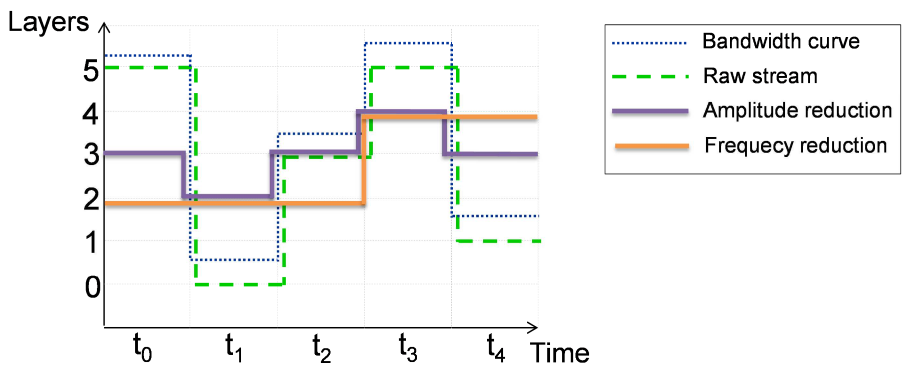


Figure 54: smoothing by amplitude/frequency reduction

In Figure 54, raw stream attempts to precisely track the changing available bandwidth. As a result, the QoE of the user may be severely degraded, especially when there is a drop from a high quality level to a much lower one (as in the case of time slot t_0 and t_1). A first smoothing technique is to reduce the size of the jump from higher to lower quality level (violet curve).

The objective here is to ensure a gradual change in quality levels, rather than subjecting the user to widely varying QoE. This technique is referred to as *amplitude reduction*. An alternative technique is where smoothing focuses on reducing the number of quality changes (orange curve) from 4 in the amplitude reduction case to 1. This technique, referred to as *frequency reduction*, aims to reduce the number of changes in quality level due to variations in available bandwidth.

The playout smoothing mechanism should take into consideration two additional factors. Firstly, the smoothing mechanism should neither be too conservative (sacrificing higher quality to achieve long term smoothness) nor too aggressive (sacrificing better smoothness to better take advantage of short-term available bandwidth). Secondly, the smoothing mechanism should also take into the consideration the extra delay for the user that may experience as a side-effect of the smoothing algorithm. This extra-delay may adversely affect the liveness of the stream, thus making it unsuitable for live streaming applications. Thus, a playback smoothing mechanism should apply both amplitude and frequency reduction to achieve a good tradeoff between user QoE and bandwidth efficiency while incurring low processing delay.

5.4 The Smoothing Mechanism

In this section, we present the core concepts behind our proposed mechanism. We assume a chunk-based, pull approach in which chunks are the basic unit of data exchange in the network; each chunk carries information for a given video segment at a given layer. The smoothing function defines the layers to be requested, taking into account the estimated available bandwidth at the receiver peer. In practice, this function operates over two distinct time horizons. The first one, which we will call the *initial quality smoothing function*, is invoked only once at the beginning of the session, and is responsible for the definition of the complete set of layers that the algorithm will consider at execution time. The second time horizon, which we will call the *runtime quality smoothing function*, is used during the execution of the algorithm to select, according to measured bandwidth variations, between the set of layers defined at startup. The detailed smoothing function is presented in next sub-section.

The core objective of the smoothing function is to select which layers will be requested during next time period in order to simultaneously reduce the quality variability for the layered stream (to increase overall QoE) while increasing the overall stream throughput (both to increase overall quality and better make use of available network bandwidth). We will achieve this by first deriving a tradeoff quality level that reduces variability while increasing overall quality, subject to the constraint that the bandwidth required for achieving this quality level should not exceed the available bandwidth at the receiver peer.

We consider a discrete-time model, where each time slot represents an arbitrary number of video chunks. Let L^t represent the selected tradeoff quality level at time *t*, and S_w the smoothing window size.

In order to provide the smoothing algorithm with relevant information regarding video chunks, we divide the receiver side exchanging window into the following three different intervals (see Figure 55).

Playing Buffer: It contains a number of chunks ready for playing.

Urgent Buffer: It contains a number of chunks that should be requested urgently from overlay neighbor peers. This buffer contains the smoothing window of length S_w, used by our proposed algorithm to define the tradeoff quality level L^t to be used as input for the scheduling function. Only those chunks belonging to layers necessary to achieve L^t are requested (i.e. $L^k \ \forall k \leq t$). The length of smoothing window introduces an extra lag time prior to decoding that should be taken into consideration for live streaming applications.

Prefetching Buffer: It contains a number of possibly useful chunks which can be prefetched in future for decoding the content.

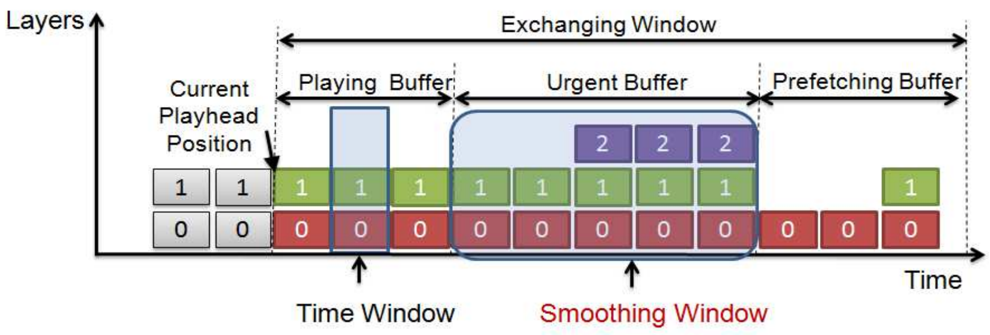


Figure 55: High level Overview of Sliding Window at Receiver Peer

As stated previously, the smoothing function operates over two disjoint time horizons: Initial startup (the initial quality smoothing function) and real-time adaptation (the *runtime* quality smoothing function). The *startup* function is invoked only once, and it is responsible for selection of appropriate layers in the beginning of the session. It is designed in such a way that each peer can determine the highest quality level it can achieve before starting to play the layered video stream. The purpose of this function is to initialize the quality smoothing function with important information regarding the user context. The important considerations for this determination are user preferences, terminal capabilities, link capacity and video decoder processing power. Initially the peer request the maximum quality, referred as *aggressive start*. Using different types of user metadata, we can filter out those layers that are not compatible with user request. The provisioning of metadata is out of the scope of this chapter and not considered in this work.

We now present our algorithms for both amplitude and frequency reduction, and then a hybrid approach that provides the benefits of both.

5.4.1 Amplitude reduction

The objective of amplitude reduction is, essentially, the reduction in the size of jumps between quality levels. If we assume that quality level jumps are a random variable, this is essentially equivalent to reducing their dispersion around the mean. Therefore, when measuring jump size, we consider the mean quality level $\bar{L}$ achieved during the previous smoothing window. In particular, $\bar{L}$ is defined as:

$$\bar{L} = f_l(\frac{1}{S_W} \sum_{j=t_c-s_w}^{j=t_c} B^j) \qquad \text{Eq. 5-1}$$

Where t_c is the current playhead position in the video, S_W is the size of smoothing window, while B^j is the available download bandwidth at the receiver peer at time periode *j*. The function f_l provides the layer corresponding to the average bandwidth.

We use the average quality $\bar{L}$ in our algorithm for selecting the quality level for next time period, as shown in Table **6**. Our algorithm relies on the mean deviation α^t to decide the quality level L^t at time t. We define α^t as:

$$\alpha^t = |L^{t-1} - \bar{L}| \qquad \text{Eq. 5-2}$$

Where L^{t-1} is the quality level at the previous time $t-1$. Let B^t denote available download bandwidth of the receiver peer at time *t*, and $\widehat{B^t}$ the estimated bandwidth at time *t*. In our case, the estimated bandwidth is the average bandwidth for last smoothing window. Further, let $\widehat{L^t}$ be the maximum quality that could be sustained with a bandwidth of $\widehat{B^t}$ and L_r^t be the already available chunks at time *t* due to prefetching. Then, the following algorithm can be used to calculate the tradeoff quality L^t at the current time t.

Table 6: Amplitude reduction mechanism

```
if $(\widehat{B^t} \geq B^{t-1})$
        $L^t = \min(\widehat{L^t}, L^{t-1} + \alpha^t)$
else
        $L^t = \min(\widehat{L^t} + L_r^t,\ L^{t-1})$
end if
```

The algorithm for amplitude reduction takes two distinct cases into account. Whenever there is an expected increase in available bandwidth, the algorithm provides an increase in the quality level without violating the available bandwidth constraint at the receiving peer. The average value minimizes the deviation from the mean and as a result there is gradual increase in the quality level. The remaining available bandwidth is used to acquire the chunks in the prefetching buffer. In case of an expected decrease in available bandwidth, the algorithm reduces the amplitude by utilizing the already available chunks. Thus, the algorithm for amplitude reduction focuses on stepwise decrease in order to reduce the jump when bandwidth decreases

5.4.2 Frequency reduction

The objective of a frequency reduction mechanism is to reduce the number of quality level changes in the layered stream that can occur as a result of varying available bandwidth at the receiver peer. In this case, our objective will be to maintain the same quality level within a particular smoothing window. In the following sections we present two strategies to perform the frequency reduction. The first strategy relies on the chunks available in the prefetching buffer to decide the quality level for the next smoothing window (*prefetching-based frequency reduction*), while the second strategy takes the mean quality level observed in a smoothing window as the target quality level for the next smoothing window (*mean-based frequency reduction*).

5.4.2.1 Prefetching-based frequency reduction

The frequency reduction mechanism is initialized by selecting the lowest quality for the first smoothing window. Then, the remaining available bandwidth is utilized to acquire the future chunks (these are of course stored in the prefetching buffer). The frequency reduction mechanism can be explained using the example in Figure 56.

In Figure 56 (a), the algorithm is initialized with the lowest quality level for the first smoothing window ($L^t = 0 \ \forall t \in [t_0, t_5]$). The remaining available bandwidth can then be used to prefetch chunks which may be useful during the second smoothing window ($t \in [t_6, t_{11}]$). At the end of first smoothing window, the highest selected quality level in the prefetching buffer is 1. If the $\widehat{B^t}$ as estimated from data in $[t_0, t_5]$ is sufficient to acquire all the missing chunks for layer 1, then the quality level will be increased for the next smoothing window and $L^t = 1 \ \forall t \in [t_6, t_{11}]$, as shown in Figure 56 (b). For frequency reduction the value of $\widehat{B^t}$ is calculated using the average value for the available bandwidth in the last smoothing window. On the other hand, if the chunks available in the prefetching buffer are insufficient to sustain the current quality level, as it happens for the third smoothing window in Figure 56 (b), L^t will be correspondingly reduced. In this example, we have that $L^t = 0 \ \forall t \in [t_{12}, t_{17}]$.

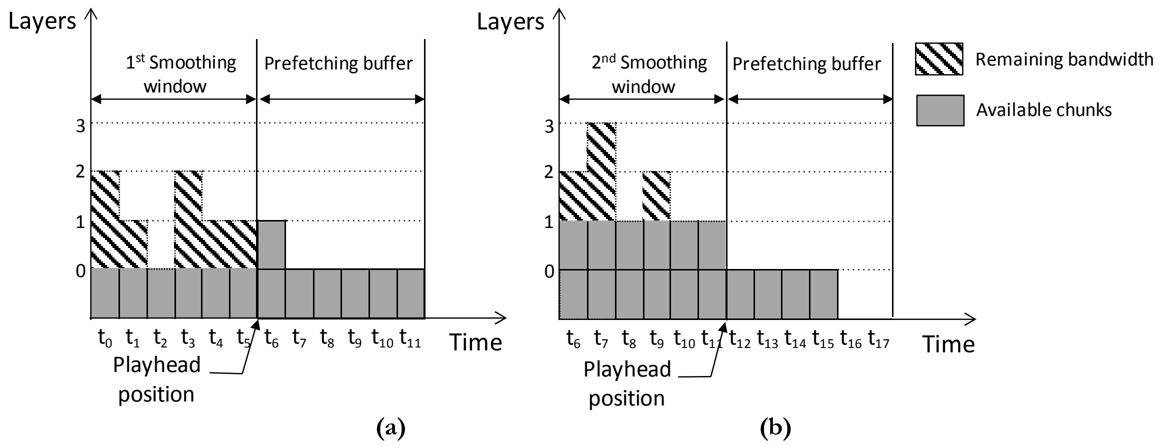


Figure 56: Frequency Reduction Mechanism

Thus the frequency smoothing mechanism ensures a constant quality level for each smoothing window. The bandwidth change in current smoothing window will ultimately affects the quality level in next smoothing window.

5.4.2.2 Mean-based frequency reduction

In this strategy the quality level decided for each smoothing window is the quality level corresponding to the average bandwidth received in the previous smoothing window. Similarly to the prefetching-based strategy, the mean-based frequency strategy is initialized by selecting the lowest quality for the first smoothing window. At the end of the first smoothing window, we decide the quality level corresponding to the average bandwidth observed in this smoothing window, as the target quality level for the second smoothing window. The chunks are requested in a conservative manner within the smoothing window (lowest layers first). If there is still a remaining bandwidth, it will be utilized to acquire chunks for the next smoothing window. This process is repeated for all the smoothing windows of the stream. A pseudo code of this algorithm is presented in Table 7.

The difference between prefetching-based strategy and the mean-based strategy is that the first one decides the quality level depending on what exists in the prefetching buffer. However the mean-based one relies on the average quality level observed in the previous smoothing window. The first one is more realistic while the second one tries to converge all the time toward a mean quality level. In the next section we study the performance of these two smoothing strategies in addition to the amplitude reduction strategy.

Table 7: Mean-based frequency reduction algorithm

```
1.  Window= 1 // the first smoothing window
    $S_w$ = constant // the smoothing window size
    Target_layer = 0 // the target layer that we hope to play during a smoothing window
    Aquire_chunks (Target_layer) // scheduling function (request the chunks corresponding to the target layer)
    Remaining_bandwidth = 0 // the total remaining bandwidth of the current smoothing window
    Average_bandwidth = Bandwidth [Window] / $S_w$ // average bandwidth of the current window
2.  while not end of stream
      Sw = Sw+1
      Target_layer = Layer (Average_bandwidth) // layer corresponding to Average_bandwidth
      Aquire_chunks (Target_layer)
      If (Bandwidth [Window] + Remaining_bandwidth) > Average_bandwidth * $S_w$
          Remaining_bandwidth = Bandwidth [Window] + Remaining_bandwidth – (Average_bandwidth * $S_w$ ) //
          Bandwidth [Window]: total bandwidth received in Window
          Played_layer = Target_layer
      else
          Remaining_bandwidth =0
          Played_layer = Layer (Bandwidth [Window] + Remaining_bandwidth)/ $S_w$
      end if
      Average_bandwidth = Bandwidth [Window] / $S_w$
    end while
```

5.5 Performance Evaluation

In this section, we present the performance evaluation of our proposed playout smoothing mechanisms for layered streaming. The proposed smoothing mechanisms were implemented in our test bed within the framework of ENVISION-LaBRI-Lab [173].

We first need to define the relevant metrics that reflect the key features of perceived video quality. We focus mainly on three important points: layers changes, longest sequence and the unused chunks.

Layer changes: Represents the number of layers changes during the studied video stream.

Longest sequence: Represents the size of the longest smoothed sequence of the stream (sequence of the same quality level).

Unused chunks: Represents the number of chunks that are sacrificed (not played) in each smoothing window, in order to ensure the smoothness.

Three scenarios are considered in this study. First we present a real scenario of smoothing and we show the difference in terms of layers change between the proposed smoothing mechanisms comparing to the row stream. In second time, we are interested in the impact of the smoothing window size on the performance of the proposed smoothing algorithms. Finally we compare the frequency reduction algorithm with the Layer2P2 in terms of number of layers changes.

5.5.1 Scenario 1

In order to evaluate the three smoothing strategies, namely amplitude reduction and frequency reduction (prefetching-based and mean-base), we consider a scenario where the video stream is composed of 8 layers (each layer is streamed at 100 kbps) under bandwidth variation (from 100 kbps to 900 kbps). The duration of the video is 400 seconds. We fixed first the smoothing window size at 15 seconds, and we study the behavior of the three proposed algorithms.

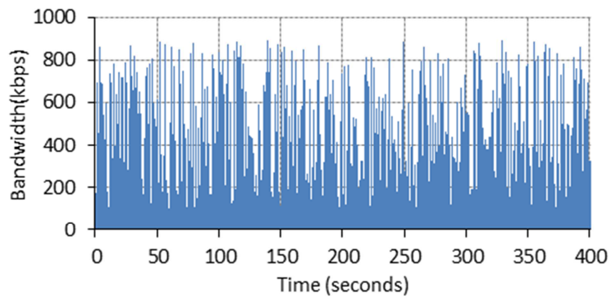


Figure 57: bandwidth variation

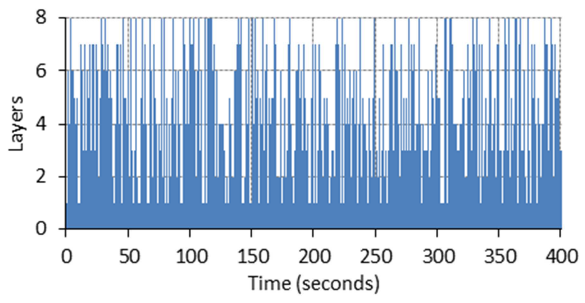


Figure 58: Row stream

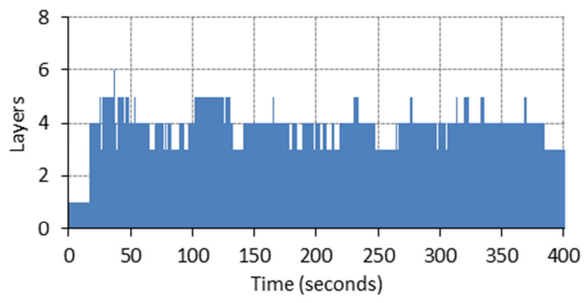


Figure 59: Amplitude reduction

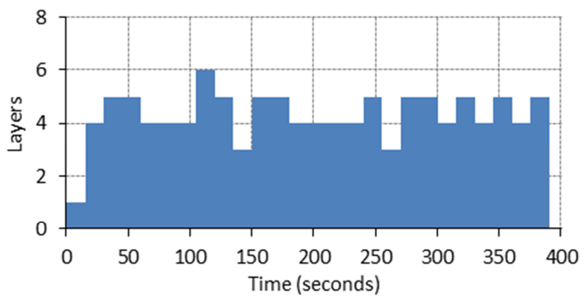


Figure 60: Frequency reduction (prefetching-based)

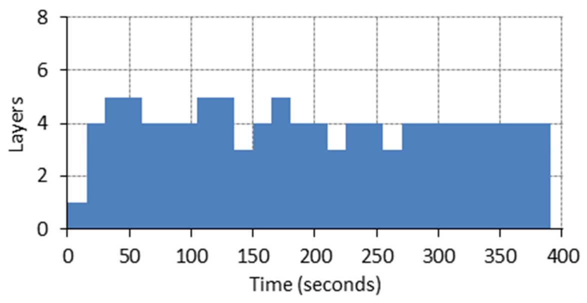


Figure 61: Frequency reduction (mean-based)

In Figure 57 plots the aggregated download bandwidth in the receiver peer. Figure 58 represents the layers changes without any smoothing action on the stream. In this figure we note that the quality level fluctuates with the fluctuation of the aggregated upload bandwidth of the peer. In Figure 59 we apply the amplitude reduction algorithm on the row stream. In this case we note that the quality level fluctuation is drastically reduced, and the user can enjoy a relative stable video quality comparing to row stream. Indeed, this figure shows that the quality level change does not exceed 1 level in most cases. However, they could be a fluctuation of the quality within a short time period (from second

200 to 250 for example). In Figure 60, we apply the prefetching-based frequency reduction algorithm on the same row stream. We note that the quality level is stable at least for a smoothing window, however we note some big quality level jumps in some cases (jump of 2 layers from 149s to 150s for example). The stream is smoother in the case of mean-based frequency reduction strategy, where we note the stability of the quality level for several smoothing windows (for example from 270s to 380s).

In Figure 62 we study the performance of the three proposed smoothing algorithms regarding the parameters defined above, under the conditions of the scenario 1 (smoothing window size = 15).

In terms of layers change, the results obtained in Figure 62 confirm the observations in Figure 58, Figure 59, Figure 60 and Figure 61. The frequency reduction algorithm (both prefetching-based and mean-based) presents the best performance in terms of number of layer changes. It diminishes the row stream layers changes about 20 times and it performs ¼ of layer changes of amplitude reduction algorithm.

Secondly in terms of unused chunks, naturally the not-exploited chunks are null in the case of the row stream, since the quality level follows the bandwidth fluctuation. However we note an important portion of unused chunks in the case of frequency reduction algorithms compared to the amplitude reduction, because the formers tries to smooth the stream along a larger portion of time (at least a smoothing window), however the amplitude reduction smooth the stream locally.

As expected, regarding the longest sequence in the stream, we note that the frequency reduction algorithms outperform the amplitude reduction. This can be explained by the fact that the frequency reduction algorithms try to keep a stable quality level for a least one smoothing window. In the contrary of the amplitude reduction that tries to reduce the jump of quality from a time slot to another.

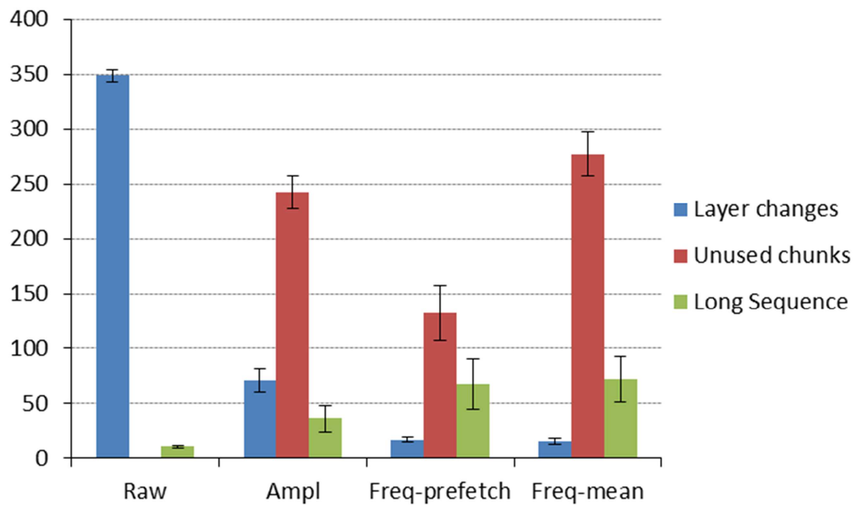


Figure 62: 1st scenario analysis

5.5.2 Scenario 2: Smoothing window size impact

In this scenario, we are interested in the impact of the smoothing window size on the performance of amplitude and frequency reduction algorithms. For that reason we performed several tests while varying the size of the smoothing window from 5 to 30 seconds.

5.5.2.1 Impact of smoothing window size on number of layers changes

Figure 63 shows the evolution of the number of layers changes while varying the smoothing window size for the amplitude reduction and frequency reduction algorithms. We note that the general trend is the decrease of the number of layers changes with the increase of the smoothing window size. It is obviously explained in the case of frequency reduction algorithm, since this later maintain the same quality level within at least smoothing window. The largest the smoothing window is, the lowest the global number of layers changes is. The number of layers changes in the case the amplitude reduction algorithm is inversely proportional to the smoothing window size, too. This can be explained by the fact that the amplitude reduction algorithm tries to reduce the quality level jump regarding the average quality level experienced in the previous smoothing window. A large smoothing window, means that the algorithms converge to same quality level (the mean of the smoothing window) for a long period of time. Thus, number of layers change is reduced.

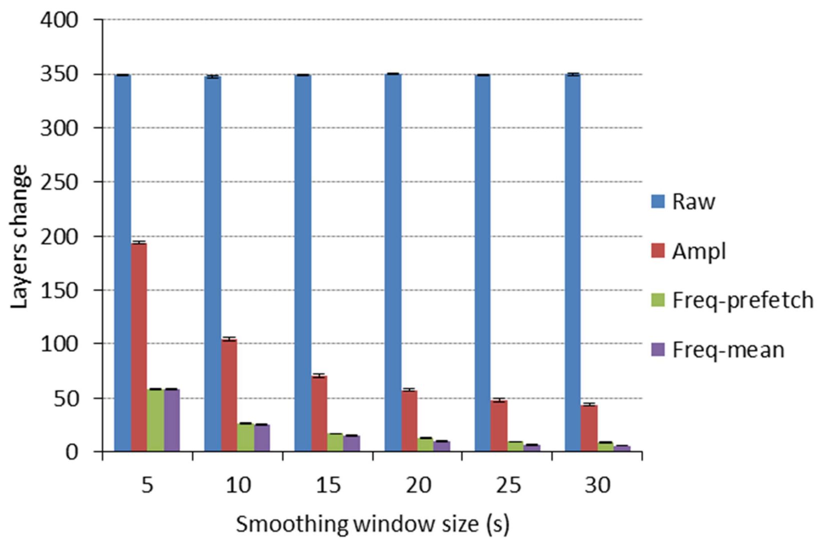


Figure 63: Impact of smoothing window size on number of layers changes

5.5.2.2 Impact of smoothing window size on number of unused chunks

We note in Figure 64 that the number of unused chunks is higher in the case of the frequency reduction algorithms compared to the amplitude reduction. In addition, the general trend is the increase of the number of unused chunks with the expansion of the smoothing window for the frequency reduction scenario. Indeed, a large smoothing windows leads to a smoothing action over a large interval of time, and consequently a large number of scarified chunks. We can see this through the following example: assume a smoothing window of 10 seconds, and the number of chunks (of 1 second each) received within the smoothing window is 19 chunks. The maximum smoothed quality level possible is layer 0. The number of unused chunks in this case is 9 chunks. However, for the same amount of received chunks (19 chunks) in the same period of time (10 seconds) with a smoothing window of 5 seconds (2 smoothing windows), we can have a first window smoothed at layer=1 (layer 0 + layer 1) and the second window smoothed at layer 0. In this case the number of unused chunks is 4.

5.5.2.3 Impact of smoothing window size on longest sequence

We are interested in the evolution of the largest sequence for each smoothing algorithm (Figure 65). We note that this parameter is in accordance with the size of the smoothing window in the case of the frequency reduction algorithms since they act on a smoothing window, the larger it is, the longer the sequence is. However the amplitude reduction is not very impacted by the smoothing window size.

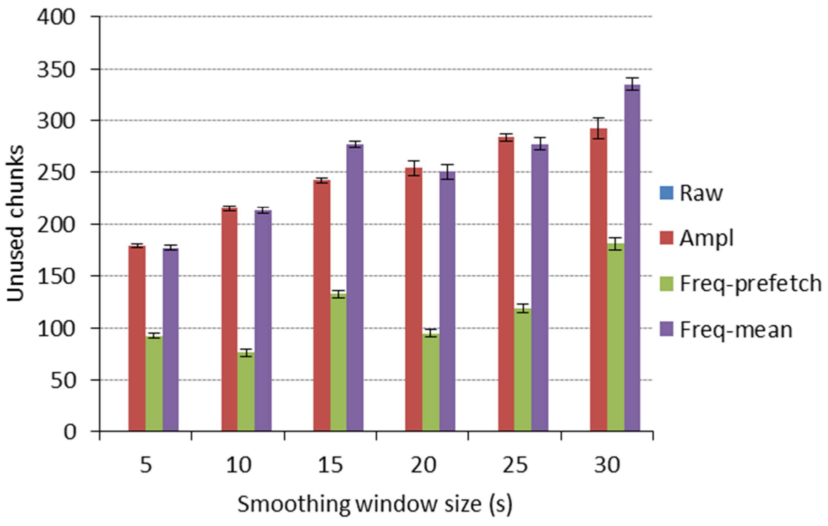


Figure 64: Impact of smoothing window size on number of unused chunks

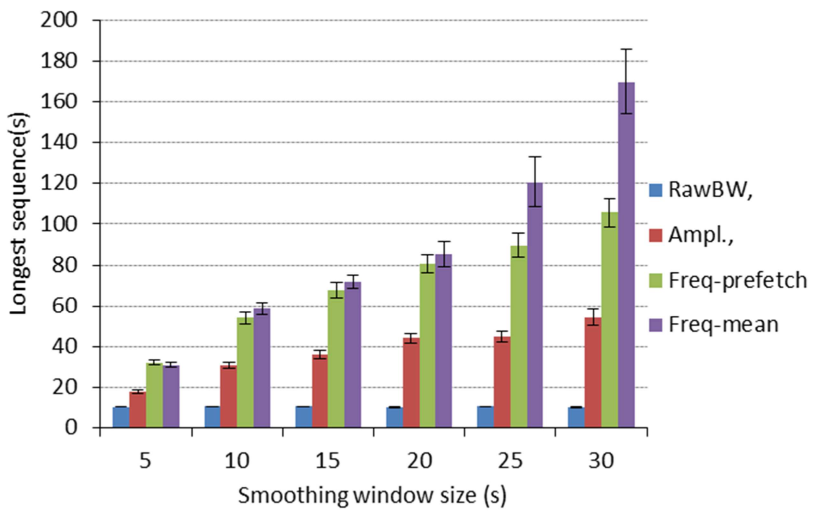


Figure 65: Impact of smoothing window size on number of longest sequence

5.5.3 Scenario 3: Comparison with layerP2P

Figure 66 shows the number of layer changes at different intervals for both mechanisms: LayerP2P and the frequency reduction mechanism (prefetching-based). The interval on x-axis represents the length of time between successive changes in the bandwidth. As expected, the number of layer changes decreases as this interval increases. Thus, it is found that bandwidth changes have an obvious effect on quality variation. When compared with the state of the art, our proposed mechanism has fewer layer changes due to the smoothing mechanism which adjusts the quality level for each smoothing window. The lower number of layer changes in the proposed mechanism is due to the smoothing window, during which we aims to keep the same quality level. As a result, the frequency of changes in quality level is minimized. Comparatively, LayerP2P system has higher number of quality changes due to the absence of an efficient smoothing mechanism.

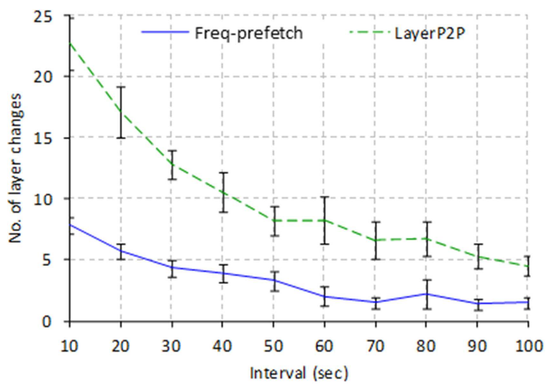


Figure 66: Number of Layer Changes

5.6 Conclusion

In this chapter, we propose a playout smoothing mechanism for layered streaming in P2P network that selects appropriate stream layers while minimizing the number of layer changes. Firstly, we proposed the mechanism for amplitude reduction which aims to reduce the size of jumps in the quality level of the layered stream. Then we proposed two mechanisms for the frequency reduction which aims to reduce the number of layer changes due to varying network conditions. The first frequency reduction algorithm relies on the prefetched chunks to decide the quality level while the second one relies on the average quality level observed in the previous smoothing window to decide the target quality level. The proposed mechanism has been implemented in real test bed and its performance evaluation shows the effectiveness of the proposed mechanism in terms of smoothness of the stream. We found also that the frequency reduction algorithms performs the best smoothness level, however they present a high percent of unused chunks (lower quality level), especially in large smoothing windows.

The main limitation of the proposed mechanisms is the lag introduced by the frequency reduction mechanisms, in particular, the smoothing window size parameter. As future work, we plan to study techniques to achieve a trade-off between the smoothing quality and the liveness of the stream by acting on the smoothing window size.

Chapter 6

6 Conclusion and Perspectives

In this chapter we conclude the dissertation by summarizing the major contributions and suggesting some guidelines for future perspectives.

The exponential growth experienced by P2P Streaming applications along with the diversification of end-user context and their growing expectation for better experience give rise for many challenges. Efficient resources management and content scheduling that better take advantage of the available bandwidth in the network in order to provide good QoE for users, and ensuring a smooth playout delivery are the key challenges tackled in this thesis. The brief summary of the contributions is presented below.

6.1 Summary of Key Contributions

In this dissertation, we theoretically study the scheduling problem in pull-based P2P video streaming and we model it, in general, as an assignment problem. In order to optimize the throughput and the delivery ratio of the system, we design new scheduling algorithm (AsSched) along with chunks prioritization mechanism for P2P layered streaming. In this context the problem is modeled as generalized assignment problem (GAP) that we proposed a heuristic to solve it. We adapt then the proposed solution to non-layered streaming (NasSched). The problem in this case is modeled as *m*-cardinality assignment problem and we propose a new solution based on Hungarian algorithm to solve it. Performance evaluation of AsSched against the traditional scheduling systems resulted in improved packet delivery ratio and effective bandwidth utilization.
We then tackled the bandwidth allocation problem in P2P layered streaming. For that we proposed a bandwidth allocation scheme that distributes appropriate amount of bandwidth to the appropriate peers while ensuring a minimum quality level to all peers. The proposed scheme relies on set of auctions organized by the upstream peer to allocate its bandwidth. An auction is organized for each layer, starting first by the lower layers, then the upper ones. This ensures that the lower layers are transmitted via the best links, consequently increasing the system throughput. Extensive simulations are performed to study the effectiveness of the proposed mechanism. We compare its performance with four other strategies. We studied different metrics that are essential in determining the performance of our proposed mechanism. We were interested first on the bandwidth utilization ratio. The results shows that the proposed mechanism outperform the others mechanism and ensure up to 90% of bandwidth use. Then we studied the quality level satisfaction of peers of our system. The obtained results confirm those found for the bandwidth utilization.

In the last chapter, we proposed a playout smoothing mechanism for layered streaming in P2P networks that minimizes the number of layer changes under varying network conditions while at the same time achieving a high delivery ratio. This was achieved through three different contributions. First, we proposed mechanism for both amplitude reduction to reduce the size of jumps and two variant of algorithms for the quality frequency reduction which aims to keep a stable quality level at least for an interval of time called smoothing window. We studied in detail the performance of these three algorithms in terms of number of layers changes, longest stable sequence and the unused chunks. We found that the frequency reduction algorithms performs the best smoothness level, however they present a high percent of unused chunks (lower quality level), especially in large smoothing windows, and introduce a liveness lag which is relative to the smoothing window size.

6.2 Open Issues and Future Work

Future perspectives for our work can develop along several directions. As the first challenge, we plan to study techniques to achieve a trade-off between the smoothing quality and the liveness of the stream by acting on the smoothing window size. In fact the liveness of the stream is very vulnerable to the smoothing windows; a long smoothing window leads to a large liveness lag.
Moreover, we would like to tackle the problem of quality bottleneck in P2P layered streaming networks. Peers usually have different download/upload capacities. When video content flows along the overlay paths between peers, the network throughput is often adjusted by the low capacity peers in the middle, and the streaming rates of the downstream peers reduce drastically. In order to fully take advantage of peer's upload capacity, bandwidth heterogeneity is an important concern in designing efficient P2P streaming systems. In layered streaming, the bottleneck problem results in the delivery of poor quality to the distant peers (from the source). As a result, these peers having enough capacity can be restricted to a low quality level. We refer this problem as quality bottleneck problem. In the future work, we aim to study the optimal organization of peers in order to overcome the quality bottleneck problem for layered streaming P2P systems.

Finally, the performance evaluations for the proposed mechanisms for bandwidth allocation and scheduling for the video streaming over P2P are performed using network simulation. However, these simulations are not enough to accurately judge the reel performance of any mechanism. Therefore, we plan to implement the core components of these mechanisms in LaBRI-SVC lab.

Appendix

Publications Arising From This Thesis

1. Abbas Bradai, Obaid Abbassi, Raul Landa and Toufik Ahmed, "An Efficient Playout Smoothing Mechanism for Layered Streaming in P2P Networks", Springer, Peer-to-Peer Networking and Applications (PPNA), 2012.

2. Abbas Bradai, Toufik Ahmed, "On the Optimal Scheduling in Pull-based Real-Time P2P Streaming Systems", in proceeding of IEEE International Conference of Communications (IEEE ICC 12), Ottawa, 10 – 15, June 2012.

3. Abbas Bradai, Toufik Ahmed, "Differentiated Bandwidth Allocation in P2P Layered Streaming", In procedding of IEEE CAMAD'12, Barcelone, 2012

4. Abbas Bradai, Toufik Ahmed, "Dynamic Bandwidth Allocation in P2P Layered Streaming", submitted to IEEE Transaction on parallel and distributed systems

5. A Bradai, T Ahmed, "An efficient algorithm for selection and management of Island multicast", The 8th IEEE International Consumer Communication & Networking Conference (CCNC), Las Vegas, 2011.

Reports

1. A. Bradai, T. Ahmed, U. Abbasi, S. Medjiah, et al. ENVISION deliverable D5.1, "Initial Specification of Metadata Management, Dynamic Content Generation and Adaptation", January 2011, FP7 ICT ENVISION project, http://www.envision-project.org

2. T. Ahmed, A. Bradai, U. Abbasi, S. Medjiah, et al. ENVISION deliverable D3.1, "Refined Specification of the ENVISION Interface, Network Monitoring and Network Optimisation Functions", January 2012, FP7 ICT ENVISION project, http://www.envision-project.org

3. A. Bradai, T. Ahmed, U. Abbasi, S. Medjiah, et al. ENVISION deliverable D5.2, "Refined Specification of Metadata Management, Dynamic Content Generation and Adaptation, Adaptation and Caching Node Functions", January 2012, FP7 ICT ENVISION project, http://www.envision-project.org

4. A. Bradai, T. Ahmed, U. Abbasi, S. Medjiah, et al. ENVISION deliverable D5.3, " Final Specification of Metadata Management, Dynamic Content Generation and Adaptation, Adaptation and Caching Node Functions", June 2012, FP7 ICT ENVISION project, http://www.envision-project.org

References

[1]. Cisco Systems. Cisco visual networking index: Forecast and Methodology, 2010 -2015. 2011

[2]. www.caida.org

[3]. www.qq.com

[4]. www.telegeography.com

[5]. www.pplive.com/en/index.html

[6]. http://zattoo.com/view

[7]. D. S. Milojicic, V. Kalogeraki, R. Lukose, K. Nagaraja, J. Pruyne, B. Richard, S. Rollins, Z. Xu, "Peer-to-Peer Computing", Technical Report, HP Laboratories, Palo Alto, CA, HPL-2002-57, March 2002.

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

[9]. O. Abboud, K. Pussep, A. Kovacevic, and R. Steinmetz, "Enabling Resilient P2P Video Streaming: Survey and Analysis", Multimedia System Journal (MMSJ), 17(2):1–21, 2011.

[10]. J. Buford, H. Yu, K. Lua, "P2P Networking and Applications", Computers, 2009.

[11]. http://tools.ietf.org/html/draft-ietf-avt-rtp-svc-06#page-15.

[12]. Y. Huang, T. Z. Fu, D.-M. Chiu, J. C. Lui, and C. Huang, "Challenges, Design and Analysis of a Large-Scale P2P-VoD System," in Proc. ACM SIGCOMM'08, Aug. 2008.

[13]. M. Mushtaq and T. Ahmed, "QoS Provisioning for Video Streaming Over SP-Driven P2P Networks Using Admission Control", IEEE, France, 2009.

[14]. O. Abboud, T. Zinner, K. Pussep, and R. Steinmetz, "On the Impact of Quality Adaptation in SVC-based P2P Video-on-Demand Systems", In ACM Multimedia Systems Conference (MMSys). 2011.

[15]. X. Xiao, Y. Shi, and Y. Gao, "On Optimal Scheduling for Layered Video Streaming in Heterogeneous Peer-to-Peer Networks", In Proc. of ACM international conference on Multimedia, 2008, pp. 785–788.

[16]. Napster. Available at http://www.napster.com/

[17]. Bittorrent. Available at http://www.bittorrent

[18]. S. German, W. Markus, U. Herwig, "Search Methods in P2P Networks: A Survey", Lecture Notes in Computer Science, Volume 3473/2006, pp. 59-68, 2006.

[19]. S. Ratnasamy, P. Francis, M. Handley, R. Karp, S. Shenker, "A scalable content-addressable network", Technical report, ACM SIGCOMM, 2001.

[20]. I. Stoica, R. Morris, D. Liben-Nowell, D.R. Karger, M.F. Kaashoek, F. Dabek, H. Balakrishnan, "Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications", IEEE/ACM Trans. Net., vol. 11, no. 1, pp. 17–32, 2003.

[21]. B. Y. Zhao, L. Huang, J. Stribling, S. C. Rhea, A. D. Joseph, J. D. Kubiatowicz, "Tapestry: A resilient global-scale overlay for service deployment", Technical report, IEEE Journal, January 2004.

[22]. B. Y. Zhao, J. D. Kubiatowicz, A. D. Joseph, "Tapestry: An infrastructure for fault-tolerant wide-area location and routing", Technical report, April 2001.

[23]. I. Clarke, O. Sandberg, B. Wiley, T. W. Hong, "Freenet: A distributed anonymous information storage and retrieval system", Technical report, July 2000.

[24]. T. Klingberg, R. Manfredi, "Gnutella 0.6", Technical report, June 2002. Available at: http://rfc-gnutella.sourceforge.net/src/rfc-0_6-draft.html.

[25]. J. Buford, H. Yu, E. K. Lua. "P2P Networking and Applications". Morgan Kaufmann, 2008.

[26]. Djamal-Eddine Meddour, Mubashar Mushtaq, and Toufik Ahmed. Open Issues in P2P Multimedia Streaming. In IEEE ICC 2006 Workshop on Multimedia Communications Workshop (MultiCom), June 2006.

[27]. S. Agarwal, J. Singh, P. e Dube, "Analysis and implementation of gossip-based P2P streaming with distributed incentive mechanisms for peer cooperation", Advances in Multimedia, 2007(2):1–12.

[28]. X. Zhang, J. Liu, B. Li, Yum, Y.-S.P, "DONet/CoolStreaming: A data-driven overlay network for live media streaming". In Proc of IEEE INFOCOM, Miami, USA, 2005.

[29]. V. Vishnumurthy and P. Francis. On Heterogeneous Overlay Construction and Random Node Selection in Unstructured P2P Networks. In Proceedings of IEEE Infocom, Barcelona 2006.

[30]. X. Zhang, Q. Zhang, Z. Zhang, G. Song, AND W. Zhu, "A construction of locality-aware overlay network: moverlay and its performance", IEEE JSAC Special Issue on Recent Advances on Service Overlay Networks, 2004.

[31]. P. Shi, H. Wang, Y. Gang, and X. Yuan, "ACON: Adaptive construction of the overlay network in CDN-P2P VoD system," in Communication Software and Networks (ICCSN), 2011 IEEE 3rd International Conference on, 2011, pp. 182–187.

[32]. Mubashar Mushtaq and Toufik Ahmed, "P2P-based Mobile IPTV: Challenges and Opportunities", in proc. of the 6th IEEE /ACS International Conference on Computer Systems and Applications (AICCSA'08), April, 2008.

[33]. T. Ahmed, A. Asgari, A. Mehaoua, E. Borcoci, L. Berti-Équille, and K. Georgios. "End-to-End Quality of Service Provisioning Through an Integrated Management System for Multimedia Content Delivery" in Elsevier Computer Communication Journal SI on Emerging Middleware for Next Generation Networks, (accepted in 2006), Vol. 30, Issue 3, pp. 638 -651, ISSN:0140-3664, February 2007

[34]. Mushtaq, M., Ahmed, T., "P2P-based mobile IPTV: Challenges and opportunities", IEEE/ACS Int. Conf. on Computer Systems and Applications, AICCSA'08, 31 March - 4 April 2008, pp.975-980.

[35]. N. Magharei and R. Rejaie. “PRIME: Peer-to-Peer Receiver- driven Mesh-based Streaming”. In Proc of IEEE INFOCOM, 2007.

[36]. http://www.pptv.com/

[37]. Y. Guo, K. Suh, J. Kurose, and D. Towsley, "P2Cast: Peer-to-Peer Patching Scheme for VoD Service". In Proc of the 12th World Wide Web Conference (WWW-03), Budapest, Hungary, May 2003.

[38]. S. Annapureddy, S. Guha, C. Gkantsidis, D. Gunawardena, and P. Rodriguez, “Is High-Quality VoD Feasible Using P2P Swarming?” In Proc of 16th Int’l Conf. World Wide Web (WWW), 2007.

[39]. J. Noh, A. Mavlankar, P. Baccichet, and B. Girod. “Time-shifted streaming in a peer-to-peer video multicast system”. In Proc of the 28th IEEE conference on Global telecommunications, GLOBECOM’09, Piscataway, NJ, USA, 2009.

[40]. Y. Liu and G. Simon, “Peer-assisted time-shifted streaming systems: Design and promises,” In IEEE ICC, Germany, 2009.

[41]. D. Gallo, C. Miers, V. Coroama, T. Carvalho, V. Souza, and P. Karlsson, “A Multimedia Delivery Architecture for IPTV with P2P-Based Time-Shift Support”. In Proc. of 6th IEEE CCNC, Las Vegas, USA, 2009.

[42]. Y. Liu and G. Simon, “Distributed Delivery System for Time-Shifted Streaming System”. In Proc. of IEEE Conference LCN, 2010

[43]. Y. Chu, S. G. Rao, S. Seshan, H. Zhang, “A case for End System Multicast,” Selected Areas in Communications, Vol. 20, No. 8. Oct. 2002, pp. 1456-1471, doi:10.1109/JSAC.2002.803066.

[44]. X. Zhang, J. Liu, B. Li, and T.-S. P. Yum, “CoolStreaming/DONet: A data-driven overlay network for live media streaming,” Proc. The IEEE INFOCOM. IEEE Press, Mar. 2005, pp. 2102-2111, doi: 10.1109/INFCOM.2005.1498486.

[45]. N. Magharei and R. Rejaie, “PRIME: Peer-to-Peer Receiver-drIven Meshbased Streaming,” Proc. IEEE INFOCOM, IEEE Press, May 2007. pp. 1415–1423, doi:10.1109/INFCOM.2007.167.

[46]. D. Jurca, J. Chakareski, J.P. Wagner, P. Frossard, “Enabling Adaptive Video Streaming in P2P Systems,” Communications Magazine, IEEE, Vol. 45, No. 6. 2007, pp. 108-114, doi:10.1109/MCOM.2007.374427.

[47]. Y. Guo, S. Mathur, K. Ramaswamy, S. Yu, and B. Patel, “PONDER: Performance Aware P2P Video-on-Demand Service,” Proc.Global Telecommunications Conference, 2007. GLOBECOM '07. IEEE Press, Nov. 2007, pp.225 – 230, doi:10.1109/GLOCOM.2007.50.

[48]. X. Zhang, J. Liu, B. Li, and T. Yum. CoolStreaming/DONet: a data-driven overlay network for peer-to-peer live media streaming. In Proc. of IEEE International Conference on Computer Communications (INFOCOM’05), pages 2102 – 2111, Miami, FL, March 2005.

[49]. V. Pai, K. Kumar, K. Tamilmani, V. Sambamurthy, and A. Mohr. Chainsaw: Eliminating trees from overlay multicast. In Proc. of International Workshop on Peer-to-Peer Systems (IPTPS'05), pages 127 – 140, Ithaca, NY, February 2005.

[50]. Y. Zhou, D. M. Chiu, and J. C. S. Lui, "A simple model for analyzing p2p streaming protocols," Proc. IEEE ICNP'07, IEEE Press, Oct. 2007, pp. 226-235, doi:10.1109/ICNP.2007.4375853.

[51]. N. Carlsson and D. L. Eager, "Peer-assisted On-demand Streaming of Stored Media using BitTorrent-like Protocols," Proc. IFIP/TC6 Networking '07, Vol. 4479, Springer, May 2007, pp. 570-581, doi: 10.1007/978-3-540-72606-7_49.

[52]. Z. Li, J.N. Cao, G.H. Chen, "ContinuStreaming: Achieving high playback continuity of Gossip-based Peer-to-Peer streaming," Proc. the 22nd IEEE International Parallel and Distributed Processing Symposium (IPDPS 2008), IEEE Press, Apr. 2008, pp. 1-12, doi: 10.1109/IPDPS.2008.4536252

[53]. http://www.pplive.com/

[54]. J.M. Lv, X.Q. Cheng, Q. Jiang, J. Ye, T.Y. Zhang, S.M. Lin, L. Wang, "LiveBT: Providing Video-on-Demand Streaming Service over BitTorrent Systems," Proc. the Seventh International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT'2007), IEEE Press, Dec. 2007, pp. 501-508, doi:10.1109/PDCAT.2007.27.

[55]. A. Vlavianos, M. Iliofotou, and M. Faloutsos, "Bitos: Enhancing bittorrent for supporting streaming applications," Proc. 9th IEEE Global Internet Symposium 2006 (INFOCOM 2006), IEEE Press, Apr. 2006, pp. 1-6, doi:10.1109/INFOCOM.2006.43.

[56]. Y. Huang, T. T. J. Fu, D. M. Chiu, J. C. S. Lui, and C. Huang, "Challenges, Design and Analysis. of a Large-scale P2P VoD

[57]. X. Zhang, J. Liu, B. Li, and T. Yum, "CoolStreaming/DONet: A data-driven overlay network for peer-to-peer live media streaming". In Proc. of IEEE INFOCOM'05, pages 2102–2111, Miami, FL, March 2005.

[58]. V. Agarwal and R. Rejaie. Adaptive multi-source streaming in heterogeneous peer-to-peer networks. In Proc. of SPIE/ACM Multimedia Computing and Networking (MMCN'05), pages 13–25, San Jose, CA, January 2005.

[59]. G. Kowalski and M. Hefeeda. Empirical analysis of multi-sender segment transmission algorithms in peer-to-peer streaming. In Proc. of IEEE International Symposium on Multimedia (ISM'09), pages 243–250, San Diego, CA, December 2009.

[60]. M. Zhang, Y. Xiong, Q. Zhang, L. Sun, and S. Yang "Optimizing the throughput of data-driven peer-to-peer streaming", IEEE Transactions on Parallel and Distributed Systems, 20(1):97–110, January 2009.

[61]. J. Chakareski and P. Frossard "Utility-based packet scheduling in P2P mesh-based multicast", In Proc. of SPIE International Conference on Visual Communication and Image Processing (VCIP'09), page 72571S, San Jose, CA, January 2009.

[62]. Y. Cai, A. Natarajan, and J. Wong "On scheduling of peer-to-peer video services", IEEE Journal on Selected Area in Communications, 25(1):140–145, January 2007.

[63]. D. Xu, M. Hefeeda, S. Hambrusch, and B. Bhargava, "On peer-to-peer media streaming," Proc. 22nd International Conference on Distributed Computing Systems, IEEE Press, July 2002.

[64]. J.B. Kwon, H.Y. Yeom, "Distributed Multimedia Streaming over Peer-to-Peer Networks," Euro-Par 2003 Parallel Processing (2003), Vol. 2790, Springer, 2004.

[65]. J. Li, "PeerStreaming: An On-Demand Peer-to-Peer Media Streaming Solution Based On A Receiver-Driven Streaming Protocol," Proc. IEEE 7th Workshop on Multimedia Signal Processing, IEEE Press, Oct.2005, pp. 1-4, doi: 10.1109/MMSP.2005.248677.

[66]. X.F. Liao, H. Jin, Y. Liu, L. Ni, D. Deng, "Anysee: Peer-to-Peer live streaming," Proc. INFOCOM 2006, IEEE Press, Apr. 2006, pp. 1-10, doi: 10.1109/INFOCOM.2006.288.

[67]. M. Zhang, L. Zhao, Y. Tang, J.G. Luo, S.Q.Yang, "Large scale live media streaming over peer-to-peer networks through global internet," Proc. the ACM workshop on Advances in peer-to-peer multimedia streaming 2005, ACM Press, Nov. 2005.

[68]. K. Suh, C. Diot, J. Kurose, L. Massoulie, C. Neumann, D. Towsley, M.Valleo, "Push-to-Peer Video-on-Demand system: design and evaluation," IEEE Journal on Selected Areas in Communications (JSAC) , Vol. 25, Dec. 2007.

[69]. T. Locher, R. Meier, S. Schmid, R. Wattenhofer, "Push-to-Pull Peer-to-Peer Live Streaming," Proc. DISC 2007, Vol. 4731, Springer, 2007.

[70]. C.Gkantsidis, and P.Rodriguez, "Network coding for large scale content distribution," Proc. IEEE INFOCOM 2005, IEEE Press, Mar. 2005.

[71]. M. Wang, and B.Li, "Lava: A reality check of network coding in peer-to-peer live streaming," Proc. IEEE INFOCOM 2007, IEEE Press, May 2007.

[72]. M. Wang, and B.Li, "R2: random push with random network coding in live peer-to-peer streaming," IEEE Journal on Selected Areas in Communications, Vol. 25, Dec. 2007.

[73]. H. Chi and Q. Zhang, "Efficient search and scheduling in P2P-based media-on-demand streaming service," IEEE Journal on Selected Areas in Communications, Vol. 25, Jan. 2007.

[74]. S. Annapureddy, S. Guha, C. Gkantsidis, D. Gunawardena, P. Rodriguez, "Exploring VoD in P2P Swarming Systems," Proc. Of INFOCOM 2007, IEEE Press, May 2007.

[75]. S. Annapureddy, S. Guha, C. Gkantsidis, D. Gunawardena, P. Rodriguez, "Exploring VoD in P2P Swarming Systems," Proc. Of INFOCOM 2007, IEEE Press, May 2007.

[76]. Y. Chu, SG. Rao, S. Seshan, and H. Zhang. Enabling conferencing applications on the Internet using an overlay multicast architecture. In Proceedings of ACM SIGCOMM, pages 55–67, August 2001.

[77]. N. Magharei and R. Rejaie. Prime: Peer-to-peer receiverdriven mesh-based streaming. In Proceedings of IEEE INFOCOM, pages 1415–1423, May 2007.

[78]. D. Tran, K. Hua, and T. Do. Zigzag: An efficient peer-to-peer scheme for media streaming. In Proceedings of IEEE INFOCOM, pages 1283–1292, March 2003.

[79]. X. Zhang, J. Liu, B. Li, and T. Yum. Coolstreaming/donet: A data-driven overlay network for efficient live media streaming. In Proceedings of IEEE INFOCOM, pages 2102–2111, March 2005.

[80]. W. Ooi. Dagster: Contributor-aware end-host multicast for media streaming in heterogeneous environment. In Proceedings of ACM/SPIE MMCN, pages 77–90, January 2005.

[81]. T.B. Ma, Sam C.M. Lee, John C.S. Lui, and David K.Y. Yau. A game theoretic approach to provide incentive and service differentiation in P2P networks. In Proceedings of ACM SIGMETRICS/PERFORMANCE, pages 189–198, June 2004.

[82]. Y. Yan, A. El-Atawy, and E. Al-Shaer. Ranking-based optimal resource allocation in peer-to-peer networks. In Proceedings of IEEE INFOCOM, pages 1100–1108, May 2007.

[83]. http://peerguardian.en.softonic.com

[84]. B. Cohen, "Incentives build robustness in BitTorrent", In: Proc. of the 1st Workshop on Economics of Peer-to-Peer systems, June 2003.

[85]. J., Garbacki, P., Epema, D., Sips, H., 2005. The Bittorrent P2P filesharing system: measurements and analysis. Lecture Notes in Computer Science 3640/2005, 205–216.

[86]. Golle, P., Leyton-Brown, K., Mironov, L., Lillibridge, M., 2001. Incentives for sharing in peer-to-peer networks. Lecture Notes in Computer Science 2232/2001, 75–87.*

[87]. Liu, Z., Shen, Y., Panwar, S.S., Ross, Keith W., Wang, Y., P2P video live streaming with MDC: providing incentives for redistribution. In: IEEE International Conference on Multimedia and Expo, Beijing, China, 2007.

[88]. T. Silverston, O. Fourmaux, , J. Crowcroft, Towards an incentive mechanism for peer-to-peer multimedia live streaming systems. In: 8th International Conference on Peer-to-Peer Computing, Aachen, Germany, September 2008.

[89]. Lin, C.-S., Cheng, Y.-C., "A barter-based incentive mechanism for peer-to-peer media streaming", In: 13th IEEE International Symposium on Consumer Electronics, Kyoto, Japan, May 2009.

[90]. Y. Tang, , L. Sun, M. Zhang, S. Yang, Y. Zhong, "A novel distributed and practical incentive mechanism for peer to peer live video streaming" In: IEEE International Conference on Multimedia and Expo, Toronto, Canada, July 2006.

[91]. I. Kofler, C. Timmerer, H. Hellwagner and T. Ahmed, Towards MPEG-21-Based Cross-Layer Multimedia Content Adaptation, Proceedings of the Second International Workshop on Semantic Media Adaptation and Personalization table of content, (2007) p.p. 3-8, ISBN:0-7695-3040-0

[92]. I. Djama, T. Ahmed, A. Nafaa and R. Boutaba: Meet in the middle cross-layer adaptation for audiovisual content delivery. IEEE Trans Multimed 10, 105–120 (2008)

[93]. I. Kofler, J. Seidl, C. Timmerer, H. Hellwagner, I. Djama, and T. Ahmed, "Using MPEG-21 for cross-layer multimedia content adaptation", in Journal on Signal, Image and Video Processing (SIVP), Springer, Volume 2, Number 4, pp. 355-370, December 2008.

[94]. T. Ahmed and M. Mushtaq, "P2P Object-based adaptivE Multimedia Streaming (POEMS)" in Springer Journal of Network and Systems Management, Special Issue on Peer-to-Peer Technologies in Network and Service Managemen, Vol. 15, Number 3, pp. 289-310(22), September 2007.

[95]. T. Ahmed, I. Djama "Delivering Audiovisual Content with MPEG-21-Enabled Cross-Layer QoS Adaptation" in Packet Video 2006 published Springer-Verlag GmbH part of Journal of Zhejiang Univ SCIENCE A (paper accepted from IEEE Packet Video PV), Volume 7, Number 5, pp. 784-793. April 2006.

[96]. T. Ahmed, A. Mehaoua, R. Boutaba, and Y. Iraqi, "Adaptive Packet Video Streaming Over IP Networks: A Cross-layer Approach" in IEEE Journal on Selected Areas in Communications (J-SAC), Special issue on Intelligent Services and Applications in Next Generation Networks, Vol. 23, NO. 2, pp. 385 - 401 Feb. 2005

[97]. G. Tan, S.A. Jarvis, A payment-based incentive and service differentiation mechanism for peer-to-peer streaming broadcast. In: 14th IEEE International Workshop on Quality of Service, New Haven, CT, June 2006.

[98]. N. Semret, R. Liao, A.T. Campbell and A.A. Lazar, Pricing, provisioning and peering: dynamic markets for differentiated internet services and implications for network interconnections. IEEE Journal on Selected Areas in Communications 18 (12), 2499–2513. 2000.

[99]. A. Habib, J. Chuang, , Service differentiated peer selection: an incentive mechanism for peer-to-peer media streaming. IEEE Transactions on Multimedia 8 (3), 610–621. 2006.

[100].B. Yang, H. Garcia-Molina, PPay: micropayments for peer-to-peer systems. In: 10th ACM Conference on Computer and Communications Security, NY, USA, October 2003.

[101].K. Wei, A.J. Smith, Y. Chen, B. Vo, WhoPay: a scalable and anonymous payment system for peer-to-peer environments. In: 26th IEEE International Conference on Distributed Computing System, Lisboa, Portugal, July 2006.

[102].S. Ye, F. Makedon, , Collaboration-aware peer-to-peer media streaming. In: 12th Annual ACM International Conference on Multimedia, NY, USA, October 2004.

[103].N. Shi, Q. Dai, A novel incentive mechanism improving peer-to-peer on demand streaming. In: International Conference on Communications, Circuits and Systems Proceedings, Guilin, China, June 2006.

[104].C. Liang, Z. Fu, Y. Liu, C.W. Wu, iPASS: incentivized peer-assisted system for asynchronous streaming. In: 28th IEEE International Conference on Computer Communications, Rio de Janeiro, Brazil, April 2009..

[105].Z. Liu, H. Hu, Y. Liu, K.W. Ross, Y. Wang, M. Mobius, P2P trading in social networks: the value of staying connected. In: 29th IEEE International Conference on Computer Communications, San Diego, CA, March 2010.

[106].Y.h. Chu, J. Chuang, H. Zhang, A case for taxation in peer-to-peer streaming broadcast. In: ACM SIGCOMM workshop on Practice and theory of incentives in networked systems, 2004.

[107].H. Hu, Y. Guo, Y. Liu, Peer-to-peer streaming of layered video: efficiency, fairness and incentive. Circuits and Systems for Video Technology, 2011.

[108].B. Chun, Y. Fu, A. Vahdat, Bootstrapping a distributed computational economy with peer-to-peer bartering. In: Proceedings of the 1st Workshop on Economics of Peer-to-Peer Systems. 2003.

[109].F. Loukos, H. Karatza, SPECT: a system for peer-to-peer economic transactions. In: 2011 IEEE Symposium on Computers and Communications (ISCC), 2011.

[110].A. Hutter, P. Amon, G. Panis, E. Delfosse, M. Ransburg, H. Hellwagner, Automatic adaptation of streaming multimedia content in a dynamic and distributed environment, IEEE International Conference on Image Processing (ICIP 2005), vol. 3, pp. 716-719, September 2005.

[111].T. Ahmed, A. Mehaoua, R. Boutaba, Y. Iraqi, "Adaptive packet video streaming over IP networks: a cross-layer approach", Selected Areas in Communications, IEEE Journal on 23 (2), 385-401

[112].S. F. Chang and A. Vetro, Video adaptation: Concepts, technologies and open issues, Proc. IEEE-Special Issue on Advances in Video Coding and Delivery, vol. 93, no. 1, pp. 148–158, Jan. 2005.

[113].I. Kofler, J. Seidl, C. Timmerer, H. Hellwagner, I. Djama, T. Ahmed,: Using MPEG-21 for cross-layer multimedia content adaptation. Signal Image Video Process. 2(4), 355–370, 2008.

[114].T. Ahmed, A. Nafaa and A. Mehaoua "An Object-Based MPEG-4 Multimedia Content Classification Model for IP QoS Differentiation" IEEE Symposium on Computers and Communications - ISCC'2003. July 2003.

[115].T. Ahmed, G. Buridant, A. Mehaoua, Encapsulation and marking of MPEG-4 video over IP differentiated services, ISCC'01 (2001) 346–352.J.

[116].T. Ahmed, G. Buridant, A. Mehaoua, Implementing MPEG-4 video on demand over IP differentiated services, GLOBECOM'01 4 (2001) 2489–2493.

[117].S. Chakareski, and B. Girod, Layered coding vs. multiple descriptions for video streaming over multiple paths, in Multimedia Systems, Springer, online journal publication: Digital Object Identifier (DOI) 10.1007/s00530-004-0162-3, January 2005.

[118].H. Schwarz, D. Marpe, and T. Wiegand, SNR-Scalable Extension of H.264/AVC, in proceedings of ICIP2004, Singapore, 2004.

[119].H. Schwarz, D. Marpe, and T. Wiegand, Overview of the Scalable H.264/MPEG4-AVC Extension, in Proc. IEEE International Conference on Image Processing, pp. 161-164, Atlanta, USA, Oct. 2006.

[120].H. Sun, A. Vetro, and J. Xin, An overview of scalable video streaming, Wireless Communications and Mobile Computing, vol. 7, no. 2, pp. 159-172, Feb. 2007.

[121].T. C. Thang, Y. S. Kim, Y. M. Ro, J. W. Kang, and J.-G. Kim, "SVC bistream adaptation in MPEG-21 multimedia framework," presented at the Int. Packet VideoWorkshop, Apr. 2006.

[122].D. Wu, T. Hou and Y.-Q. Zhang, Scalable Video Coding and Transport over Broadband Wireless Networks, Proceedings of the IEEE, Sept. 2000.

[123].M. Mushtaq, T. Ahmed, Adaptive packet video streaming over P2P networks using active measurements, in: IEEE Symposium on Computers and Communications (ISCC), June 2006, pp. 423–428.

[124].M. Mushtaq and T. Ahmed, "Hybrid Overlay Networks Management for Real-Time Multimedia Streaming over P2P Networks", in proc. MMNS 2007, LNCS volume 4787, pp. 1–13, 2007.

[125].M. Mushtaq and T. Ahmed. Smooth Video Delivery for SVC Based Media Streaming Over P2P Networks. Consumer Communications and Networking Conference, 2008. CCNC 2008. 5th IEEE, pages 447{451, 2008.

[126].A. Kourtis, G. Kormentzas, C. Skianis, G. Xilouris, D. Negru, A. Mehaoua, Toufik Ahmed, E. Borcoci, H. Asgari, S. Eccles, E. LeDoeuf "Provisioning of End to End QoS in Diverse Environments: The ENTHRONE View" in WSEAS Transactions on Communication, pp. 626-631, Issue 2, Volume 3, April 2004, ISSN 1109-2742.

[127].M. Mushtaq and T. Ahmed, "QoS Provisioning for Video Streaming Over SP-Driven P2P Networks Using Admission Control", Proceedings of 6th IEEE Consumer Communications & Networking Conference (IEEE CCNC) - 2009, January 2009

[128].Lu, et al., "Research and Design on Peer Selection Strategy of P2P Streaming". In Proc. WiCom 2009, pp.1-4, Sept. 2009

[129].Y. Guo, C. Liang, and Y. Liu, "AQCS: Adaptive Queue-based Chunk Scheduling for P2P Live Streaming,". In Proceedings of IFIP Networking, 2008.

[130].R. K. Rajendran and D. Rubenstein, "Optimizing the Quality of Scalable Video Streams on P2P Networks". Computer Networks: The International Journal of Computer and Telecommunications Networking, vol. 50, no. 15, pp. 2641–2658, Oct. 2006.

[131].P. Baccichet, T. Schierl, T. Wieg, and B. Girod, "Low-delay Peer-to-Peer Streaming using Scalable Video Coding". In Proc. of International Conference on Packet Video, 2007.

[132].L. Zhou, X. Wang, Y. Li, B. Zheng, and B. Geller, "Optimal Scheduling for Multiple Description Video Streams in Wireless Multihop Networks,". IEEE Communications Letters, vol. 13, no. 7, pp. 534–536, Jul 2009.

[133].I. Lee, "Reliability Analysis of a Multiview Multi-Description Video Streaming System", in proceeding of 7th International Conference on Autonomic & Trusted Computing (UIC/ATC), 2010.

[134].M. Zhang, Y. Xiong, Q. Zhang, and S. Yang, "Optimizing the throughput of data-driven peer-to-peer streaming". IEEE Transactions on Parallel and Distributed Systems, vol.20, no.1, 2009.

[135].T. Szkaliczki, M. Eberhard, H. Hellwagner, L. Szobonya, "Piece Selection Algorithm for Layered Video Streaming in P2P Networks". Electronic Notes in Discrete Mathematics, Volume 36, 1 August 2010.

[136].D.W. Pentico, Assignment problems: A golden anniversary survey, European Journal of Operational Research 176 (774–793), 2007.

[137].R. M. Nauss, "Solving the generalized assignment problem : An optimizing and heuristic approach". INFORMS J. Comput.15(3) 249–266, 2003.

[138].Z.W. Geem, J.-H. Kim and G.V. Loganathan, "A new heuristic optimization algorithm: harmony search," Simulation 76 (2), pp. 60–68, 2001.

[139].K. Lee and Z. Geem, "A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice", Computer Methods in Applied Mechanics and Engineering, 2005.

[140].Harold W Kuhn, "The Hungarian method for the assignment problem", Naval Research Logistics Quarterly, 2:83-97, 1955.

[141].SIMULINK, http://www.mathworks.com/products/simulink/

[142].Hsu Cheng-Hsin and M. Hefeeda, "Optimal Coding of Multilayer and Multiversion Video Streams", IEEE Transactions on Multimedia, vol.10, no. 1, pp. 121-131, Jan. 2008.

[143].W. Wang, and B. Li, "Market-Based Self-Optimization for Autonomic Service Overlay Networks," IEEE J. on Selected Areas in Communications, Vol. 23, No. 12, pp. 2320-2332, december 2005.

[144].Y. Cui, Y. Xue, and K. Nahrstedt, "Optimal Resource Allocation in Overlay Multicast," IEEE Transactions on Parallel and Distributed Systems, Vol. 17, No. 8, pp. 808-823, august 2006.

[145].E. Takahashi, Y. Tanaka, "Bandwidth Allocation by Using Generalized Vickrey Auction", Asia Pacific Symposium on Information and Telecommunication Technologies, 2001.

[146]. B. N. Chun, P. Buonadonna, A. AuYoung, C. Ng, D. C. Parkes, J. Shneidman, A. C. Snoeren, and A. Vahdat. Mirage: A microeconomic resource allocation system for sensornet testbeds. In Proceedings of 2nd IEEE Workshop on Embedded Networked Sensors (EmNetsII), 2005.

[147].C. da Silva, A. P., E. Leonardi, M. Mellia, and M. MEO, "Exploiting Heterogeneity in P2P Video Streaming", IEEE Transactions on Computers, vol. 60: IEEE, pp. 667–679, May, 2011.

[148].F. Liu, B. Li, L. Zhong, Ba. Li, H. Jin, X. Liao, "Flash Crowd in P2P Live Streaming Systems: Fundamental Characteristics and Design Implications," IEEE Transactions on Parallel and Distributed Systems, vol. 23, no. 7, pp. 1227-1239, July 2012, doi:10.1109/TPDS.2011.

[149].R. Kumar, Y. Liu, and K. Ross. Stochastic Fluid theory for P2P streaming systems. In Proceedings of IEEE INFOCOM, pages 919-927, Anchorage, AK, May 2007.

[150].A. da Silva, E. Leonardi, M. Mellia and M. Meo, "Chunk distribution in mesh-based large-scale P2P streaming systems: a fluid approach". IEEE Trans Parallel Distrib Syst. 22(3), 451–463, 2011.

[151].M.P. Manzillo, L. Ciminiera, G. Marchetto and F. Risso, "CLOSER: A Collaborative Locality-Aware Overlay SERvice". IEEE Trans Parallel Distrib Syst. 23 (6), 1030- 1037, 2012.

[152].M. Zhang, Q. Zhang, and S. Yang. Understanding the power of pull-based streaming protocol: Can we do better? IEEE Journal on Selected Areas in Communications, 25(8):167(8)1694, 2007.

[153].X. Hei, C. Liang, J. Liang, Y. Liu, and K. W. Ross. "A measurement study of a large-scale P2P IPTV system," IEEE Transactions on Multimedia, 9(8):1672-1687, December 2007.

[154].B. Li, Y. Qu, Y. Keung, S. Xie, C. Lin, J. Liu, and X. Zhang, "Inside the new Coolstreaming: principles, measurements and performance implications," In Proceedings of IEEE INFOCOM, Phoenix, AZ, April 2008.

[155].C.Wu, B.Li and Z.Li, "Dynamic Bandwidth Auctions in Multi-overlay P2P streaming with Network Coding", IEEE Trans. on Parallel and Distributed Systems, vol. 19, no. 6, pp. 806–820, June 2008.

[156].C. Liang, M. Zhao, and Y. Liu, "Optimal Resource Allocation in Multi-Source Multi-Swarm P2P Video Conferencing Swarms," accepted for publication in IEEE/ACM Trans. on Networking, 2011.

[157].S. Medjiah, T. Ahmed, E. Mykoniati, and D. Griffin, "Scalable video streaming over P2P networks: A matter of harmony?", 16th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), 2011.

[158].H. Hu, Y. Guo and Y. Liu, "Peer-to-Peer Streaming of Layered Video: Efficiency, Fairness and Incentive," Circuits and Systems for Video Technology, IEEE Transactions on , vol.21, no.8, pp.1013-1026, Aug, 2011.

[159].Z. Wang, C. Wuy, L. Sun, and S. Yang,, "Strategies of Collaboration in Multi-Swarm Peer-to-Peer Content Distribution", Elsevier Journal of Network and Computer, 2012.

[160].S. Parsons, J. A. Rodriguez-Aguilar, M. Klein, "Auctions and bidding: A guide for computer scientists", ACM Computing Surveys (CSUR), Volume 43 Issue 2, January 2011.

[161].L. R. White, "Effective Marginal Costing: Know Your Resource Needs", Resource Consumption Accounting Institut, March 2010.

[162].R. Agarwal, M. Meehan and D. O'Regan, "Fixed point theory and applications", Cambridge university press, 2001.

[163].A. Bradai and T. Ahmed, "On the Optimal Scheduling in Pull-based Real-Time P2P Streaming Systems: Layered and Non-Layered Streaming". In proceeding of ICC'12, Ottawa, 2012.

[164].D. S and K. W. Ross, "Optimal Streaming of layered video". In Proc of IEEE INFOCOM, April 2000.

[165]. M. Zhang, Y.Q. Xiong, Q. Zhang, and S.Q. Yang, "On the Optimal Scheduling for Media Streaming in Data-driven Overlay Networks". In Proc. IEEE GLOBECOM 2006.

[166]. L. Dai, Y. Cui, and Yuan. Xue, “Maximizing Throughput in Layered Peer-to-peer streaming”. In Proc of IEEE ICC, 2007.

[167]. Hossain. T, Yi Cui, Yuan. Xue,. “On the Optimality of Layered Video Streaming Rate in a P2P Mesh Network”. In Proc of ICCCN, 2009.

[168]. A. T. Nguyen, B. Li, and F. Eliassen. “Chameleon: Adaptive Peer-to-Peer Streaming with Network Coding”. In Proc of IEEE INFOCOM’10, 2010.

[169].A. T. Nguyen, B. Li, and F. Eliassen. “Chameleon: Adaptive Peer-to-Peer Streaming with Network Coding”. In Proc of IEEE INFOCOM’10, 2010.

[170].H. Hu, Y. Guo, Y. Liu. “Mesh-based peer-to-peer layered video streaming with taxation”. In Proc of the 20th International workshop on Network and Operating Systems Support for Digital Audio & Video (NOSSDAV 2010), 2010.

[171].A.T. Nguyen, B. Li,F. Eliassen,” Quality- and Context-aware Neighbor Selection for Layered Peer-to-Peer Streaming. In Proc of IEEE ICC 2010.

[172].M. Zink, O. Kuenzel, J. Schmitt, and R. Steinmetz. “Subjective Impression of Variations in Layer Encoded Videos”. In International Workshop on Quality of Service, 2003.

[173].http://www.envision-project.org/