Analysis, modelling, and optimisation of a peer-to-peer-based video-on-demand streaming system

Thibaud Rohmer

UNIVERSITE PARIS-EST

ECOLE DOCTORALE MATHEMATIQUES ET STIC
(MSTIC, E.D. 532)

THESE DE DOCTORAT
EN INFORMATIQUE

par

Thibaud ROHMER

Sujet de la thèse :

Analysis, Modelling, and Optimisation of a Peer-to-Peer-based Video-on-Demand Streaming System

---

soutenue le 10/09/2014

Jury :

Yacine Amirat	Professeur des universités	Université Paris-Est Créteil	Président
Farouk Yalaoui	Professeur des universités	Université de Technologie de Troyes	Rapporteur
El-Gazali Talbi	Professeur des universités	Université de Lille 1	Rapporteur
Amir Nakib	Maitre de Conférences	Université Paris-Est Créteil	Examinateur
Liam Murphy	Professeur des universités	University College Dublin	Co-directeur de thèse
Eric Petit	Professeur des universités	Université Paris-Est Créteil	Directeur de thèse

Abstract

Due to its growing success with consumers as a preferred way to access video content, VoD (Video On Demand) services streaming is gaining unprecedented interest from the IPTV and the Internet video streaming industries alike. Unlike the rigid linear video services programming that is the norm in broadcast TV, the on-demand content access provides a great flexibility allowing end-users to browse and consume video content in a non-scheduled way. Besides providing a VoD offering to meet evolving end-users expectations for flexibility and differed-time access, it is today very important for a service provider to offer a very large content library (popular and niche content) to capture the broadest audience possible. Targeting a very large audience with a large video content library puts a very heavy burden on service providers with scalability being the chief design issue.

This thesis takes the view that P2P streaming systems can be an appropriate candidate to meet the challenges of professional VOD system in terms of large content library, large number of customers, and cost imperatives. P2P VOD streaming essentially relies on the many-to-one communication approach where the VOD session is provisioned through a multisource streaming session. In other words, the receiving peer receives multiple complementary video sub-streams from different contributing peers.

A first contribution of the thesis focuses on the content injection strategy and how the different content fragments should be dispatched in the network to achieve the highest performance at VoD services provisioning epoch. In fact the way the content library is fragmented and spread in the network plays a critical role in ***(i)*** determining the content availability level, and ***(ii)*** improving the network capacity in meeting the demand for content by maximizing the number of VOD sessions that can be delivered.

We demonstrate that the random injection strategy is not appropriate to maximize the number of simultaneous VoD streaming sessions in the network. After gaining a better understanding of the factors driving P2P-based VoD streaming systems, we provide guidelines to better operate such system to achieve different performance objectives and/or fit specific network configurations. Further, we propose a new content dispatching strategy that maximizes the number of served VoD

sessions. This approach relies on the content popularity to effectively distribute the content among peers in the network in a way to facilitate streaming load balancing, and thus achieve better system resources utilization.

Our second scientific contribution consists of addressing the issue of resources allocation in P2P streaming systems. First, we introduce three basic resource allocation models that use a single-criterion algorithm (Available Uplink, Popularity Score, and Critical Score) to select contributing STBs and satisfy an incoming VoD request. We individually evaluate the performance of every single-criterion resource allocation strategy, and highlight its strengths and weaknesses in dealing with different situations. The learning from this analysis has been employed to develop a dynamic resource allocation algorithms for P2P streaming systems.

Last but certainly not least, we addressed on the problem of maximizing the P2P streaming system capacity by using Bayesian approach to dynamically alternate between different resource allocation strategies. This switching between different resource allocation strategies is guided by a dynamic statistical analysis of performances. A key contribution resides in effectively combining different, and potentially conflicting, performance objectives when deciding on which resource allocation strategy to use for the given time interval. Moreover, a VoD service operator can specify any performance objectives (decision criteria) that meet its requirement, and our approach will adapt in order to maximize them. We show that this is an efficient way to greatly improve the performances of a P2P streaming system, such as minimizing the VoD requests rejection rate, when facing constantly changing content demand patterns.

As part of our effort to develop algorithms that can be effectively deployed by broadband operators, we developed a full-scale emulator for P2P streaming that can support up to 10,000 peers. The P2P streaming emulator includes a VOD request generators, along with a SuperNode (with backend DB) that tracks and allocate network resources.

Dedicated to my parents, my grandmother, and to my brother, Loïc.

Acknowledgements

I would like to first thank my supervisors, Dr. Hamid Nafaa and Dr. Amir Nakib, for their support and encouragement over the last few years. I am very grateful for the many discussions, advices and opportunities they have provided. I would also like to thank Prof. Liam Murphy and Prof. Eric Petit for being very involved and giving me many encouragements during the discussions and presentations.

I wish to thank my colleagues and friends in Ireland, for making this such an entertaining experience. Too many names come to mind, but I would like to give my special thanks to Sandra, who motivated me into doing this PhD, and to Laurent, for always being where the pints were.

I also wish to extend my thanks and gratitude to all my friends and colleagues in France. Matheo, my flatmate, for supporting me every day, and resisting the urge to jump out of a window. Marion and Florent, for helping Matheo not to jump out of a window. Elie and Isabelle, of course, for getting me fired up, and for being awesome. Riad, for being at his desk longer than anybody in the world could ever be. And finally, of course, all of the N'importe Quoi team, for showing up and supporting me.

Finally, I would like to thank my parents, Claude and Frédérique, my grandmother Ginette, and my brother Loïc, for their love, support, and encouragement.

A special thanks goes to Florent, Michael, Nicolas, Isabelle, and Matheo. Obtaining my PhD would not have been possible without their ability to boost me, cheer me up, make me laugh, and, mainly, put me back on tracks. I will be forever grateful for their confidence in me, and the times we have spent together.

Contents

Analysis of a P2P-VoD System

P2P-VoD System

Static Resource Allocation

Dynamic Resource Allocation

Simulation Platform

Content Injection

Resource Allocation

List of figures

List of publications

International Journals

- J. Muñoz-Gea,, A.Nafaa, J. Sanahuja, and T. Rohmer. "*Design and analysis of a peer-assisted VoD provisioning system for managed networks.*" ***Springer's Multimedia Tools and Applications Journal*** (2012): 1-36.

- T. Rohmer, A. Nakib, A. Nafaa," *A learning based Resource allocation approach for P2P Streaming systems*", To appear in ***IEEE Network Magazine 2014***.

- T. Rohmer, A. Nakib, A. Nafaa, *"Priori knowledge guided approach for optimal peer selection in P2P VoD systems"*, 2nd round review in ***IEEE Transactions on Network and Service Management.***

International Conferences

- Rohmer, T. Nafaa, A. Nakib, A., "*On resource allocation strategies in managed peer-assisted VoD streaming systems*," **17TH IEEE Computers and Communications (ISCC'12)**, p.551,558, 1-4 July 2012

- Rohmer, T. Nakib, A. Nafaa, A., "*Optimal Peer Selection Strategy in P2P-VoD Systems Using Dynamic Evolution Strategy,*" ***27TH*** **IEEE International Parallel and Distributed Processing Symposium (IEEE IPDPS'13)**, p.474,481, 20-24 May 2013

- T. Rohmer, A. Nakib, J. Lepagnot, A. Nafaa, "*Adaptive Peer Selection Strategy in P2P-VoD systems based on Dynamic Metaheuristic*", **19TH IEEE International Conference on Parallel and Distributed Systems (ICPADS'13)**, Dec. 15-18, 2013, Seoul, Korea.

- Rohmer, T. Nakib, A., Nafaa, A.,"*Optimisation Dynamique de l'allocation de ressources appliquée à la Vidéo à la Demande,*", ***ROADEF, 2013 (international French conference)***, 13-15 February 2013

I – INTRODUCTION

Section 1 – Introduction

Faced with ever growing video quality and bandwidth requirements, Video-on-Demand (VoD) service providers are looking for ways to reduce their cost base through the design of scalable VoD systems. Peer-to-peer (P2P) networks have increasingly become a dependable scalable alternative to traditional point-to-point communications. As a consequence, many research works have shown interests in combining VoD systems with P2P communication paradigm to improve scalability and reduce operation costs – traditionally driven by bandwidth cost.
The Internet streaming industry is today powered by commoditized streaming resources offered by Content Delivery Networks (CDNs), which is driving the service cost down and favoring the raise of professional, QoS-enabled Internet video streaming services such as Hulu, Netflix, Amazon VOD, etc. The obvious economies of scale offered by CDNs backend streaming capacities are mainly behind the success of Internet streaming services.
In this context, VOD streaming systems operated by traditional broadband operators need to be significantly overhauled to achieve higher scalability, cut the per-service cost, and ultimately be a viable alternative to Internet streaming services. P2P-based VOD streaming services have the potential to deliver great benefits in this sense.
Peer-assisted VOD streaming systems in managed networks builds on the tremendous resources (bandwidth, storage space, and power) available at end-systems to deliver the VOD service at a fraction of the cost. In such VOD systems a peer that requests access to a given VOD content will receive the service in form of a multi-source streaming session from different other STBs in the network.

P2P-based streaming in the context of a broadband network consists of fragmenting the different video contents into complementary fragments and then spreading them in the different STBs. This way a VoD service requested by a given requesting peer can be served by other contributing peers in the network, provided that they have the necessary fragments and uplink capacity to do so. Both content and bandwidth availability are two important dimensions that should be managed by P2P streaming systems.
In our streaming architecture, the actual VoD streaming service is provisioned through a multipoint-to-point streaming session where

multiple complementary video sub-streams are streamed from a set of contributing peers to the receiving peer.
Clearly, the contents offered for streaming have to be fragmented and spread in the network beforehand. Content Injection is in fact a key part of the system that necessitates optimization. Depending on how the content is duplicated and spread in the network, the system can more or less better satisfy the demand for content. In this thesis, we introduce and evaluate multiple strategies, and present the advantages of each one of them.

In a second scientific contribution of this thesis, we address the problem of resource allocation (RA) in P2P-VoD systems. Indeed, as the central node receives VoD requests and fulfills them by allocating contributing peers to stream the VoD session, uplink resources of peers are committed for the duration of the title (resp. movie). Considering that each peer would typically store a wide range of content parts, the resource allocation decision becomes critical for maximizing the network utilization, and minimizing VoD requests rejection rates. In fact, as the peers' uplink capacity gets saturated, all content parts contained in these peers become unavailable to satisfy new VoD requests. Ideally, the resource allocation algorithm should carefully weigh which peer to commit so as to leave necessary resources available to satisfy the most probable future VoD requests.
We present several single-criteria resource allocation strategies, which can be alternatively used to deal with a wider range of network conditions in terms of (i) popularity trend shifts, (ii) a changing level of network saturations, (iii) specific content scarcity, etc. In order to decide which RA strategy to choose as the system is operating, we use two simple performance objectives: maximize VoD request success rate and balance the streaming load among active peers. A key scientific contribution here resides in using Bayesian fusion approach to combine the two performance objectives and evaluate the different RA strategies in a much more granular and effective way. Such a statistical fusion of performance objectives allows to considerably outperforming a linear combination of objectives.

Another key contribution of our work resides in developing a full-scale emulator of a P2P-based VoD streaming system that handles actual VoD requests and relies on a full-scale peers database to track available resources. Our system is a credible proof-of-concept for the implementation of P2P-based VoD streaming systems that can handle hundred thousands of peers while using complex Bayesian algorithms to optimally allocate system resources.

1. Problem Statement

Many P2P systems have already been studied in the literature for video streaming. Therefore, there currently exists a multitude of approaches for content streaming. Most of these approaches are focused on live streaming. Little work has been done for VoD streaming. In such approach, the library of contents is usually larger (10,000) than in a live streaming context (usually around 10 contents available at a given time). Therefore, it is necessary to design a solution that can support large libraries of contents.
As revealed above, it is clear that while the video distribution market is developing quite rapidly, and the current solutions do not meet the challenges of high quality, real-time distribution of legal video content. New solutions are necessary to solve these problems. Particularly, a broadband operator–focused solution is a very promising approach that conciliates both (i) the QoS imperative by operating in a managed network and (ii) the scalability of peer-to-peer approach.

The focus of this work is to address two critical aspects related to the scalable distribution of VoD content in a managed broadband network.

First, this work focuses on investigating different content fragmentation and dispatching algorithms. In fact, the way different titles in a content library are spread in the peers play a major role in defining the availability of the content and meeting the varying demand from users. If a small group of peers holds all of the popular contents, it becomes more likely that these popular contents will rapidly become unavailable as the small set of peers get their uplink saturated with an increasing demand. On the other hand, spreading the content as much as possible require making compromises on the size of a content fragment and thus the number of complementary sub-streams to deliver the VOD session. Too many sub-streams would increase the chances of failure.

Second, resource allocation (RA) in peer-assisted VoD systems plays a major role in the system performance and its viability. The resource allocation here refers to the process of assigning contributing peers to stream the content requested by the receiving peer. The resource allocation strategy is another critical aspect that determines the streaming system performances. In fact, the way resources are allocated, and progressively consumed, to satisfy the arriving VoD requests is critical in defining when the network reaches saturation and starts rejecting VoD requests.

We now present the main research questions addressed in this work.

1. What are the key design elements for an efficient peer-to-peer video-on-demand streaming system? Where do the key performance trade-offs lie? And what drives the per-VoD service delivery cost? The focus here is to identify key implementation issues.
2. What are the best content dispatching approaches? And for what use cases are they individually best suited? Determining the best content dispatching algorithm requires rigorous mathematical modeling of how the content availability varies for a given popularity model.
3. What are the best resource allocation strategies? And how do they perform for a different content demand variation? How can resource allocation strategies be combined in a genetic algorithm to produce the best results against a wide range of content popularity changes? What mathematical models are best suited to capture and predict content popularity dynamics (demand)?
4. How do the dynamic aspects of the demand in VoD systems impact the efficiency of the streaming system?

2. Overview of the proposed solution

This thesis focuses on developing a fully scalable VoD streaming system. In order to provide scalability, we explore the use of the P2P communication paradigm. The P2P streaming system is managed by a central server, the Super Node, so as to control all the system resources and ensures coordination, cost-effectiveness, and QoS. Ideally, our P2P streaming system should be supplemented by a CDN as a fallback option during extremely high demand hours. This hybrid P2P-CDN network is a more reliable, scalable, and manageable system for VoD that can be deployed in broadband networks where minimum service-level agreements are required.

The first scientific contribution of this thesis consists of developing effective P2P content dispatching strategies that optimize the system performance in dealing with demand for content. The initial content injection is, indeed, what determines how the content availability in the system varies throughout peak demand cycles.
Based on our thorough investigation of the content injection issues we devised a new content injection mechanism called Popularity-Weighted Content Dispatching (PW-CD). PW-CD exploits the knowledge of individual title popularity to spread the content parts in the P2P network in a way to ensure that peers are equally popular. In other words, the expected utility of each peer is the same, which means the overall P2P network resources (peers bandwidth) will be equally consumed as the content demand increases. The key to superior performances here is to ensure that streaming load is always distributed over the peers, and peers become saturated at the same pace.

Another major scientific contribution in this thesis is related to resource allocation in the P2P streaming system. Besides the content injection, resource allocation is another key factor that determines how the P2P streaming system capacity is maximized by optimally preserving the resources of the right peers in anticipation of future content demand.
We first designed and evaluated three different resource allocation strategies: Highest Uplink First (HUF), Lowest Critical Score (LCS), Lowest Popularity Score (LPS). Further, we designed an original dynamic resource allocation framework that can switch between different basic resource allocation strategies depending on current network conditions and the most likely content popularity shifts. Our approach is called Learning-Based Resource Allocation (LB-RA) and it relies on Bayesian fusion to statistically predict future demand and assess which basic resource allocation strategy would be most suitable for the next time period. More specifically, two methods are used in LB-RA to determine the “best fit” strategy by using a combination of multiple performance metrics: Bayesian Fusion, and Evidence Theory. LB-RA is meant to be a P2P resource allocation framework that feature any

number of basic resource allocation strategies, which are developed by service providers to meet their particular needs in terms of network size, active users base, content library size, etc.

Last but not least, we developed a full scale P2P streaming emulator that can emulate a large network with up to 100,000 peers and a content library containing up to 50,000 video titles characterized by a different popularity.

3. Thesis outline

This thesis is broken down into five parts. Part I is the current introduction chapter. Part II presents the background and consists of Chapter 2 and Chapter 3. The Chapter 2 offers a comprehensive literature review discussing many aspects presented in this thesis. This related works chapter covers content streaming networks, peer-to-peer video-on-demand systems, and resource allocation strategies.
Chapter 3 presents an in-depth analysis of P2P-VoD systems. An overview of such systems is proposed and the key performance factors described. We also define multiple performance evaluation metrics that are used to evaluate the performance of P2P streaming system along dimensions that matters the most to the service provider and end-users.

Part III is dedicated the system design and architecture part, and features Chapters 4 to 7. Chapter 4 focuses on the network design, and presents the hybrid P2P-CDN architecture of our approach. It also features the video fragmentation process applied to the video contents in the system.
Chapter 5 investigates the important issue of content injection. It presents a content replication algorithm to compute the number of duplicates for each content in the network. Then, it presents several content injection approaches, including the Popularity-Weighted Content Injection, which ensures a fair dispatching of contents through the network of peers.
Chapter 6 and 7 focus on the important issue of resource allocation in P2P streaming systems. First, Chapter 6 presents multiple basic approaches to perform the task of resource allocation in P2P streaming

systems, starting with single-metric ones, and moving on to multi-criteria based resource allocation.

Chapter 7 introduces the Learning Based Resource Allocation, a dynamic approach that dynamically adapts to the content demand pattern while considering the system current conditions in terms of saturation. Two techniques are used here to drive the LB-RA adaptation: Bayesian fusion, and Evidence theory.

Part IV presents the simulation parameters and the results obtained for all approaches. It consists of Chapter 8 to 10. Chapter 8 gives an overview of our full-scale experimental platform, its advantages, and limitations.

This is followed by Chapter 9, which provides an evaluation of the Content Injection algorithms. Finally, Chapter 10 offers a comprehensive performance evaluation of the resource allocation approaches for P2P streaming system.

The final part of the thesis is the conclusion and future work part, which contains Chapter 11. Chapter 11 concludes the thesis and provides a discussion of potential future work.

II - BACKGROUND

Ch. 2 : Literature Review

In this section we discuss current Video Streaming mechanisms and the processes involved in such systems. A large number of approaches are based on Peer-to-Peer networks, but none offers a viable solution to Video-on-Demand streaming. Moreover, a lot of work has been done in the literature for peer resource allocation in P2P networks, with each approach being efficient when facing a particular demand pattern.

1. Content Streaming Networks

Multiple solutions exist, in order to stream contents. In this section, we present each of those network paradigms, and evaluate their strengths. We start by investigating server-based approaches, before presenting peer-based solutions.

1.1. Server-based approaches

The traditional solution for streaming video over the Internet relies on the client-server communication paradigm [1][2]. A client sets up a connection with a video server, and video content is then streamed to the client directly from the server (see Figure 1.a). This method presents numerous limitations, as investigated in [3]. In this approach, the maximal number of simultaneous sessions is limited by the uplink capacity of the server. Standard servers present an uplink capacity of around 20 MB/sec. A standard video stream requires more than 400 KB/sec, meaning that such approach reaches its limit with 50 simultaneous sessions (see Figure 2).

One slight variation of client-server communication model is the Content Delivery Network (CDN) [4][5]. In a CDN-based solution, the video source server first pushes video content to a set of content delivery servers placed strategically at the network edges. Instead of downloading from the video source server, a client is directed to a nearby content delivery server to download the video (see Figure 1.b).

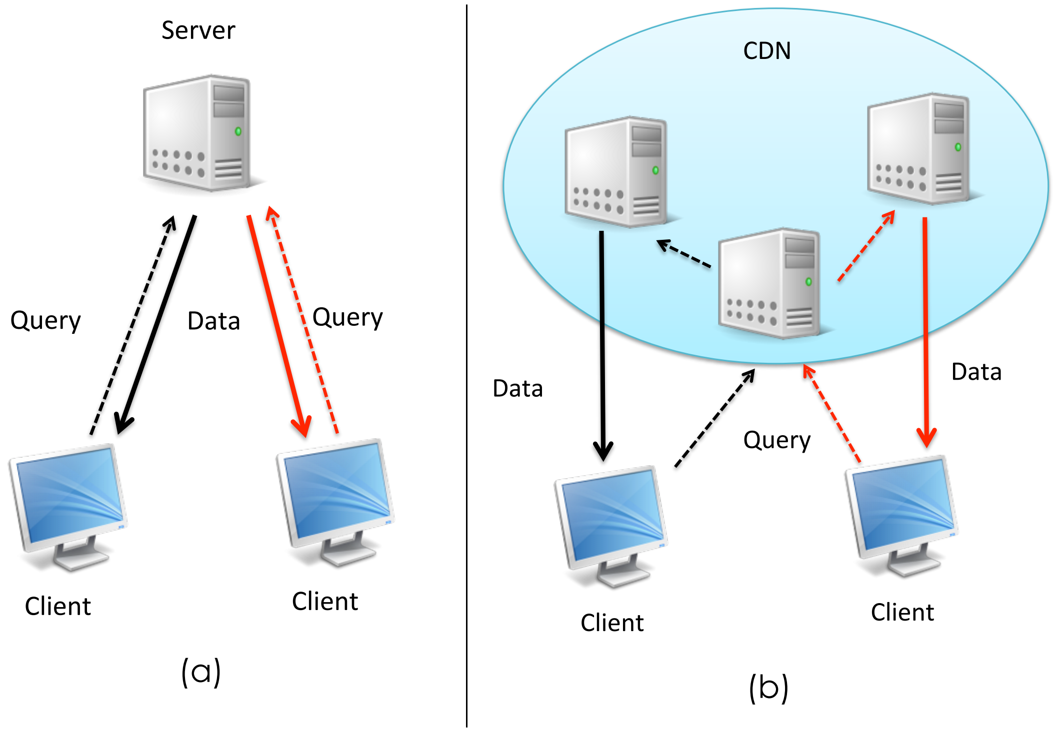
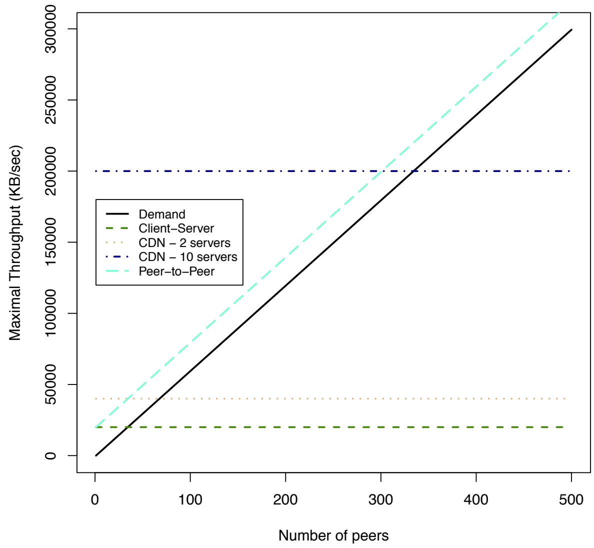


Figure 1: Server-based content distribution systems architecture. (a) Client-Server (b) CDN.

CDNs effectively reduce the video startup delays, and the traffic imposed on the network (reduced number of network hops), then serves more users. For example, YouTube employs CDN to stream video to the users, as do most professional video streaming services such as Hulu[7] and Netflix[8]. The major challenge for server based video streaming solutions, though, is its scalability[9] (see Figure 2).

Indeed, the maximal number of simultaneous sessions is directly linked to the number of servers in the CDN. In order to support more sessions, it is then necessary to increase the number of servers, thus increasing the overall cost of the system. With VoD systems growing faster every day, the bandwidth provisioning, at video source servers or in CDNs, must grow proportionally with the number of video users.

Maximal demand per peer (KB/sec)	600
Maximal peer uplink (KB/sec)	600
Maximal server uplink (KB/sec)	20.000
Cache server uplink (KB/sec)	20.000



(a) (b)

Figure 2: Maximal throughput of content distribution systems.(a) Parameters (b) Estimation.

1.2. Peer-based approaches

In a network of thousands of clients, VoD streaming systems operated by traditional broadband operators need to be significantly overhauled to achieve higher scalability, cut the per-service cost, and ultimately be a viable alternative to Internet streaming services. P2P-based VoD streaming services (see Figure 3) have the potential to deliver great benefits in this sense [10][11]. Those systems are storing data at the peers, and using the peers to send this data to each other [12][44][45]. The more clients join the network, the more internal throughput there is.

In order to provide content to all requesting users, it is necessary to be able to cope with the maximal theoretical demand of the network. Therefore, the maximal throughput capacity needs to be higher than the maximal theoretical demand.

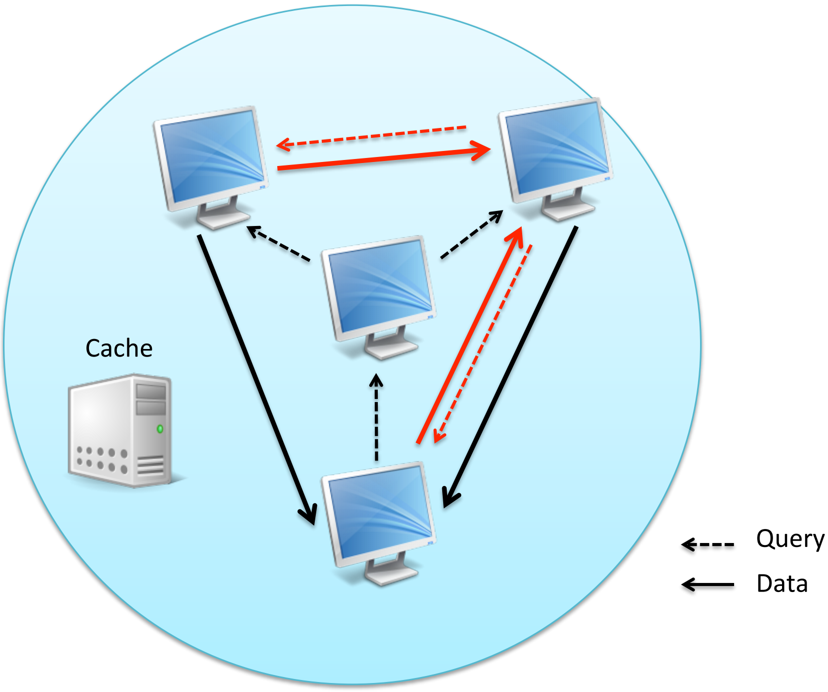


Figure 3: Content distribution system : Peer-to-Peer.

Figure 2 presents the maximal streaming capacity of each of the introduced content delivery systems, variated over the size (number of clients) of the network. It appears clearly that client-server and content delivery networks approaches can support a limited number of peers. Because the P2P approach uses the clients uplink capacity for sending data, its internal theoretical throughput increases with the number of peers.

2. Peer-to-Peer Video-on-Demand Systems

An extensive amount of research works address the scalability issues that arise when designing VoD streaming systems [13][14]. An important effort has been put into multicast-based approaches, which are more focused on the live streaming scenario [15][16]. In the following, we review the most relevant approaches.

2.1. Tracker-based approaches

2.1.1. pcVoD

In pcVoD [17], some of the peers are only providing data. In other solutions, such as BitTorrent, if peers become offline, a "no-seeds" problem can appear. In order to prevent this from happenning, in pcVoD, some peers always stay online. The pcVoD solution relies on a centralized and reliable Control Center to authenticate user and node and also to provide content-directory services (see Figure 4).

Content-providing peers with large bandwidth guarantee the streaming bitrate of new released lack-of-cache videos. Each demanding peer has storage space to cache videos. Cached video on normal peers is encrypted to prevent illegal spread. For implementation simplicity and networking efficiency, like in BitTorrent, pcVoD uses reliable Trackers to manage the connections of peers and caching. There are multiple Trackers to manage the connections and cache of different video. When a peer demands a video from the video list at the Control Center, the address of Tracker with the demanded video will be obtained from the URL of the video list. Afterwards, the peer sends a request to the tracker, which returns a list of candidate peers with the video cached. The receiving peer is then able to simultaneously retrieve the video fragments from the different candidate peers. This approach requires that the providing peers be always online, in addition to the availability of the Control Center and the Trackers.

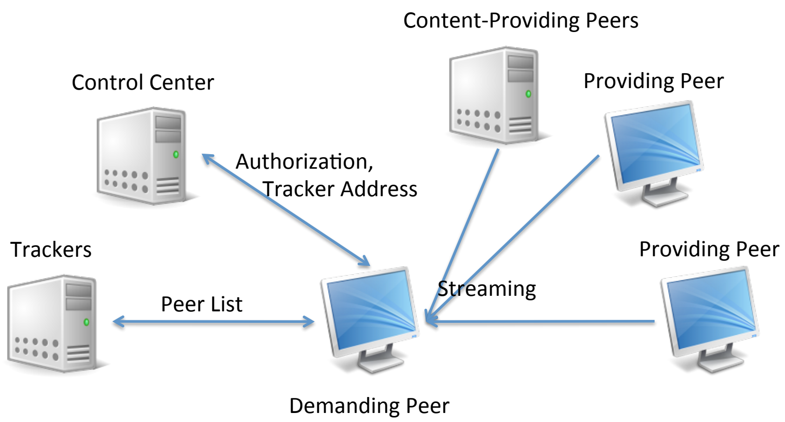


Figure 4 : pcVoD Architecture.

pcVoD introduces storage management to improve the efficiency of storage space in P2P systems. Each peer is required to share certain size of hard-drive storage cache, and an algorithm manages how to use the space. The algorithm computes the priority of the caching of a stream, based on the number of peers requesting the stream, and the number of peers that already cached this stream. This algorithm is different from traditional caching schemes, such as Least-Recently-Used, aging, and popularity-based [44].

This architecture adds some security to the whole P2P concept, by using a Control Center, and by encrypting stored data at the peers. However, it uses a simple tracker to track the peers, meaning that the requesting peer is the one that will choose which peer will be seeding data to it. Nothing manages the overall network, as each peer decides what will be downloaded from whom.

2.1.2. BitTorrent and BASS : BitTorrent Assisted Streaming System

BitTorrent [18][19][20] splits a file into many pieces and pushes the different pieces to different clients, allowing them to trade pieces amongst each other. It uses a tracker program running on a server (as opposed to a gossip protocol) to disseminate lists of peers. To govern how pieces of the file are requested and swapped amongst peers, it follows rarest-piece-first and tit-for-tat policies, respectively. In rarest-piece-first, the client requests a part based on the number of copies it sees available and chooses the least common one. In tit-for-tat, the peers seeding the most will have the highest priority when requesting content themselves.

Due to the rarest-piece first policy, BitTorrent P2P content sharing system is quite unsuitable for multimedia streaming, which requires data sequencing and timely delivery. Simply forcing BitTorrent to request pieces in-order would not be sufficient because clients would only contain subsets of each others data. Then, bittorrent's tit-for-tat policy, which aims to have requesting peers seeding parts to each-other would fail, for those peers would all already possess the same parts. In order to assess those issues, BASS [21] augments BitTorrent with an external media server (see Figure 5), with the only modification to BitTorrent being that it does not download any data prior to the current playback point.

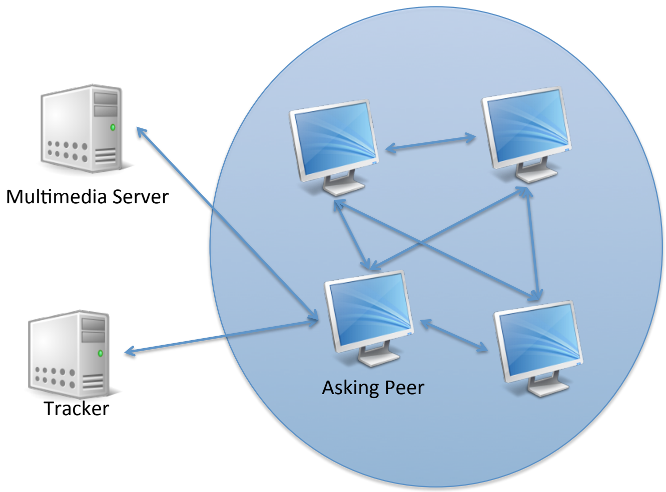


Figure 5 : BASS Architecture.

It is allowed to use the rarest-piece-first (subject to the previous condition) and tit-for-tat policies. Data from BitTorrent is held in local storage until it is needed. From the media server, BASS downloads pieces in-order, skipping over pieces that have already been downloaded by BitTorrent, or are currently in the process of being downloaded and are expected to finish before their playout deadline arrives. If the media server is altered to limit the amount of data a client is allowed to stream from it, BASS can also encourage users to participate in distribution using the tit-for-tat policy.

In this approach, whenever a peer requests a content, it is the one choosing the seeding peers (among a list of peers holding the content, generated by the Tracker), without an overview of the whole network. This is a big limitation for the system.

2.2. Tree-based approaches

2.2.1. P2Cast and P2VoD

P2Cast [22] is an architecture that relies only on unicast connections among peers (see Figure 6). The key idea of P2Cast is to have each client act as a server while it receives the video. P2Cast peers do not only receive the requested stream, but also contribute to the overall VoD service by forwarding the stream to other clients, as well as, caching and serving the initial part of the stream. This system uses a tree-based approach for streaming any given video content. In such a system, whenever a peer requests a content, two cases can occur: first, if the content is not currently requested, the peer will start a new branch, and request a direct seed from the Server. Second, if the content is already requested by another peer, the new requesting peer will attach itself to the lowest level of the branch formed by the peers requesting this content, thus asking other peers to seed the content to it, instead of overloading the server.

P2VoD [23] has a very similar approach, except that the algorithms involved in the creation of a tree differ in some aspects. Also, in P2Cast system, a video session is only open for a short period of time starting when the first client joins the video session. The result is that P2Cast can support more video sessions than P2VoD, which means greater bandwidth requirement to the server in P2Cast than in P2VoD. Several works use a network scheme very close to P2Cast and P2VoD [24][25].

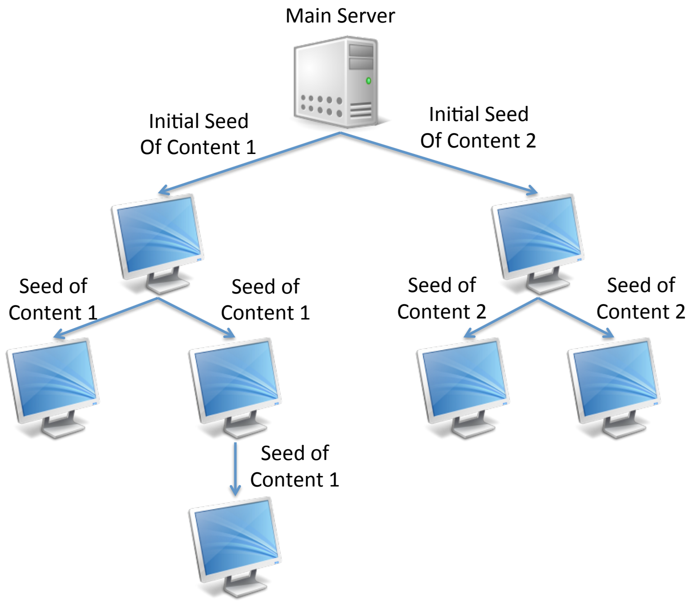


Figure 6 : Tree-based approach of P2Cast.

In the case of a lot of clients requesting the same content at the same time, this approach is very efficient in Live Streaming. On the other hand, if we consider VoD streaming, a lot of various contents can be requested at a given time from the server. This means that a lot of client will request a direct seed from the main server, overloading that server. Indeed, VoD streaming systems need to offer large video content libraries, which will put some scalability strain on the overall system: the larger the content library the less likely to have enough peers receiving the same content, and it is more difficult to construct a video content distribution tree.

2.2.2. GridCast

GridCast [26][27][28] is a system, based on OCTOPUS [29], that aims to perfect Live Streaming by using Peer-to-Peer networks. Each peer, when watching a content, stores the content, and acts as a server for any other peer requesting the same content. Another advantage of storing already watched parts of a content, is that when a client decides to go back in time, it can use the data it already stored, instead of requesting a new seed. As for the peer willing to go forth in time, the system uses anchors: some key parts of the movie are pre-loaded, and when the user tries to advance, playback is automatically moved to the closest pre-loaded anchor.

In order to select the seeding peers, and the peers you seed for, an algorithm of distance, based on relative playback positions, is used. According to this algorithm, your closest neighbors are the ones whose movie is at the same playtime as yours (see Figure 5).

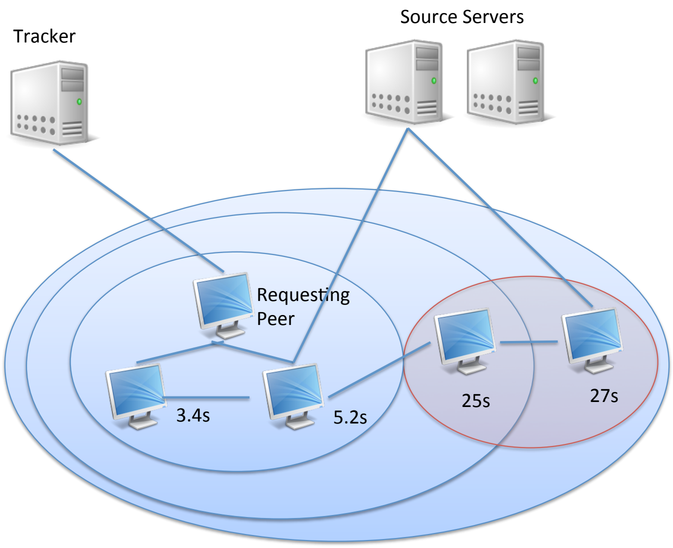


Figure 7 : GridCast Architecture

This approach tends to cluster the peers in sub-networks, and reduces a bit of the interest of having a large Peer-to-Peer network. Furthermore, while it is efficient for contents watched by many peers at the same time (like, for a Live Streaming System). For a large library of contents, the peers are less likely to be watching the same content, thus reducing the interest of GridCast.

2.3. Overlay

2.3.1. PROMISE

In PROMISE [30][31], peers are interconnected through a P2P network. For each session, multiple sending peers cooperate to serve a requesting peer. Senders are chosen based on the current network conditions and the reliability of peers to render the best quality.

PROMISE is a system that can be deployed on the top of multiple existing P2P solutions. It has the advantage of taking into account the fact that peers in a P2P network are not proper servers, and, therefore, may reduce their sending rate at any time, or even disconnect. Therefore, it tends to use multiple seeding peers for the same content.

The design of PROMISE relies on an application level P2P service called CollectCast. CollectCast chooses the sending peers and orchestrates them in order to provide the best quality for the receiver. CollectCast monitors everything in the network, therefore making it possible to predict peers streaming rates and decide from which peers to seed.

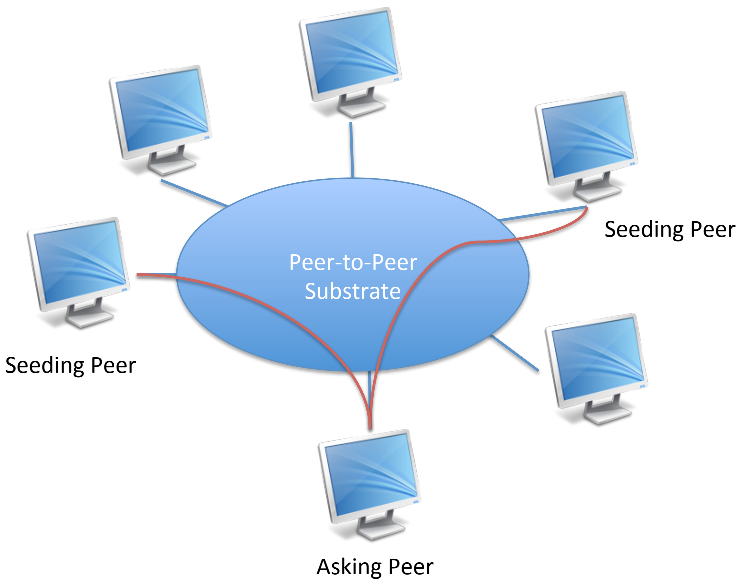


Figure 8 : PROMISE Architecture

In order to select the seeding peer, PROMISE proposes three strategies: random, end-to-end, and topology-aware. While random selects randomly a peer among those available, the others tend to predict the most efficient transfer, network-wise. End-to-end evaluates the link quality between the requesting peer, and the potential seeding peer, and selects the one with the highest link quality. The Topology-Aware strategy constructs an approximate topology, and considers the quality of each segment in the path. Furthermore, this strategy takes into account the shared segments (when multiple peers stream, they may come to use the same segments). This allows to prevent bottleneck situations during the transfer.

Set-top-boxes tend to stay connected all the time. Therefore, it would be a waste of bandwidth to download the same piece of content from multiple peers.

2.4. Hybrid CDN-P2P

2.4.1. LiveSky

LiveSky [32][33] uses three major components : (1) a Management Center, (2) Cache servers, (3) clients. The Management Center monitors the whole system, the cache servers store the contents, and the clients watch and serve contents. When a client requests a content, it is directed to its Edge Super Node (Edge SN), the closest cache server. Edge SN have multiple roles. The first role is to send data to the requesting peer. They also work as a tracker, meaning that, if any peer in the network has the requested content, they redirect the peer to the P2P overlay.

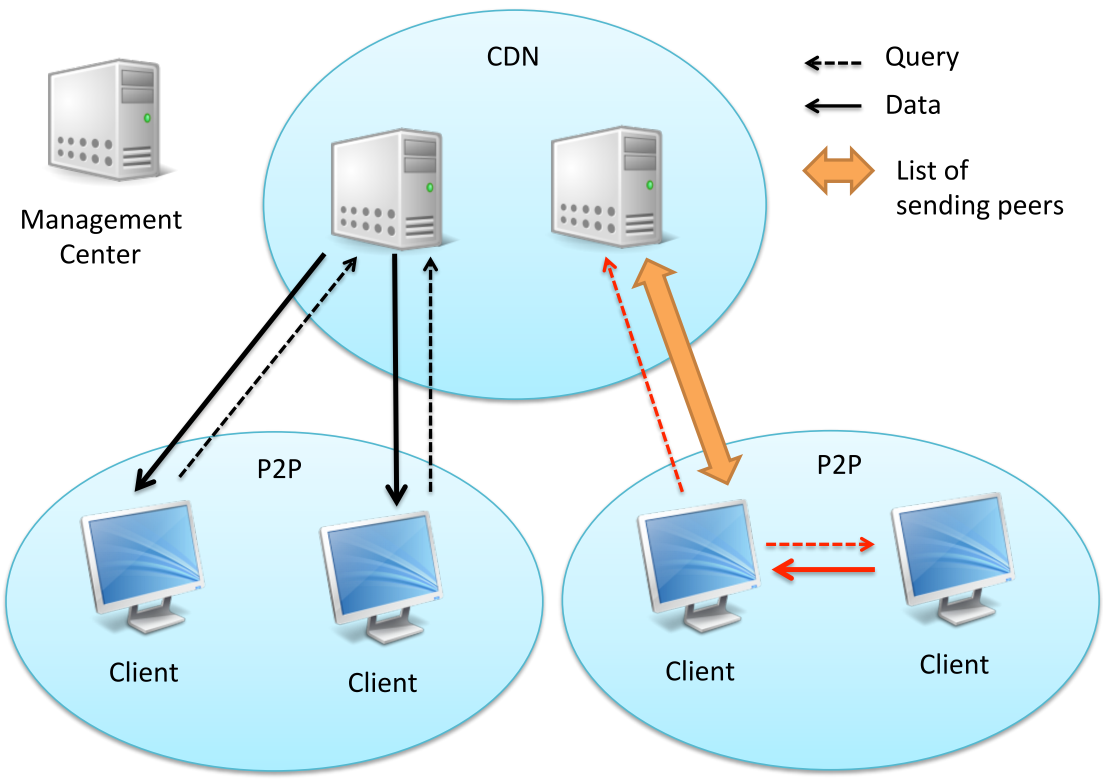


Figure 9: LiveSky architecture.

Note that it means that the P2P network is not entirely used for the transfers. Indeed, for a given peer, LiveSky only uses the ones that are close to said peer. Such system greatly reduces the interest of P2P, which comes from the huge network size, because it clusters the peers in sub-networks of smaller size.

3. Resource Allocation Strategies in P2P-VoD

The decision of allocating streaming resources to an incoming VoD request is a complex one with diverse implications on the system performances. The resource allocation algorithm can identify many contributing peers that can be used to satisfy the VoD request – these latter peers can have different available bandwidth capacities, and different stored content. The decision to use one of these peers to satisfy the current VoD request means reducing its uplink capacity for an extended period of time, and thus possibly shutting the peer's content out of the system. In this section, we investigate several strategies for peer resource allocation in VoD systems.

3.1. User-based strategies

Several strategies were proposed by Zou et al. [34]. First, they present User algorithms. Those strategies are based on data computed directly from the peers. With those strategies, each peer computes some information, it is ready to send to the requesting peer, when requested. The requesting peer then selects the best peer, based on this information. The User-based Algorithms only uses data computed by the peers individually. This information is easy to access, and doesn't require any special type of peer-to-peer network.

3.1.1. Random algorithm

This algorithm randomly chooses one serving node regardless of bandwidth and other state information. In most research works reported in the literature, this algorithm serves as a baseline for performance evaluation [47].

3.1.2. Fastest Link

This algorithm selects the peer with the largest uplink bandwidth regardless of the number of current sessions being served by that same peer. It emulates some file sharing systems where the only information available from the searching processes is the peers' speeds in terms of bandwidth capacity. In this case, it may be reasonable to choose the fastest serving peer [34].

3.1.3. Greedy algorithm

The Greedy strategy [48] selects the peer that has the maximal ratio of uplink bandwidth available over the number of sessions currently served by the peer. It compares the values of $\frac{b}{n+1}$, with b the uplink bandwidth, and n the number of seeding sessions of the peer, and selects the maximal. The rationale here is to tap the available bandwidth capacities at peers while avoiding unfairly putting a burden on peers with many downstream activities.

3.1.4. Fit algorithm

The idea of the fit algorithm [34] is to match the link speed of the requesting peer and the residual bandwidth of serving peers and save fast peers for later requests. It is directly linked to the Greedy Algorithm. However, in the Fit algorithm , if several peers are available for seeding at a speed higher than the download rate of the receiving peer, the receiving peer would select the source peer with uplink capacity that is the closest to the asking peer.

3.2. Global-based strategies

Zou et al. also introduce Global strategies [34], based on information of the overlay network. Those strategies require a global overview of the network, and therefore need a central entity gathering the information. By using a central node, it is possible to monitor the network. Then, this node is able to provide information of everything happening in the netwok. This overview the network is relevant for the following resource allocation algorithms.

3.2.1. Greedy/Fit algorithms

These algorithms are the same as the ones presented previously [34], except that they use the information gathered from to compute the bandwidth available at the peers, instead of querying each peer.

3.2.2. Batch Greedy/Batch Fit

These strategies [34] try to group the queries, in order to treat them in a "batch" mode. The central node gathers a number of queries, before starting to respond to the peers. Those queries are not treated in the ranking they were sent, but in a rank depending on the bandwidth available at the requesting peer. First, the queries from the peers with the highest downlink capacity available are envisaged, and then, the queries for the peers that have lower downlink capacity. The idea is to maximize the number of peers for streaming for the peers that have a high downlink capacity. Even though, with this method, response time will be higher, these methods tend to allocate the resources more efficiently.

3.3. Reputation System

A substantial amount of work has been done into reputation-based resource allocation strategies in peer-to-peer networks [35][36][37]. Each peer in the network is assigned a Reputation value. This value is based on multiple criteria, such as bandwidth, storage capacity, and the amount of content it already streamed.

3.3.1. Highest Reputation Strategy

The Highest Reputation Strategy [38] is the most straightforward strategy. When choosing the best peer to provide a content, we look at the reputation of all available peers, and select the one with the highest reputation. This strategy tends to assure a good streaming quality for the user. On the other hand, it may push all users to ask to the same peers, thus resulting in a faster saturation of the peers with the highest reputation.

3.3.2. Tit-for-tat Strategy

The tit-for-tat strategy was rendered popular by Bittorrent [46]. This approach favours the peers that contribute the most to the network, by allocating upload to first to the peers uploading the most.

3.3.3. Black List Strategy

This strategy identifies peers with the lowest performance, or the ones with reputation values below a certain threshold [39]. Thus, peers that consistently offer services of low quality for a certain period are excluded from the set of eligible contributing peers. Thus, this strategy improves the quality offered to the remaining peers actively participating in the P2P system.

3.3.4. Comparable Reputation Strategy

In this strategy, peers are able to request services only from peers that have reputation values close to theirs [40]. The underlying idea of this strategy is the matching of the performance level offered by a peer with the performance level provided to him. Thus, this strategy results in layered communities, that is, services of similar quality are exchanged among peers of the same layer. The quality of offered services is high in the top layer if there are high-performing peers in the peer-to-peer system, while in the bottom layer the services offered are in most cases useless or even harmful for other peers.

3.3.5. FairTrust

FairTrust [41] is a system with a trust-based fairness-oriented peer selection. This approach employs a reputation system for the peers, and aims to have the peers participating equally. The focus is rather on fairness of contribution load rather than maximizing the resources utilization.

3.4. Topology-based Strategies

3.4.1. Neighbor-based

Koo et al. [42], present a neighbor-based approach to resource allocation. These approaches try to group peers into classes, based upon their similarities. A first approach is based on the number of bytes downloaded by a peer at a given time, while a second one introduces the idea of comparing the contents stored in peers, in order to group the peers together in virtual groups. This second approach introduces the concept of similarity among peers in the same group.

3.4.2. Localization and Congestion-aware system

More recently, Fouda et al. [43] introduced a Localization and Congestion-aware system in order to improve overall capability by reducing total link cost. In their approach, they use the topology of the network to compare the available peers. According to this topology, and to their knowledge of the transfers already occurring in the network, the aim is to reduce the probability of congestion in the network, by avoiding highly used paths.

4. Summary

In this chapter, an overview of existing approaches to content streaming has been presented. By far, the most scalable approach is the Peer-to-Peer network paradigm. However, this approach raises some new issues for content streaming.

First, multiple P2P network schemes exist for content streaming. Indeed, a large amount of work has been done in Peer-to-Peer Video Streaming, with multiple approaches being created. Each of those work proposes a new network scheme. LiveSky proposes the most efficient scheme for Video-on-Demand streaming, with its hybrid CDN-P2P approach.

Peer-to-peer networks also raise a peer resource allocation problem. Several approaches aim to provide algorithms for automatic peer allocation. Those strategies tend to be limited to a single metric for evaluation. Furthermore, those approaches keep the same strategy for peer selection in any given context, meaning that their peer selection is not induced by the demand pattern faced by the system. Therefore, they could be enhanced by using multiple metrics, and adapting their peer selection algorithm to the context.

Ch. 3 : Analysis of a P2P-VoD System

In this section, we first analyze the main performance factors in P2P-VoD systems. Then, we present some evaluation metrics used with such networks.

1. Key performance factors of a P2P-VoD System

A P2P-VoD system is a complex architecture with respect to many aspects that impact the system performances. In this subsection, we present each of those performance factors. First, we will review the key performance factors affecting a P2P-based streaming system: the number of peers and uplink at each peer. Then, we explain how modifying the number of titles (size of the content library), and the number of parts (fragments) per title can influence a P2P-VoD system. Finally, we investigate the impact of the demand (VoD requests distribution over the content library) before showing how important the initial content dispatching is for the overall system performance.

1.1. Number of sessions per peer

Each peer can contribute to a certain amount of sessions in parallel. The more uplink a peer possesses, the more sessions it can contribute to at any given time. The number of simultaneous VoD sessions to which a given peer *j* can contribute simultaneously is given by (1):

$$U_j = \frac{E_j}{B} \tag{1}$$

where U_j is the number of simultaneous sub-sessions for peer *j*, E_j is the uplink available at peer *j*, and B is the uplink bandwidth required to stream a title part.

U_j is critical to increase the overall P2P VoD system capacity. By increasing the capacity of each individual peer to contribute VoD sessions, the overall system is able to manage and satisfy more sessions.

1.2. Number of peers

In a P2P-VoD network, each peer is both able to request and to stream titles available in the content library. This means that the VoD system capacity increases with the number of peers active in the network – as the number of peers increases the demand (VoD requests) becomes more distributed across the peers' base, and the system can manage ratios of requesting/idle peers. Each peer has its own uplink capacity, with its own limit on the number of simultaneous sessions to which it can participate. The theoretical maximum number of sessions U that can be in the VoD system is given by (2):

$$U = \sum_{j=0}^{N} U_j \tag{2}$$

where U_j is the maximum number of sessions of peer *j*, *N* is the total number of peers, and *U* is the maximum number of sessions that can be supported in the system. If we assume that all peers have the same maximum number of sessions U_0, (2) is reduced to (3):

$$U = N * U_0 \tag{3}$$

where *N* is the number of peers, U_0 the maximum number of sessions that can be supported by a peer (based on uplink capacity), and *U* the maximum number of sessions in the network.

Furthermore, it is worth recalling that the titles are fragmented into parts and stored at the peers. By increasing the number of peers, we also increase the overall space for storing titles. Then, assuming all peers have the same storing capacity, the overall storage capacity of the network is defined by (4):

$$K = N * K_j \tag{4}$$

where *N* is the number of peers, K_j the storage of peer *j*, and *K* the total storage capacity of the network. We show that the storage capacity K is less critical than the uplink capacity U; this latter is the main constraint of the overall system capacity, in terms of the number of simultaneous VoD sessions that can be supported.

1.3. Number of titles

In a P2P-VoD system, a large content library of titles should be made available for all peers. A large content library is indeed critical to increase the VoD system utility. Those titles are fragmented and stored at the peers in a distributed manner. The larger the content library, the more space we require in order to store all of them. Furthermore, with a bigger library, the users are given more choices, and their queries may become more varied. Working with a large number of titles is one of the main difficulties in VoD systems [45]. Rarely requested content poses an important burden on the VoD system: as the demand (VoD requests) is horizontally stretched over different titles, it becomes extremely hard to maintain a high VoD system capacity in terms of simultaneous VoD sessions that can be supported [51].

1.4. Popularity per title

Titles provided in a VoD system have their own popularity, defined by the probability of being requested during a given period of time. Highly popular titles require a higher availability in the network (with more copies spread across more peers) in order to satisfy the demand for these latter titles and minimize the VoD requests rejection rates. It is important to note that the popularity distribution across the content library usually follows a Pareto model where 20% of titles tend to generate 80% of the demand [53].

Zipf's law (see Figure 10) is commonly used as a model to capture demand distribution. Early in 1994, Dan and Sitaram [54] considered the distribution of hits on the available videos and chose Zipf distribution to model video popularity. Wolf and Yu, in 1997, noticed that in their study, with varying degrees of skew week by week. Breslau et al. [55] confirmed that Zipf-like distribution roughly matched their access pattern when analyzing webpage requests distribution. Later in 2000, Acharya and Smith [56] showed that Zipf distribution with a fixed parameter α does not accurately model the video file popularity distribution. In 2002, Cherkasova and Gupta [57] argued that although the distribution of client accesses to media files can be approximated by a Zipf-like distribution, the time scale plays an important role in this approximation.

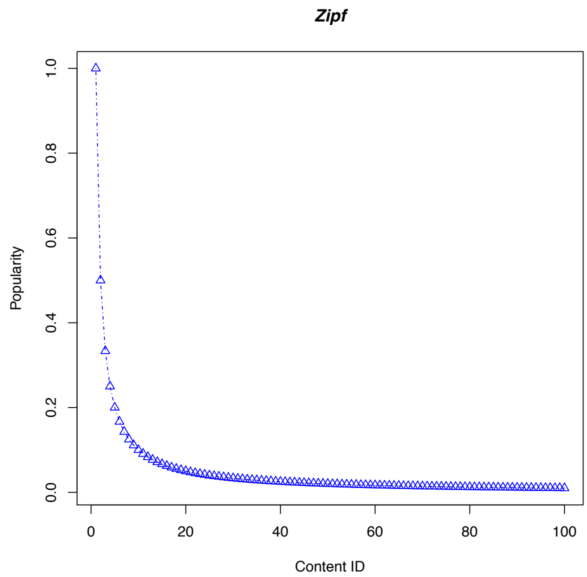


Figure 10: Representation of Zipf's Law for 100 contents.

1.5. Number of parts per title

In most P2P-VoD systems, titles are divided into smaller parts [45]. This enables peers to stream more titles at a time, because the bandwidth required to stream a title fragment is much lower than the one required for the entire title. Having titles fragmented into a higher number of fragments means that less uplink bandwidth is required at each contributing peer, while more peers will be required to satisfy a given VoD session. While the overall quantity of uplink required is unchanged, a larger number of fragments tend increase the overall system capacity [45].

It is important to note that a higher number of title fragments would mean that a given VoD session will be provisioned through as many VoD sub-sessions streaming from contributing peers. Testbed tests shows that it is very challenging to coordinate a large number of multi-source streaming sessions while meeting stringent video streaming QoS [52].

1.6. Demand pattern and evolution

The VoD content demand generated by the peers (i.e., users) usually varies drastically throughout the day and for different days of the week. Peak demand hours usually happen around primetime hours, and lead to a surge in the number of VoD requests with a high number of active multi-source streaming sessions that pushes the system to saturation. It is important to highlight the link between content demand and network saturation. A VoD request is indeed satisfied using the uplink of several contributing peers and for the duration of the content (usually 2 hours for a movie). Many research works in the literature have proposed mathematical models to analyse demand variations over the time.

1.7. Original Content Dispatching: initial titles injection in the P2P network

In order to deploy content in the network and make it available for later P2P VoD streaming, each title is split into parts that are duplicated based on their expected popularity before being stored in different peers. First of all, it is important to determine the number of times each title should be duplicated. Multiple works have been done in that direction. In this work, we use a popularity-to-availability model to establish how many copies of every title should distributed in the network (5).

$$C_i = K * p_i \tag{5}$$

where C_i is the number of copies of title i, K the total storage space, p_i the popularity of title i. Then, a strategy is used to inject the titles in the network. Existing research works have different approaches to achieve an efficient content dispatching strategy.

2. Performance evaluation metrics of a P2P-VoD System

In order to evaluate P2P systems, it is necessary to define some metrics. In this subsection, we introduce the Rejection Rate, the Success Rate, the Entropy, and the Latency metrics.

2.1. Rate of VoD sessions rejected

Whenever a peer requests a title, if all of the parts are available then this VoD request is deemed successful. If a part of this title is not available in the network, the VoD request is rejected. The rejection rate of our system is calculated as (6).

$$R(\Delta) = \frac{r(\Delta)}{d(\Delta)} \tag{6}$$

where $R(\Delta)$ is the VoD rejection rate over a time period Δ, $r(\Delta)$ the number of sessions rejected over the time period Δ, and $d(\Delta)$ the total number of VoD requests received over the time period Δ.

The VoD rejection rate is an important performance metric as it can be used by the service provider to quantify to what extent the demand is being rejected. This metric has a direct implication how the content demand is being met, and whether the service providers are missing on revenue opportunities. The VoD rejection rate can therefore be a useful indicator of the efficiency of the resource allocation strategy.

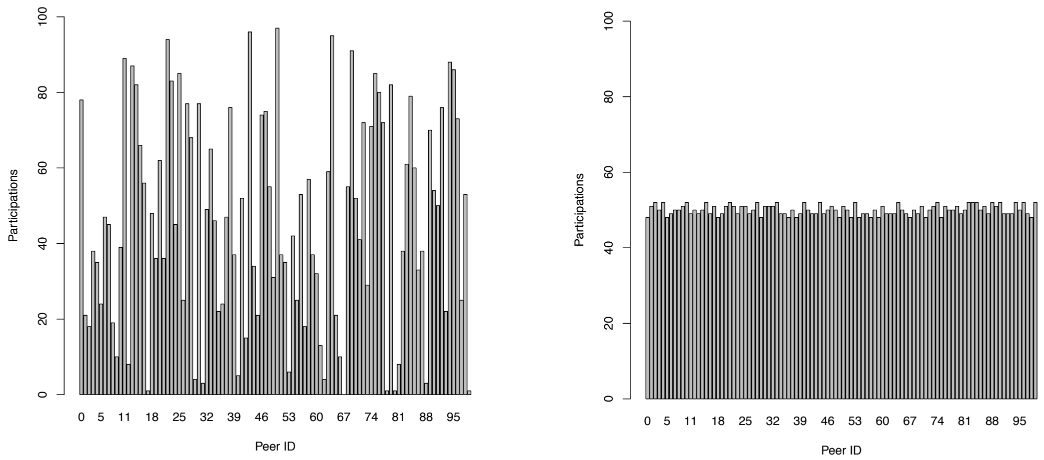
We also define the success rate, $S_l(T)$, described in (7).

$$S_l(\Delta) = 1 - R_l(\Delta) \tag{7}$$

where $S_l(\Delta)$ is the Success Rate of strategy *l* on period Δ, and $R_l(\Delta)$ the rejection rate using the strategy *l* on period Δ.

2.2. Peer Participation and streaming load distribution over active peers

In a P2P network, each peer contributes to the system by streaming parts of titles to other peers. To evaluate the peers participations' level, we use the entropy of the participation of every peer during the considered emulation time. It allows to define how well spread the participation is among the peers (see Figure 11). Clearly, one of our objectives in designing a P2P-VoD system is to make sure that all peers are equally taped, which should increase the system utilization.



(a) (b)

Figure 11: Comparison of two Peer Participation Entropies – (a) H=6.1 (b) H=6.8.

The definition of entropy is introduced in information theory [49], which describes entropy as a way to express the level of heterogeneity of a variable. Gomez et al [50] also used the entropy to analyze P2P traffic.

We express the participation rate of a peer in (8):

$$R_j = \frac{r_j}{\sum_{k=0}^{N} r_k} \tag{8}$$

where R_j is the participation probability of peer *j*, and r_j the number of times peer *j* participated. Then, the Entropy ($H(N)$) is calculated by (9):

$$H(N) = -\sum_{j=0}^{N} \left[R_j * log(R_j) \right] \tag{9}$$

where $H(N)$ is the entropy for *N* peers, and R_j the participation rate of peer *j*. Then, the aim is to maximize the entropy, meaning that all peers participated equally.

2.3. VoD Provisioning Responsiveness

The responsiveness measures the average delay necessary to fulfill a VoD request: from the time of the reception of VoD request, to the time a list of contributing peers is generated. Responsiveness is a very important aspect to consider when considering delay-sensitive VoD streaming systems. Managing thousands of requests per hour could be very challenging for the SuperNode, as every single VoD request involves multiple database queries. When dealing with complex multi-criteria resource allocation (RA) algorithms, there will always exist a tradeoff between the efficiency of the RA algorithm, and its responsiveness.

3. Summary

In this chapter an overview of performance factors and evaluation metrics for P2P-VoD systems was presented. First, we discussed the multiple parameters that can impact on the performances of such systems, such as the size of the network and the number of contents shared within the network. Then, we presented metrics to evaluate the performances of a P2P-VoD system, the most important one being the VoD Success Rate.

III – SYSTEM DESIGN

Section 4 : P2P-VoD System

P2P-VoD systems are complex systems, relying on multiple parameters and design choices. In this section, we present an analysis of such systems, the parameters that influence the efficiency of it, and some methods to evaluate this efficiency.
First, we present the choices made for our solution, and present other implementation designs from the literature. Then, we propose some metrics to evaluate the efficiency of a P2P-VoD system. Two main metrics are then kept: server load, and peer participation fairness. In order for us to evaluate our optimization methods, we need a fast and reliable method. Therefore, we propose to build a P2P emulator, designed to evaluate the efficiency of our system, based on our evaluation metrics. This emulator consists of two parts: the central node, and the peers. In our implementation, the central node is the actual code that would be running in a real deployment. On the other hand, peers are emulated through a single computer that creates the peer requests and uses the central node responses to evaluate the efficiency of the approach.

1. Video Fragmentation

Before describing the basic components of the system, it is important to first briefly describe the main characteristics of the video fragmentation process, which is central to the peer-assisted VoD streaming architecture.

The video content is first transformed from a track-based video file format into a packet-level video file format ready for streaming. Afterwards, the aggregated packet-level video file is further sub-streamed into several complementary sub-streamed files (content fragments); each sub-streamed video file is meant to be streamed by a different contributing peer to the requesting peer. The objective here is to reduce the contribution (in terms of bitrate) of each contributing peer so as to overcome the limited uplink capacities in asymmetrical broadband networks.

The advantage of packet-level (in contrast with Frame-level) video content sub-streaming resides in the fact that the original video file would be split into sub-streams of packets with a, more or less, predictable constant data-rate for each sub-stream, leading to a more deterministic streaming system. As illustrated in Figure 12, an original video stream is sub-streamed at packet level into different sub-streamed complementary video files. The couple (start, step) is used to uniquely identify a given sub-stream from a particular video content. The parameter start represents the first RTP's Sequence Number (SN) of a sub-stream, while step represents the stride between successive RTP sequence numbers of packets belonging to the sub-stream. For instance, the sub-stream containing the first 20% of the packets in the aggregated video stream is identified by the filter start=1/step=5, meaning the sub-stream is composed of RTP packets with the SN= 1, 6, 11, 16, 21, etc.

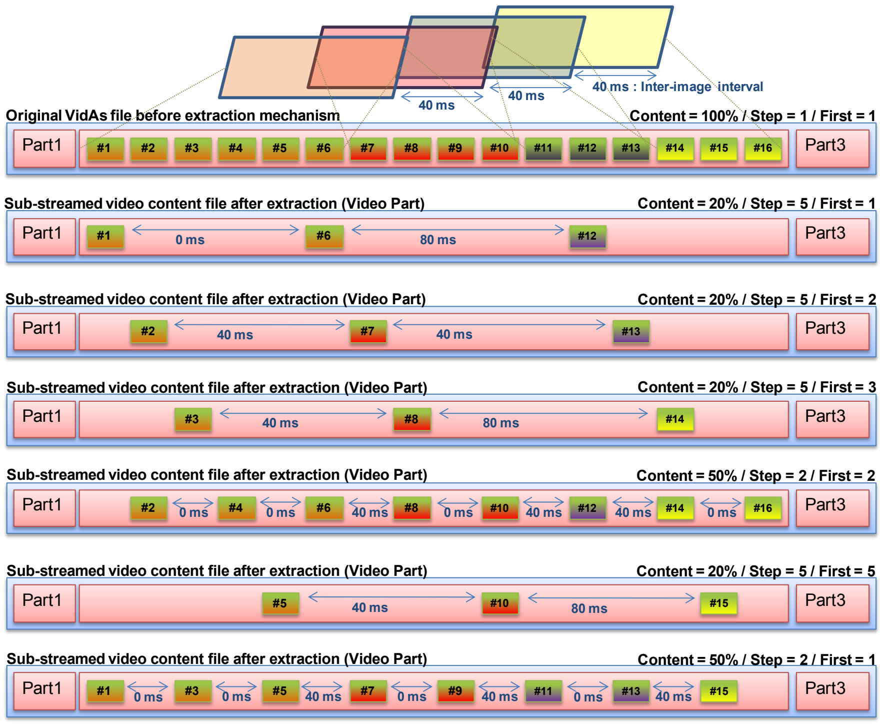


Figure 12: Packet-level video file sub-streaming.

The packet-level video file fragmentation requires managing different aspects to allow for effective multi-source streaming. The packets should be sequence-numbered and time-stamped in the same space to allow the receiving peer to multiplex the different sub-streams into one original aggregated packet stream to be appropriately decoded and displayed. Therefore, both RTP's [58] sequence numbers and time stamps are generated at the video fragmentation time with fixed starting RTP sequence number and time stamp so that the video content stay consistent as it spreads in the overall network upon subsequent VoD sessions completion.

2. System Components

It is now clear that the entire concept of Peer-assisted VoD architecture for broadband networks is based on the multi-source streaming that, in turn, relies on the sub-streaming of video content into complementary versions. In the following, we describe the overall network architecture, describing the role of all entities and the interaction among them.

The different components of the peer-assisted VoD architecture described above are depicted in Figure 13.

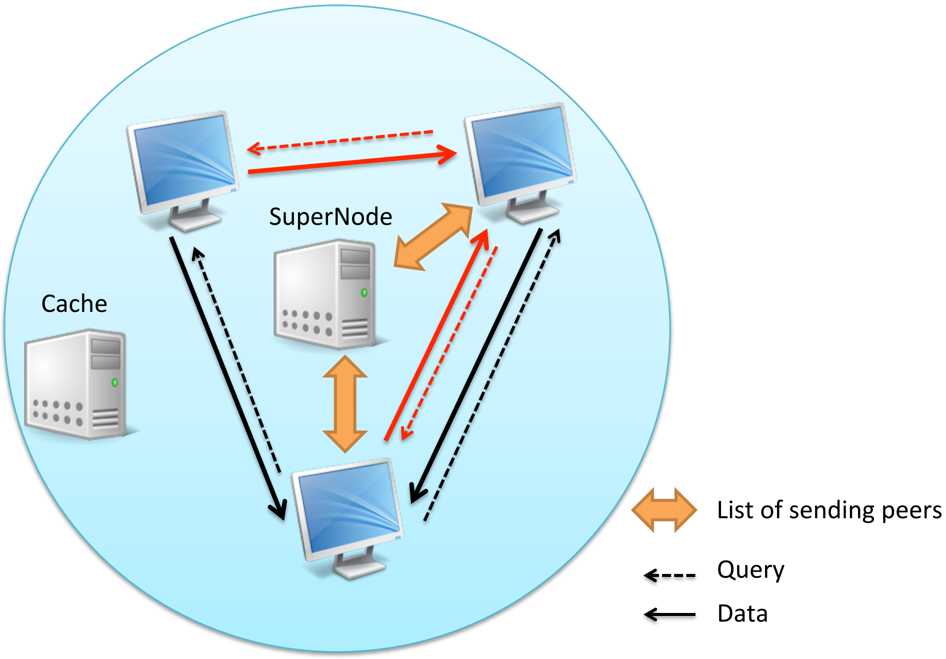


Figure 13: Content distribution system architecture: Managed Peer-to-Peer.

At the highest level, the system comprises of three basic components: the Supernode, the peers, and the Cache.

2.1. SuperNode

The SuperNode (SN) is where the main intelligence in the system lies – it allocates resources (uplink bandwidth of contributing peers) for each incoming VoD request. In order to perform this, the SN tracks two main resources: (i) the currently available uplink bandwidth at each peer, and (ii) the content stored in each peer. When a given receiving peer requests a specific VoD content, a VoD request is sent from that latter receiving peer to the SN. The SN looks up its database to determine the most appropriate set of contributing peers that might stream complementary streams to the receiving peer. Clearly, the SN handles the VoD session initiation signaling, while the load of serving the actual video streams is handled by the contributing peers and eventual caches;

2.2. Peer

The peer can be a Set-Top Box or some other intelligence (device with computing capabilities) which typically sits close to a user's viewing capabilities (TV, projector, and monitor). The peer is assumed to have significant storage capacities materialized by a hard disc that can contain sub-streams from different video titles of a video content library; upon sending a VoD request to the SN, a requesting peer will receive a response from the SN with a list of potential contributing peers that will provide complementary VoD sub-streams. The requesting peer then initiates real-time sub-streaming sessions with the contributing peers through RTSP/RTP protocols.

2.3. Cache

The cache is a large data store. The cache can be considered as a passive peer in the sense that it can contribute to VoD sessions, but will never request a VoD session in the network. The main role of the cache is to offset the limited uplink bandwidth capacities available at peers – the cache can also contribute to VoD sessions with much higher data volumes. Typically, in an Internet video streaming scenario, the caches will be located in a CDN provider's network in order to ensure high availability and performance predictability. The caches can be hosted by the broadband operator if the peer-assisted VoD platform is deployed as part of an IPTV solution.

3. Summary

In this section, we introduced a P2P-VoD system, complete with its implementation. It features three main types of components, (1) the Super Node, (2) the Peers, and (3) the Cache. This system has been fully tested and is used as a test-bed.

Section 5 : Content Dispatching

In this section, we investigate a very important aspect of CDNs and P2P systems: content dispatching.

The first question that comes to mind, once you have a full system, is: how and where do I store the contents? Because a large number of peers are available, we then have a large storage space at our disposal. This means that each content can be stored at the peers multiple times. We investigate the optimal content replication strategy. Also, peers have a limited uplink capacity. Therefore, multicast streaming will prove more efficient than a single-cast. This is one of the reasons presented for splitting the contents into chunks, which are then spread throughout the peers. Once the optimal replication and chunk number are found, it is then time to select at which peer each chunk is going to be stored. Content dispatching is a very complex problem, and we propose several algorithms to solve it, each of which is then evaluated with our simulator.

1. Popularity-to-Availability Translation

1.1. Impact of Video Content Fragmentation

It is clear that the number of copies of title in the network plays a preponderant role in the overall performances of the P2P-VoD system. Fragmentation of the video content into several complementary sub-streams has the advantage of creating a higher availability of the content since it is more likely to find all necessary sub-streams (for a VoD session) in the network, and have necessary uplink capacities at peers contributing with these latter sub-streams, during peak demand hours. Obviously, to a certain extent, the finer the granularity of content fragmentation the better the content availability will be. Any given video title will be available in a higher number of peers, lowering the probability of network saturation for that same content. Yet, a too high fragmentation granularity might not necessarily translate into a higher availability, and might instead lead to a lower reliability of the VoD system when considering a realistic failure probability in the system due to peers being frequently turned off or other reliability factors related to the underlying network infrastructures.

1.2. Proportional Content Popularity to Content Availability Model

In the proportional popularity-to-availability model, two video contents with the exact same popularity will be allocated the same storage capacity in the network with the same number of copies spread around the P2P network – assuming that the two video contents have the same duration and bandwidth requirements.

In the same way, a video content Cont-1 that is 10 times more popular than Cont-2, will be 10 times more available in the network. In this model, the relative popularity of each title is first calculated in terms of average number of requests received during a fixed period of time which gives a sense of the absolute popularity of the content. Afterwards, the different absolute popularities per content are transformed into relative popularities - with the sum of all titles' relative popularities equal to 1. This relative popularity is then translated into a relative storage space and, finally, to an exact storage space (i.e., number of copies in the P2P network) since the overall storage capacity available in the network is a-priori known by the network operator. The relative popularity of a title is calculated as shown in (1).

$$RP_i = \frac{P_i}{\sum_i^n P_i} \tag{10}$$

Here, RP_i is the relative popularity of the title *i* expressed as a fraction of 1. The relative popularity RP_i is derived from the absolute popularity P_i of the title *i* that is retrieved from the Zipf distribution. P_i is expressed in terms of an average number of requests received for the title *i* during 10 hours. This relative popularity RP_i is then easily transposed into a storage space in the network S_i to be allocated to the title i, using the total storage space available in the network TS that is a-priory known (see (11)). TS is, in fact, the sum of storage spaces available in each peer.

$$S_i = RP_i * TS \tag{11}$$

Finally, one can determine the exact number of copies per title *i* using the size (in bytes) of the video content *i*. It is worth mentioning that the video content size is usually tied to the video content duration, which has an important impact in the network dimensioning as it will be revealed in the following sub-section.

2. Content dispatching strategies

2.1. Serial content fragments dispatching

The serial content fragments dispatching strategy is the simplest algorithm to inject the content in the network. After, allocating a given storage space for each fragment using the popularity distribution in the content library, the algorithm consists in sequentially injecting any given fragment from any given title. We start by injecting the first fragment of the most popular title, moving to the second fragment, and so on. The peers are organized in a predefined list from peer #1 to peer #10,000. When injecting any new fragment, the algorithm checks if there is enough space in the peer #1, then the peer #2, and so on.

2.2. Random content fragments dispatching

It is important to balance the overall VoD streaming load evenly over the peers. One way to ensure that is to dispatch the content fragments randomly in the network – in this case the peers are selected randomly when the different content fragments are being injected. It is important to note that the overall content availability per title is still the same as with the serial content dispatching; it is based on the relative popularity-to-availability model discussed in 1.2.

2.3. Popularity-Weighted content fragments dispatching

As discussed earlier, it is important to make sure that all peers contribute to the same extent in the peer-assisted VoD delivery process. Not only the content fragments pertaining to a given title need to be as spread in the network as possible, but also all the peers need to equally contribute in the P2P streaming delivery process.

In order to achieve this last point, the fragments will be first assigned a weight that is proportional to their relative popularity (i.e., relative popularity of the title they are part of); then, each time a peer receives a fragment its overall popularity score is incremented by the fragment's popularity weight. In this case, the content dispatching process will each time make sure that all peers have a popularity score associated to them, and updated as the fragments are being dispatched in the P2P network.

The popularity-weighted dispatching algorithm takes also into account the content spreading constraint by favoring the title' fragment injection in a new peer if this latter already contains a fragment of the same title being currently injected. More specifically, the algorithm first considers injecting a given fragment content in the peer with the lowest popularity score; if this latter already contains a fragment of that content, then the algorithm considers the peer with the second lowest popularity score, and so on. After each fragment injection in a given peer, the peer popularity score SPS of this latter peer is augmented with the fragment popularity; also, the list of peers, re-ordered from the lowest SPS to the highest, might change after each single fragment injection. The algorithm can as well track the number of fragments from a given title being stored in any peer in order to extend the above reasoning line to a case where there are fewer peers with larger capacities compared to the content library size.

The popularity-weighted content dispatching strategy aims at resulting in a balanced P2P streaming network with the different peers having the same popularity score and contributing at the same level in delivering VoD sessions.

3. Summary

In this section, we showed the impact of content dispatching on P2P-VoD systems. Then, we introduced multiple dispatching strategies. In order to propose a fair strategy, we created the popularity score, used to rank peers based on the popularity of the contents they hold. Then, we presented a content dispatching strategy based on this ranking, the Popularity-Weighted Content Dispatching.

Section 6 : Static Resource Allocation

When a peer requests a content, because of content replication, multiple peers are available to seed the content. In this section, we investigate how to select the contributing peers.

Each content is replicated multiple times in the network of peers. Therefore, when a peer requests a content, multiple peers are able to seed said content. Peer selection is a very complex problem in peer-to-peer systems. We first present this problem as a resource allocation problem. Then, we present and evaluate static strategies for resource allocation. Those static strategies have many advantages, and each performs well when facing certain content demand. Such demand fluctuates over time. Depending on the time of the day, the day of the week, and even the month of the year, demand changes. We present some work from the literature regarding this evolution. Then, we characterize each strategy in relation with the observed demand pattern.

1. Introduction to Peer resource allocation

1.1. Why resource allocation is a key performance factor

The architecture presented previously clearly shows how a broadband operator can push to the network edge most of the complexity and cost associated with the process of provisioning VoD services. Accommodating incoming VoD requests at the SN becomes an exercise of finding all necessary/complementary content fragments (sub-streams) at peers that have necessary uplink bandwidth. In the following, we refer to the process of finding appropriate contributing peers to satisfy a given VoD request as "*resource allocation*" (RA). As it will be revealed in the following Sections, the resource allocation task is of utmost importance to optimize the network (peer) resources utilization by delaying the occurrence of network saturation events, and reducing their persistence.

Since the resource allocation task is performed in real-time basis every time the SN receives an incoming VoD request, the VoD service provider can dynamically vary resource allocation strategies to do several things, such as:

- *Accommodate a change in the content library,* where new titles are added and other replaced.
- *Vary the RA strategy throughout the time* in order to better accommodate different time of the day (primetime, working hours, etc) different days of the week, and different weeks of the year. One can build dynamic RA strategies that better accommodate the usual popularity distribution (per content category) changes.
- *Capture a shift in general popularity trends,* when the initial popularity distribution over the titles of the content library shows a fundamental long-term change.

It is worth noting that the content fragmentation and dispatching is done beforehand following advanced popularity-to-availability translation models.

Following the content dispatching, the SuperNode (SN) is made aware of what every peer contains in terms of content fragment. The SN is also aware of peer's uplink capacity reserved for the contribution to VoD sessions (through multi-source streaming); the SN then keeps track of the available uplink capacity per peer as VoD sessions are served and uplink capacity consumed.

Clearly, the uplink capacity available at peers is the most important resource in the above introduced P2P VoD streaming system. A non-optimal resource allocation strategy would typically over-use the uplink bandwidth of critical peers and cause a premature and prolonged situation of network resources. Network saturation will lead to very high VoD request rejection rates, and obviously a loss of revenue for the service provider.

1.2. Characterizing the resource allocation process

The objective of any P2P streaming resource allocation strategy is to maximize the success rate, $S_l(T)$, described in (16). This success rate, computed for a given period *T,* depends on the strategy *l* used in that previous period. The number of titles and the number of peers are fixed during the evaluation. Characteristics such as the number of sessions available at each peer and the number of sessions necessary to stream each title, are also fixed parameters. Every period of time T, we evaluate the performance of each strategy *l,* and select the one that maximizes $S_l(T)$.

Parameters:

$S_l(T)$: Success Rate of strategy *l* on period *T*

x_k: Success of demand *k*: 1 if success, else 0

d: Total number of VoD requests – this is the aggregated demand level

Variables:

l: Resource allocation strategy selected

Metric:

We select the strategy *l* that maximizes $S_l(T)$, the success rate on period T (12).

$$S_l(T) = \frac{\sum_{k=0}^{d} x_k}{d} \tag{12}$$

2. Single-metric based resource allocation

In this Section we introduce three resource allocation (RA) strategies we consider, explaining the characteristics of every strategy and the performance objectives behind their respective designs. We will particularly emphasize the ability of RA strategies to accommodate varying content popularity distribution, fairness among content popularity categories, high demand for VoD services, etc.

It worth noting that we classify the different resource allocation algorithms into two distinct categories: passive and active. In the passive the resource allocation algorithm use pre-calculated metrics to select appropriate contributing peers for an incoming VoD request. On the other hand, an active resource allocation will rely on performance metrics that vary over time.

2.1. Caracterisation of basic resource allocation strategies

Resource allocation strategies aim to define a ranking function *f*, in order to select the best peer θ that can satisfy the current VoD request. *f* is defined in (13).

$$\theta = arg_j \max\left(f(\theta_j)\right) \tag{13}$$

Where θ is the best peer for the current VoD request, *f* the ranking function, and θ_j is peer *j*. In the following subsection, we introduce three resource allocation strategies and their respective ranking function, *f*.

2.2. Higher Available Uplink Capacity First (HUF)

In this simple active resource allocation strategy, the peers are discriminated and ranked using their available uplink bandwidth. Whenever a new VoD request is received, the central server searches its database and retrieves all peers that can contribute in delivering the VoD session. At this point, the server will choose contributing peers with the highest available uplink bandwidth. Each time a resource allocation decision is made the uplink bandwidth capacities associated with the selected contributing peers is updated to reflect the resources used up by the provisioning of the recent VoD session. This means a current resource allocation decision will unavoidably influence future ones.
The ranking function is therefore defined as (14).

$$f: \theta_j \to u_j \tag{14}$$

Where θ_j is the peer *j*. u_j number of new VoD sessions that can still be supported by peer *j*.
The focus of the HUF strategy is to make sure that the uplink capacities of the different active peers are equally exploited. The idea behind this strategy is to maximize the utilization of the peers in an effort to not over-use some peers – and lock the title fragments they contain – while other peers still have abundant uplink capacity. At the highest level, the objective of HUF is to deplete the peers resources in a uniform manner as the demand for title surges. This prevents the rapid saturation of part of the peers pool, which will cause a shutdown of important bandwidth and content resources.
By focusing on the available uplink, this strategy tends to keep the number of peers saturated to a minimum. This proves to be very efficient when the demand is increasing – with HUF the peers (peers) resources are evenly depleted as the demand surge, which means the content availability stay relatively high. This strategy is said to be “active” because based on how saturated the system is, it can choose different peers based on their available uplink capacity.

2.3. Lowest Popularity Score (LPS)

This resource allocation strategy relies on the SPS (peer Popularity Score) metric, which is used to individually measure the popularity (resp., importance) of every active peer. During the title dispatching phase, every time we inject a new title fragment in a given peer, we increment the peer popularity score (PS) with the relative popularity of the title fragment. The PS essentially measure the importance of the peer and the likelihood of it being tapped to satisfy future VoD requests. The PS is also used in some advance title injection strategies (e.g., popularity-weighted title injection, introduced in Section 2) in order to make sure that all peers have comparable popularity, and will consequently be equally utilized during the VoD service provisioning phase.

We assume that the popularity's score of each title which is directly related to the probability of the title to be requested. Then, we define the Popularity score of a peer as shown in (15).

$$\rho_j = \sum_{i \in Q_j} p_i \tag{15}$$

where ρ_j is the Popularity's Score of peer *j*, p_i is the popularity of title *i*, and Q_j the set of parts held by peer *j*.

LPS is a passive resource allocation strategy that uses the SPS (peer Popularity Score) to select the different contributing peers necessary to satisfy an incoming VoD request. The idea here is to each time select peers with the lowest SPS in order to preserve the peers containing the most popular title fragments (i.e., peers with the highest popularity score). A resource allocation strategy that relies on LPS would primarily preserve the resources of popular peers; on the other hand, the LCS strategy (described below) does exactly the contrary by preserving peers with the scarcest title.

This strategy is very suitable for most demand environments. As long as the demand is according to the predicted content popularity, this strategy is very efficient. It starts showing its limits as soon as there is a shift in the content popularity trends. It also underperforms during peak demand hours because it doesn't take into account the available uplink, and thus cause certain peers to be depleted long before others.

2.4. Lowest Critical-Score (LCS)

This is a passive resource allocation strategy based on the Critical Score (CS) associated with each peer. The critical score is used to rank the different peers in respect to their criticality to the VoD session delivery process. It is used as a complementary indicator besides the SPS to identify peers that contain very rare title; these latter peers should be

consequently preserved as much as possible because the less popular titles are usually the most affected with high VoD rejection rates during peak demand periods. In other words, peers containing less popular titles are the first to get their uplink capacity saturated, leading to excessive rejections of VoD requests targeting the less popular titles.
We compute the Critical Score ∂ of a title in (16).

$$\partial_i = \frac{1}{C_i} \tag{16}$$

where $\partial_i$is the Critical Score of the title *i,* and C_i is the number of copies of the title *i* in the network. Then, we define the Critical Score of a Peer as shown in (17). A title with a low critical score means that the title is not very popular so there are only few copies of the title stored in the network.

$$\Delta_j = \sum_{i \in Q_j} \partial_i \tag{17}$$

Where Δ_j is the Critical Score of peer *j,* $\partial_i$is the Critical Score of the title *i,* and Q_j the set of parts stored at peer *j*. Calculating the critical score of a peer would essentially allow one to measure how many scarce title is contained in any active peer.
After computing the critical score of each peer, the peers that contain less popular titles will have a higher critical score. This is due to the fact that less popular titles have a very limited number of title fragments spread in the network compared to popular titles.
In the LCS-based resource allocation strategy the RA algorithm tries to use peers with the lowest CS when building the list of contributing peers in response to a VoD request. This way, we minimize the excessive rejection rates affecting the VoD requests targeting the less popular titles. Less popular titles are usually affected by the highest rejection rates. The ranking function is therefore defined as (18).

$$f: \theta_j \rightarrow \frac{1}{\Delta_j} \tag{18}$$

Where θ_j is the peer *j*. and Δ_j the critical score of peer *j*.
When the demand is very high, rare (long tail) contents are likely to be requested among all the queries. Considering that there are a disproportionate number of long tail titles, preserving peers containing these content tend to produce better performances during peak demand hours. Further, LCS tends to outperform other RA strategies when the content [popularity shift, and new titles become very popular.

3. Multi-criteria resource allocation

3.1. Hierarchically Combined Resource Allocation Strategy

The Hierarchically Combined Resource Allocation strategy is based on selecting the peers by using multiple metrics, organized in a hierarchical way. By increasing the number of metrics in and improving the resource allocation decision with a two-round process we aim to increase the overall system performances and flexibility by exercising a high degree of control.

Very simply put, during a resource allocation decision the objective is to first choose the set of suitable contributing peers using a first criterion (say HUF). Afterwards, a second criterion (say LPS) is used to have a second level. Figure 14 shows an example of Hierarchic Resource Algorithm.

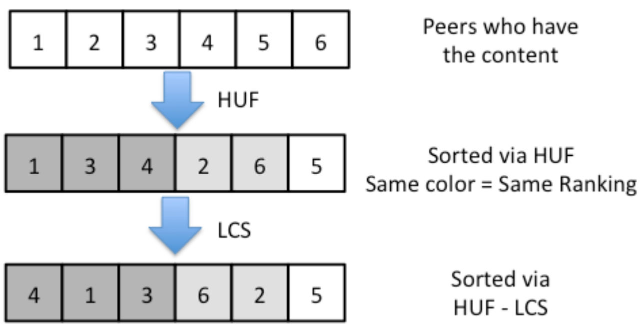


Figure 14: Combined Resource Allocation example : HUF-LCS

3.2. Multi-objective Optimization

To solve the multi-objective problem, the plain aggregating approach is used. This approach consists of converting the problem into a single-objective optimization problem by aggregating the objectives. Afterwards, we can apply any single-objective optimization technique [59][60][61] to the new objective function. This technique has the advantage of producing a single compromise solution requiring no further interaction with the decision maker. Mathematically, the new objective function (criterion) f_{eq} is written as shown in (19).

$$f_{eq}(\vec{x}) = \sum_{i=1}^{k} w_i . f_i(\vec{x}) \tag{19}$$

Where $f_1 \dots f_k$ are the original objective functions to be minimized and $\vec{x}$ is parameters vector; w_i are weighting parameters that satisfy the following relations (20).

$$\begin{cases} 0 \le w_i \le 1 \\ \sum_{i=1}^{k} w_i = 1 \end{cases} \tag{20}$$

The weights w_i are also known as *importance factors* and are considered as a measure of the significance of each objective in the optimization process. A representative convex part of the Pareto[62] set can be sampled by running several times a single objective optimization algorithm, each time with a different vector of importance factors.

This process can be illustrated geometrically. In the objective function space a line *L* according to $w^T f(\vec{x}) = c$ is drawn (see Figure 15). The minimization of f_{eq} can be interpreted as searching for the value of *c* for which *L* just touches the boundary of the feasible domain **A**, as it proceeds outwards from the origin. Selection of weights w_1 and w_2, therefore, defines the slope of *L*, which in turn leads to the solution point where *L* touches the boundary of **A**.

w^T is the transpose of the vector of weighting parameters;

c is a constant;

A is the feasible domain;

Line *L* corresponds to: $w_1 f_1 + w_2 f_2 = c$

f_1 and f_2 are the objective functions;

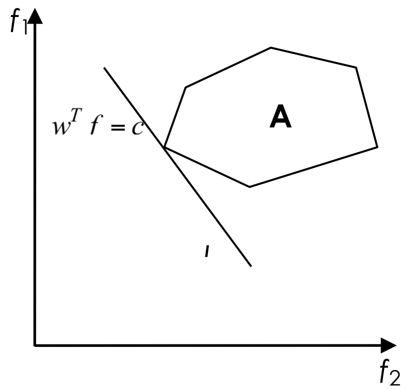


Figure 15: Representation of the plain aggregating approach in the bi-objective case.

Upon the reception of a VoD request at the SN, each of the basic RA strategies returns a list of peers able to seed the fragments (sub streams) necessary for the VoD session: HUF, LCS, and LPS. Then, we build a new list by combining those lists: for each requested content fragment, we randomly pick one of the three solutions obtained. It is worth recalling that every VoD sessions is provisioned from 4 sub-streams (content fragments. The new objective function will afterwards minimize (21).

$$F(\vec{x}) = \sum_{i}^{STB} \alpha * \frac{PS_i}{PS_{max}} + \beta * \frac{CS_i}{CS_{max}} + \gamma * \frac{Up_i}{Up_{max}} \qquad (21)$$

With: $\alpha + \beta + \gamma = 1$, $\vec{x}$ the current solution, where : peer is the total number of peers (10,000), Up_i is the available uplink of the peer i, CSi is critical score of the peer i and, PSi is the critical score of the peer i.

Each time we generate a new solution we compare its fitness (F) with the previous best solution. We store the best solution, and repeat these steps until the stopping criterion is satisfied. Here, the stopping criterion is N iteration in order to achieve a minimum responsiveness; where N is an empirically fixed integer.

At the end of the process, we return the solution stored, which has the best Fitness amongst all the generated ones. The whole procedure is illustrated in Figure 16.

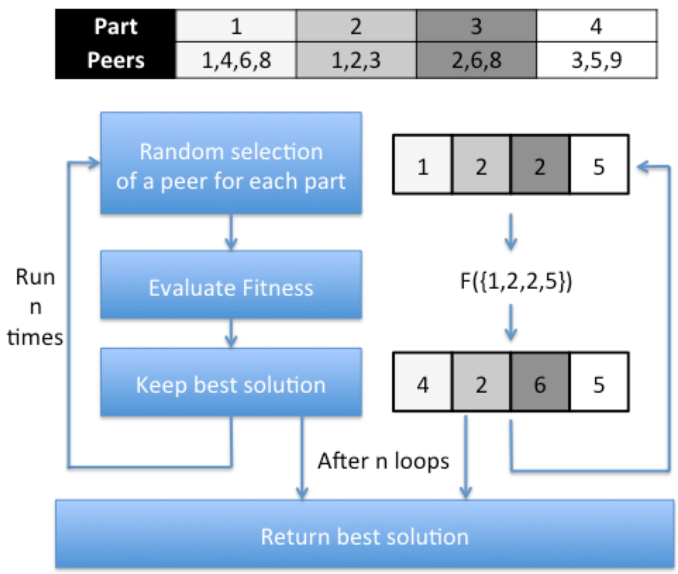


Figure 16: Multi-objective resource allocation optimization.

The values of α, β, and γ were fixed empirically and allow to obtain the best approximation of the problem to a linear problem.

This approach will come with significant drawbacks, however. In fact, it requires pre-established weights for the different RAs. Obviously, one can determine ideal weights for each of the RA strategies in order to achieve the best performances over a long time period. However, this fixed strategy will fall short of achieving high performances during the different conditions that the system transition between in the short-term. The end result will be a significant underperformance compared to potential.

4. Summary

In this section, we presented resource allocation and its impact on P2P-VoD systems. Then, we proposed multiple single-metric based resource allocation strategies. In order to combine those resource allocations, we examined the possibility of using a hierarchical combination of the criterias, as well as a multi-objective optimization method. Those approaches are usually efficient for combining multiple strategies, but, in the P2P-VoD scheme, merging strategies tends to reduce the advantages of each approach, resulting in a less performing method.

Section 7 : Dynamic Resource Allocation

In this section, we describe an advanced RA strategy that uses a Bayesian approach to effectively alternate between RA strategies based on the popularity trends and the P2P network resources (content/uplink) availability. The originality resides in the very design of a Bayesian [63] approach that can characterize popularity trends and rely on any number of performance objectives (metrics) to come to effectively the final decision in terms of selected RA strategy. Finally, this dynamic RA algorithm can easily be augmented with additional basic RA strategies from which to alternate.

1. Towards dynamic resource allocation

1.1. Why performances evolve over time

In a real-life deployment of a VoD system, many aspects change over time as the system is running. First of all, the library of titles evolves, with new titles introduction and old titles deletion from the P2P streaming system. Also, the popularity of each title varies over time, depending on multiple criteria, such as the time of the year, dependency/correlation between titles' demand, and so on. Several works [64][65][66][67] have been made to predict the evolution of the popularity of a title in a VoD system. Content demand tends to vary on hourly, daily and weekly basis.

All the above fluctuation in demand will lead to a different pattern of VoD requests arrival rate and a different distribution of popularity over the content library. In this context, multiple resource allocation strategies would lead to various performances in terms of VoD system utilization, and timing and frequency of the saturation situation. Employing one single resource allocation strategy to cope with the fundamental changes in the demand would not produce the best performances. One should consider the use of different resource all action strategies to deal with different situations in terms of content demand shift. An effective dynamic switching between different RA strategies is of utmost importance here.

1.2. Why resource allocation is a key performance factor

As presented before, the content demand exhibits a changing pattern over time so the P2P-VoD system needs to adapt to this situation. The decision of allocating participating resources to an incoming VoD request is a complex one with diverse implications on the system performances. For instance, the resource allocation algorithm can identify many contributing peers that can be used to satisfy the VoD request – these latter peers can have different available bandwidth capacities, and different stored content. The decision to use one of these peers to satisfy the current VoD request means reducing its uplink capacity for an extended period of time, and thus possibly shutting the peer's content out of the system.

2. Dynamic resource allocation

As revealed above, this P2P streaming resource allocation (ARA) dynamically switches between different RA strategies on a real-time basis as the VoD system is running and processing VoD requests. In order to achieve this, we select the best-fit strategy by using a combination of multiple metrics.

2.1. Dynamic Switching between different RA strategies

At the core of this dynamic RA switching scheme is the ability to predict the most likely trends shifts in the content popularity. The VoD system operation time is split into discrete time periods and the popularity trends are constantly observed within each one of these time periods. In this experiment we use a 4-hour time period, but this can be changed in practice to match specific VoD consumption trends observed in the target audience. ARA essentially uses long-term popularity trends as the aggregate point to which the average popularity should eventually revert. In other words, ARA assumes relatively stable long-term popularity trends and treats the popularity distribution within the individual 4-hour time periods as deviations from the mean.

By analyzing the popularity trends in the previous time period, ARA can predict the following time period with a certain level of confidence. Again, the assumption here is that the long-run popularity trends are still valid, and any short-term deviation from the trend is only transient. More on the popularity prediction model are given in following sub-sections.

Once the popularity prediction for the next time period is performed – and the system has a good idea of the likely frequency and intensity of VoD requests for every title in the content library – ARA will assess how every basic RA strategy would perform in terms of both success rates and entropy. Bayesian approach is here used to effectively blend together both performance objectives (success rate and entropy) and use the result to compare how the different RA strategies would perform in the following time period.

It is worth recalling that the time period between each strategy switch can be adjusted to accommodate any particular situation. For instance, the service provider can increase the time period to 6 hour on a weekend where the demand is strong and sustained throughout the day. On the other hand, the time period can be reduced to 2 hours in a week day where most of the demand is concentrated around prime time.

2.2. Resource Allocation Strategy Evaluation and Selection

In this section, we aim to select the best resource allocation strategy for the upcoming period. For each of the resource allocation strategies (HUF, LCS, LPS), we estimate the probability of reaching a certain level of performances. Performances are evaluated in terms of two performance metrics: the success rate and the entropy. In order to combine those metrics, we use a Bayesian Fusion approach. As revealed earlier, this approach allows one to fuse different performance metrics.

Bayesian Inference is a statistical approach that aims to determine a probability, known as posterior probability, based on a prior probability and the previously gathered data. In the following, we explain how Bayesian is applied in dynamic RA strategy:

a) The popularity trends are predicted for the following time period (say 6 hours) based on past observations and assuming a long tail type popularity model that has been well researched in the literature [68].
b) We determine the *prior probabilities*: the success rate and the entropy, when using each one of the three RA strategies. This

consists of evaluating the performance of each one of the 3 RA strategies against the predicted popularity distribution for the following time period.

c) Based on the observations above, we calculate the probabilities of having the performance metric (success rate or entropy) in a certain range. We can then build a histogram that associates a probability of having the performances fall in any of the different ranges.
d) At this point, we can express the *probability density* as a function by applying a Gaussian fit on the histogram of the prior probabilities. The objective here to derive a function for the probability distortion that can be leveraged by the Bayesian approach. It is important to note that this process will lead to 6 Gaussian functions: 3 RA strategies by 2 performance metrics.
e) Then, we use Bayes' theorem to essentially combine the two Gaussian functions associated with the two performance metrics for any RA strategy. At this point, we obtain a common way to compare the different RA strategies while still considering the performances along the two dimensions: success rate and entropy. This processed is referred to as obtaining *posterior probabilities*.
f) In order to evaluate the different RA strategies against each other we use *Montecarlo Simulation* to pick the strategy that jointly maximizes both the success rate and the entropy throughout the following time period.

2.2.1. Prior probabilities

Bayes' rule provides a way to make inferences about an object described by a state x given an observation z. It requires an expression of the relationship between x and z to be expressed as a joint probability: $p(x, z)$.
In this case, the states x are the strategies HUF, LCS, and LPS, while the observations z are the entropy or the success rate. The goal is to determine the strategy with the highest probability of producing a high success rate and entropy level.
The chain-rule of conditional probabilities expands as shown in (22).

$$p(x, z) = p(x|z)p(z) = p(z|x)p(x) \tag{22}$$

Thus, Bayes' rule is obtained using (23).

$$p(x|z) = \frac{p(z|x)p(x)}{p(z)} \quad (23)$$

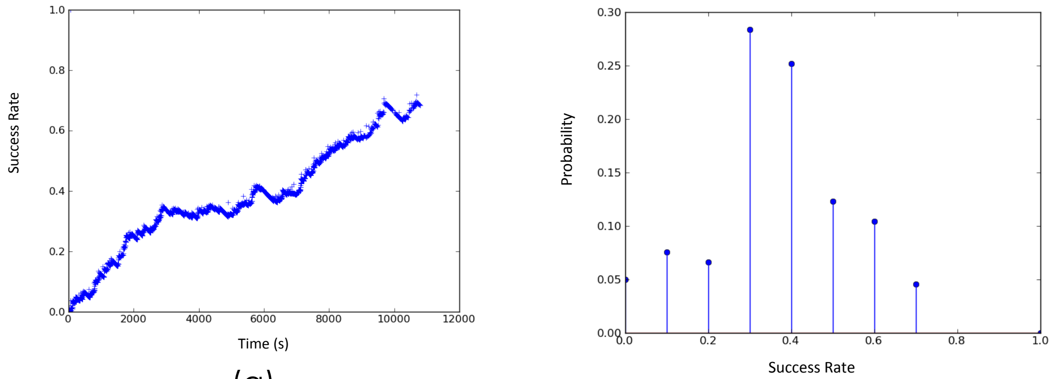
The value of this fundamental result depends on the interpretation of the probabilities p(x|z), p(z|x), and p(x). The objective here is to determine the various likelihoods of different values of an unknown state x. The prior probability, p(x), expresses the values of x that may be expected according to prior beliefs. On the other hand, p(z|x) describes, for each fixed state x, the probability that the observation z will also be made: the probability of z given x. The new likelihoods for the state x are obtained by multiplying the original prior information and the new information gained by observation. This gives p(x|z), which is the posterior probability, describing the likelihoods of x given the observation z.

In this approach, we then determine six probabilities distributions, two per strategy (one for each performance metric). The posterior probability of a strategy is in fact the probability of achieving a certain level of performance (success rate or entropy level) when using the said strategy.

2.2.2. Probability density

This fusion is based on the probabilities of achieving a certain level of performances in terms of success rate or entropy level. Therefore, in this approach, we start by evaluating p(S|Y_l), the probability density of success S knowing that we chose strategy Y_l, and p(H| Y_l), the probability density of entropy with strategy Y_l.

In order to better illustrate the core idea behind this Bayesian approach, Figure 17a shows how the success rate evolves over time in a typical time period, while Figure 17b shows the likelihood of having the success rate in any range given in Figure 17a. The likelihood is here expressed using a probability distribution in order to better substantiate the expected performance with more details than a mere average.



(a) (b)

Figure 17: Success rate analysis. (a) Success Rate over time, (b) Histogram of the success rate.

Finally, we apply a Gaussian fit on the probability density of each performance metric (success rate and entropy) in order to express those histogram as functions, as shown in Figure 18. The interest of doing so is to use the function in the Bayesian procedure to fuse together two (or more) performance metrics.

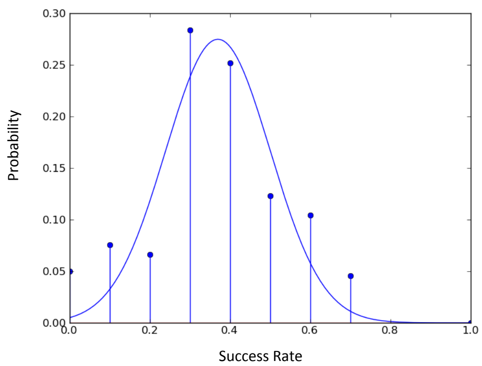


Figure 18: Gaussian fit of the success rate histogram.

2.2.3. Posterior probabilities

In this stage of the process, we compute the posterior probabilities of each strategy. To do so, we use Bayes' theorem in (24). Those posterior probabilities represent the probability of using each RA strategy, knowing individual performances along two dimensions: entropy level and success rate.

$$P(Y_l|H,S) = \frac{P(H,S|Y_l)P(Y_l)}{P(H,S)} \tag{24}$$

Where Y_l is one strategy, H the entropy, S the success rate.
Then, we select the best strategy, based on a maximum "*a posteriori*" estimation, computing the value (25).

$$Y_0 = arg_l \max P(Y_l|H,S) \tag{25}$$

Where Y_0 is the best strategy, *l* belongs to the group of strategies, Y_l is strategy *l*, H the entropy, S the success rate.

2.2.4. Monte-carlo Simulations

Now, we compare the performances of the three different RA strategies by evaluating the "a posteriori" probability of each strategy : $P(Y_l|H,S)$. We use *Montecarlo* simulation (Figure 19). This allows us to evaluate the different RA strategies with a significant added level of detail, which narrow the level of uncertainty regarding the outperformance of the final RA we choose for the following time period.
In a Monte-Carlo simulation, we randomly place points in a previously known area. In this case, we use the 1x1 square, since the curves are in that space. Then, we count the number of points that are in the area we aim to determine : here, under the curve. The area under the curve, as shown in Figure 19, is the number of points under the curve divided by the total number of points we placed, multiplied by the total area (26).

$$A = \frac{I}{O} * G \tag{26}$$

With A the area under the curve, I the number of points inside/under the curve, O the total number of points, and G the area where the points are generated.

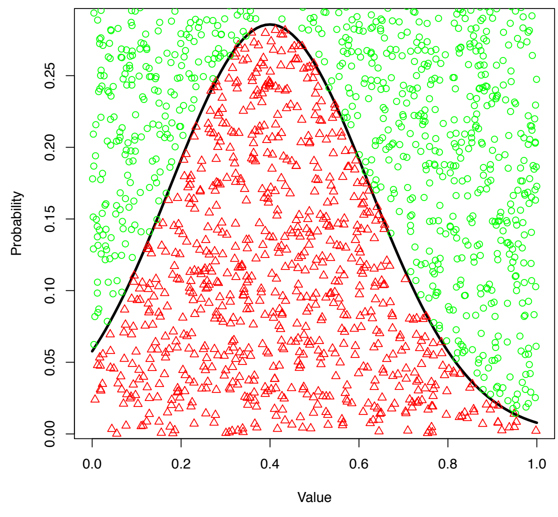


Figure 19: MonteCarlo simulation - 2000 shots - area: A = $\frac{746}{1256} * 0.33 = 0.19$.

The results of the Montecarlo simulation returns a value for each posterior probability. We then get three values, one per strategy. Since we are using a maximum a posteriori estimator, we select the strategy with the highest Montecarlo value. This strategy is the best fit for the upcoming demand, based on a Bayesian fusion.

3. Strategy selection based on Evidence Theory

In this section, we start by presenting the Evidence theory. This theory can be viewed as an extension of probability theory. It is suitable for characterizing uncertainty when evidence is imprecise because it allows one to estimate probabilities of intervals instead of probabilities of specific values. We then explain how this approach can be used to efficiently dyniamically select a peer resource allocation strategy.

3.1. From Bayes to Dempster-Schafer

Although some aspect of probability like the Bayesian probability is closely related with possibility theory, both differ in some major aspects. The Bayesian approach requires to make strong assumptions to estimate the likelihood of the available evidence. Because it is based on posterior probability evaluation, it is sensitive to imprecision in our predictions. Furthermore, this approach is quite heavy, due to the need to keep prior values stored in memory, in order to compute the new posterior probabilities.

On the other hand, the Evidence theory approach does not require the user to assume anything beyond what is already available. This approach treats uncertainty due to imprecision differently than uncertainty due to randomness. It is lighter to set up, and even computes faster.

3.2. Dempster-Schafer Theory

We first present the probabilities in those intervals, called basic probabilities. Then, we introduce Plausibility and Belief, which are used in evidence theory to characterize the belief in the occurrence of an event.

3.2.1. Basic Probability Assignment

The idea of the Dempster-Schafer Theory is that numerical values of uncertainty may be assigned to overlapping sets and subsets of hypotheses, as well as to individual hypothesis. These measures of uncertainty are known as "basic probability assignment".

Let $\Theta = \{h_1, h_2, \dots, h_n\}$ be a finite set of hypotheses (frame of discernment). A basic probability assignment (bpa) is a function $m: 2^{\Theta} \rightarrow [0,1]$ such that :

$$m(\emptyset) = 0 \tag{27}$$

And:

$$\sum_{x \in 2^{\Theta}} m(x) = 1 \tag{28}$$

All of the assigned probabilities sum to unity and there is no belief in the empty set. Any subset x of the frame of discernment for which $m(x)$ is non-zero represents the exact belief in the proposition depicted by x.

3.2.2. Belief function

The belief represents the confidence that a value lies in A or any subset of A. Therefore, a belief measure is a function $Bel: 2^{\Theta} \rightarrow [0,1]$, computed from the sum of probabilities that are subsets of the probabilities in question. We have :

$$Bel(A) = \sum_{B \subseteq A} m(B) \qquad \text{for all } A \subseteq \Theta \tag{29}$$

With A a subset of Θ, B the subsets of A, and $m(B)$ the basic probability assignment of B.

3.2.3. Plausibility function

The plausibility of a subset A represents the extent to which we fail to disbelieve A. It is a function $Pls: 2^{\Theta} \rightarrow [0,1]$, defined by:

$$Pls(A) = \sum_{B \cap A \neq \emptyset} m(B) \qquad \text{for all } A \subseteq \Theta \tag{30}$$

With A a subset of Θ, B the subsets of Θ that do not intersect with A, and $m(B)$ the basic probability assignment of B. The plausibility measure is clearly related to the belief function :

$$Pls(A) = 1 - Bel(\neg A) \quad (31)$$

With A a subset of Θ, and $\neg A$ the rest of the subsets in Θ. $Bel(\neg A)$ is also called the doubt in A.

3.3. Application to Resource Allocation Strategy selection

3.3.1. Plausibility applied to strategy evaluation

Several methods use the Evidence Theory for modeling uncertainty with consideration on the analysis of computed measures in expert systems [69]. There, the analysis is basically the comparison of the measures, i.e. possibility measures, coherent lower previsions, additive probabilities and belief function. Method of uncertainty is also known to be useful in the analysis of prognostics [70].

The aim here is to select the best strategy ω^* in a set Ω, based on multiple criteria. Figure 20 presents an analytical hierarchic modelisation of this problem.

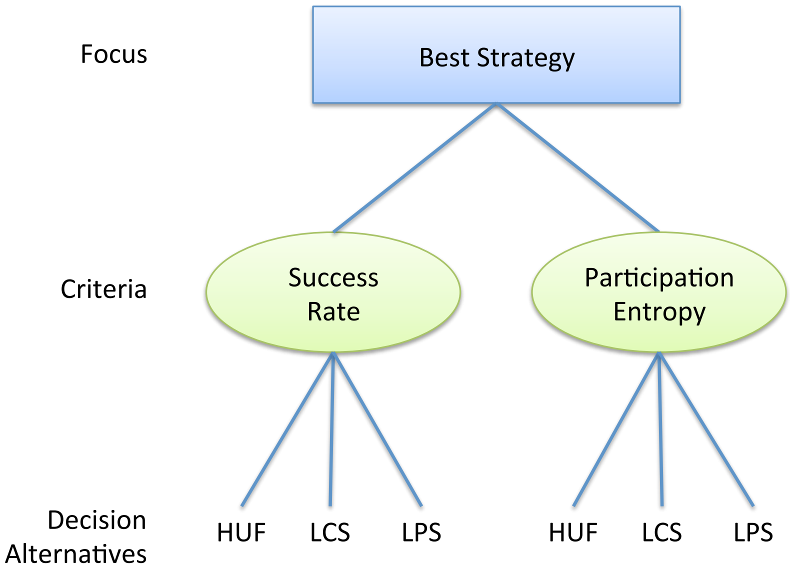


Figure 20: Analytical Hierarchy Process model of Strategy Choice

3.3.2. Problem formulation

In our approach, we select the best strategy using the maximum of plausibility estimator. We express the problem as follows:

Find $\omega^* \in \Omega$ such as :

$$\omega^* = arg_{\omega^* \in \Omega} \max \left(Pls(\omega)\right) \quad (32)$$

where ω^* is the best strategy, Ω is the set of all strategies, ω a strategy, and $Pls(\omega)$ the plausibility of strategy ω.

3.3.3. Plausibility estimation

In order to select the best strategy, we estimate the plausibility for each strategy to be the best one. In order to combine the basic probability assumptions, obtained for each strategy with each criteria, we use :

$$[m_1 \oplus m_2](x) = \begin{cases} 0 & if\ x = \emptyset \\ \dfrac{\sum_{B \cap A = x} m_1(A) * m_2(A)}{1 - \sum_{B \cap A = \emptyset} m_1(A) * m_2(A)} & if\ \ x \neq \emptyset \end{cases} \tag{33}$$

An important feature in the above formula is in the denominator, which can be interpreted as a measure of conflict between the sources. It is directly taken into account in the combination as a normalisation factor. The measure represents the mass which would be assigned to the empty set if masses were not normalised.
We then select the strategy that presents the maximum of plausibility : this strategy is the most plausible to be the best strategy.
It is key to note that, with a larger set of criteria and strategies, the number of solutions to evaluate increases, thus increasing the complexity. In such cases, a genetic algorithm can be used to reach an optimal solution in fewer iterations.

4. Summary

In this section, we presented a dynamic resource allocation. The system evaluates the performances of multiple strategies, and automatically uses the best fit strategy. In order to select the best strategy, two parameters are used : rejection rate, and peer participation entropy. Therefore, a combination of those parameters is required for selecting the best performing resource allocation. We first presented a Bayesian method. Then, another method, based on Evidence Theory, was introduced.

IV – SIMULATIONS AND RESULTS

Section 8 : Simulation Platform

In this section, we present the simulation platform used for our evaluations.

1. Experimental platform overview

Our emulator is a very close approximation of the behavior of a full-scale peer-assisted VoD streaming system. The Super Node is a complete implementation, that could immediately be used in a real system. The Peers, on the other hand, are simulated. A simulator generates the requests from the peers and communicates with the SuperNode.

First, we have a full-scale implementation in *Python* of the central server, called the Super Node (SN). The SN can process VoD requests in real-time. Each VoD request targeting a specific title leads the SN to lookup the database for peers with the content parts and enough uplink capacity to stream the content parts. The SN then returns a list of peers that can satisfy the VoD request by contributing a specific part to the multi-source streaming session – note that by deciding which peers to select for the VoD request, the SN is effectively making a resource allocation decision. The SN keeps the database up-to-date by reflecting changes after processing every new VoD request. Figure 21 illustrates the process at the highest level.

A second part of our emulation environment consists of the actual VoD requests emulator. Its role is to randomly generate VoD requests from different peers, and following a specific popularity distribution over the content library. The VoD requests are generated based on a realistic content popularity model with a typical Pareto's 20/80 rule.

The VoD requests generator reads a trace file containing a list of VoD requests, organized per hour, title requested, and the source of the request (peer's ID). Clearly, we can test different system parameters using the same demand, over the content library. Finally, the VoD requests generator is able to run up to 60 times faster than real time, which makes it possible to run a 10 hours emulations in 10 minutes.

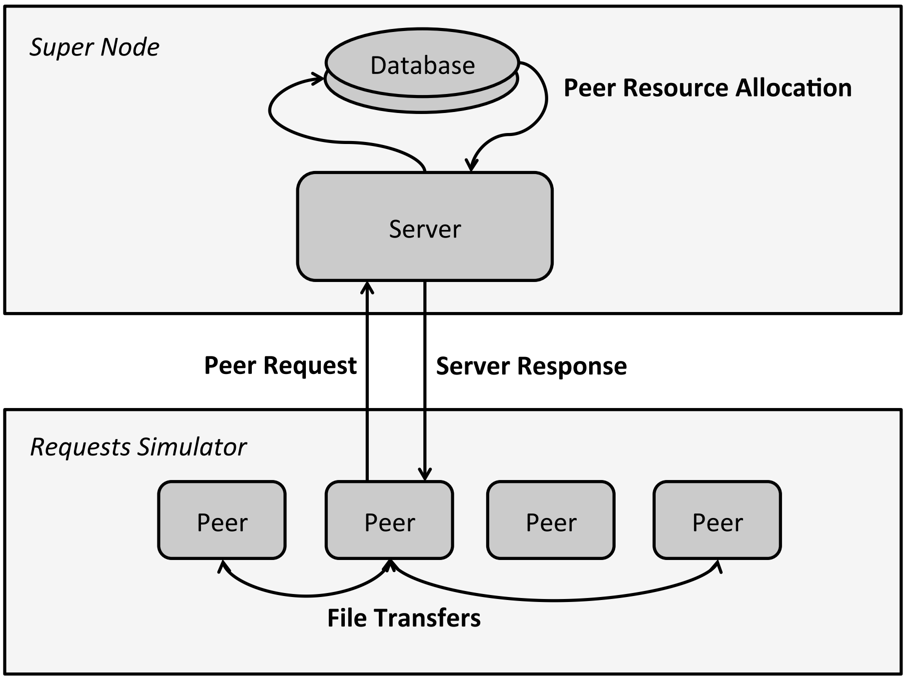


Figure 21: Implementation of a P2P assisted VoD system.

2. Dynamic evaluation platform

In order to select the best resource allocation strategy, we use an SN simulator to evaluate each strategy's performance for the upcoming time period (see Figure 22). The simulator implements our LB-RA algorithm with the Bayesian approach to select the best RA strategy. The SN simulator runs 10 times faster than the actual SN and is responsible of evaluating every RA strategy and logging its performance in terms of our two performance metrics: success rate over time, and entropy level over time.

There are two major steps in our experimentation with two different time frames. First, the VoD requests generator and the SN work in real-time to replicate a typical scenario that an SN will be faced with in a real deployment. On the other hand, the SN simulator actively evaluates the different RA strategies against predicted future content demand. It is crucial to note that the simulator computes the best strategy for the upcoming time period while the SN is still running and processing VoD requests of the current time period.

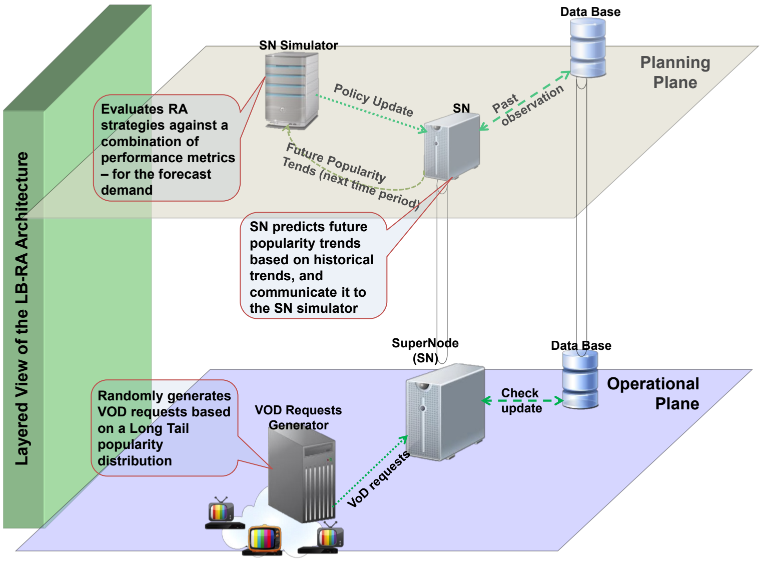


Figure 22: High-level view of the dynamic evaluation platform for LB-RA.

In order to evaluate each resource allocation strategy, we generate a trace file corresponding to the upcoming demand, and test each strategy with this trace file. Every 3 hours, the predictor (here located at the SuperNode) generates a trace file of the upcoming demand covering the next 3 hours. Then, it sends to the simulator a trace to request the *Strategy Update*. At this step, the simulator provides the best RA strategy to be used using the PL-RA algorithm.

The entire system is implemented in Python and is available on Bitbucket as an open source project [71].

Section 9 : Content Injection

In this section we analyze the performance of several content injection strategies in P2P-VoD streaming systems. We work with a broadband network composed of 10,000 active STBs, a content library of 20,000 titles with an average duration of 2h, and an uplink capacity of 1024 Kbps at each peer. Note that each presented result is actually an average of up to 10 simulation runs of 10 hours each. Each run consists of generating VoD requests originating from different peers and following a Zipf-based popularity distribution model over the content library. The Zipf model has been found to be very accurate in capturing the popularity of media content provisioned through different forms (e.g., books, CDs, DVDs, etc.).
By using the Zipf popularity model we try to capture what is usually referred to as the "Long Tail" phenomenon reported by many media retail companies such as Amazon.com and Netflix . This allows us to reproduce a 20/80 Pareto distribution where 80% of the VoD requests are issued for 20% of the most popular titles.

1. General Study: Random algorithm

Figure 23 represents the total number of simultaneous VoD sessions being served throughout the simulation time, which reveals how well the VoD streaming system handles peak hours in terms of VoD service demand. We can observe that the number of simultaneous VoD sessions stays relatively stable, hovering around 2,930 sessions.

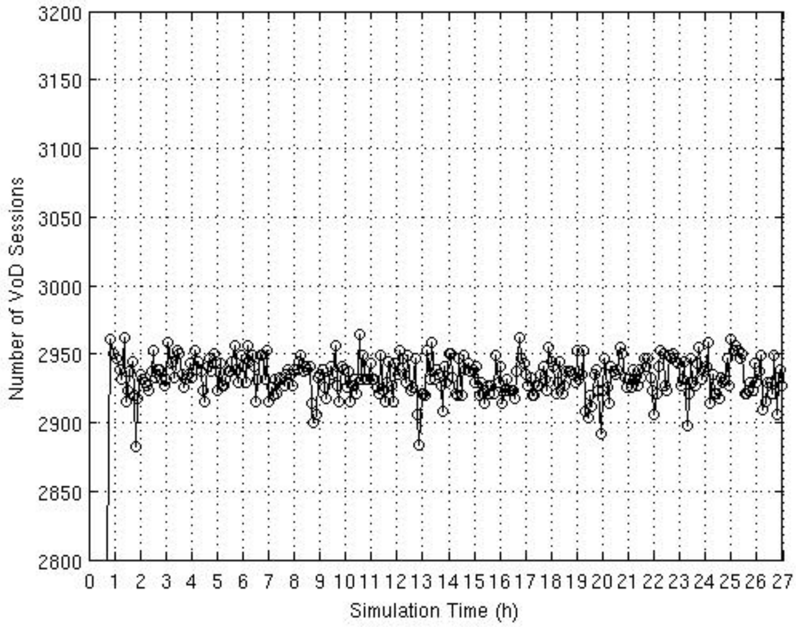


Figure 23: Number of VoD Sessions over time.

Figure 24 represents the participation dispersion. It is shown that, although most of the peers participate in a number of VoD sessions between 80 and 90, there exist some peers which participate in a low number of VoD sessions (between 1 and 60). This result means that some peers are underused and it could explain that the number of simultaneous VoD sessions do not reach its maximum value (3,000).

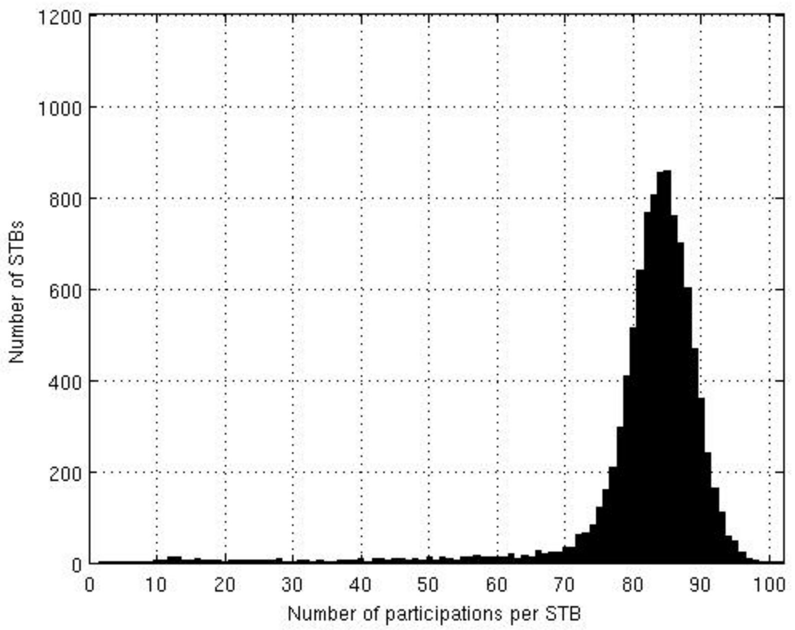


Figure 24: Participations dispertion.

Therefore, at the moment, our goal should be to try to explain why some peers only participate in a low number of VoD sessions. Regarding this concern, Figure 25 represents the peer popularity score associated to each peer. This parameter represents the sum of the relative popularity of all the sub-streams stored in each peer. We can observe that the peers have a popularity score which oscillates between 0.13 and 0.16 for the different nodes. Taking into account that in our simulations the most popular content has a relative popularity of 0.0004, this result means that there are peers which have much more popular contents than others, due to the random placement of the content.

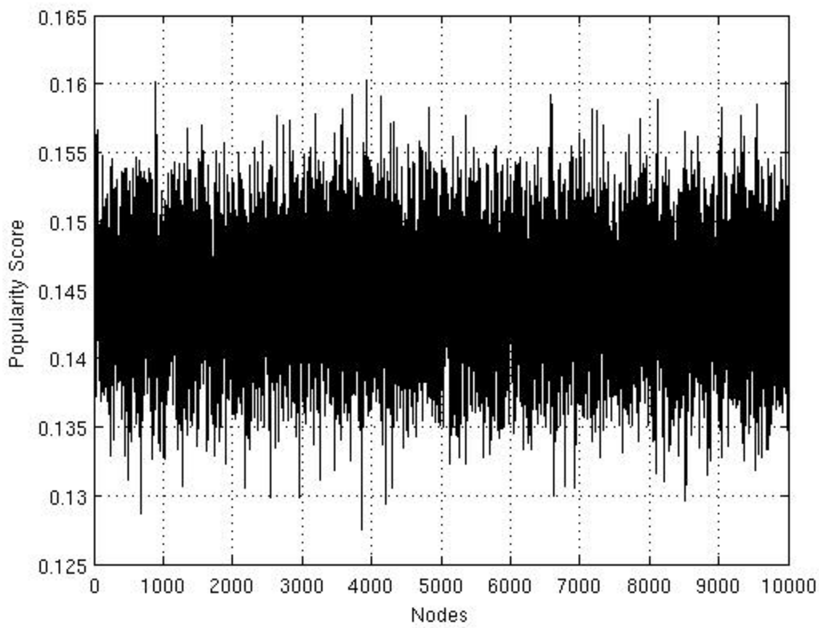


Figure 25: Peer Popularity Score.

Figure 26 represents the average popularity score associated to the peer which participates in a specific number of VoD sessions. It appears is clearly that the peers participating in a low number of sessions have a lower average popularity score than the peer which participate in a higher number of sessions. The reason why some peers are underused is because they do not have very popular contents.

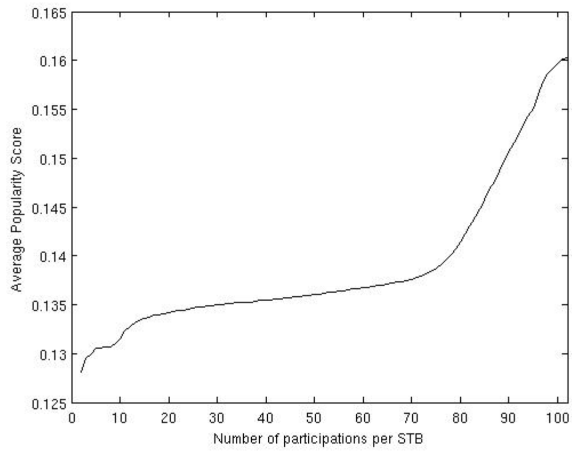


Figure 26: Relation between popularity score and participations dispertion.

2. Popularity-Weighted Content Dispatching (PWCD) Algorithm

Figure 27 illustrates the popularity score figures associated to each peer when the popularity-weighted dispatching algorithm is used. Clearly, this algorithm guarantees that the sum of popularity of all sub-streams contained in a given peer is equivalent to the sum of popularity of all fragments contained in any other peer in the network.

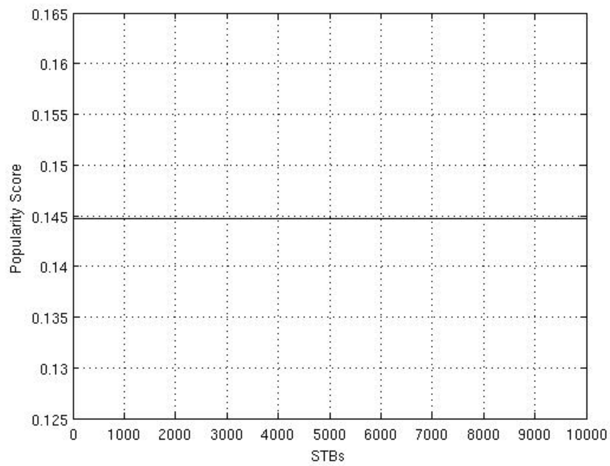


Figure 27: Popularity score for each peer, using the PWD algorithm.

It is expected that the different peers contribute at the same level in the process of delivering VoD services.
Figure 28 represents the participation dispersion. This figure shows that all the peers participate in a number of VoD sessions between 70 and 100, and most of them between 80 and 90. It means than using this

new algorithm there are no underused peers, and therefore the number of simultaneous VoD sessions should have been increased.

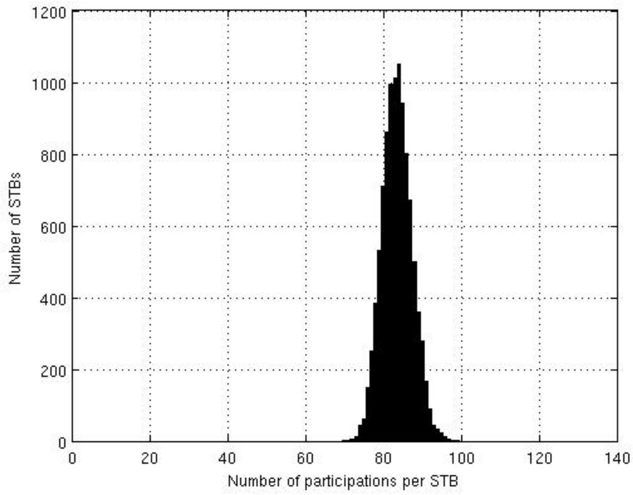


Figure 28: Participations dispersion, using the PWD algorithm.

Figure 29 represents the number of simultaneous VoD sessions obtained using the new algorithm. We can observe than now this parameter has increased from approximately 2,950 sessions to around 3,000 sessions, which was previously presented as the maximum theoretical number of simultaneous VoD sessions. Therefore, this mechanism achieves our goal. However, we can observe in the presented results that it is possible to have more than 3,000 sessions. Before sending a VoD request to the indexing server the peer checks if it has any of the required sub-streams in its local storage. If it has some of them, then the indexing server only tries to establish connections for the rest of sub-streams. It means that sometimes some VoD sessions use less than 10 uplink connections in the network. Therefore, there are more uplink connections available for the new requests, and consequently the network is able to support a bit more than 3,000 simultaneous sessions.

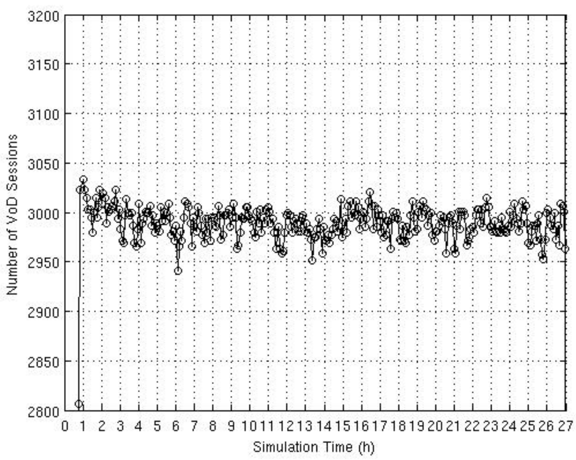


Figure 29: Number of VoD sessions over time.

Section 10 : Resource Allocation

In this section, we evaluate static and dynamic resource allocation approaches in P2P-VoD systems. In order to assess their performances in large Peer-to-Peer deployments, our simulations feature a system with 100,000 peers. Each peer can simultaneously seed up to 5 simultaneous VoD sub-streams to other peers, and can hold up to 500 movie parts, which means the system can store up to 50,000,000 parts.

We use a content library containing 50,000 different video titles. Each title in the library is characterized by a different popularity behavior as illustrated in Figure 30, and the overall popularity distribution over the entire content library follows a Long Tail model modeled with a Zipf law. It is worth recalling that the different titles are duplicated in the network based on their expected popularity. Each duplicate is split into 5 complementary parts before storage in the network. The way the parts are spread in the network was investigated in the literature. It will be fixed throughout the experimental evaluation in order to more closely evaluate the resource allocation strategies.

1. Dynamic demand evolution

Most of the related research works in the literature agree on choosing a Zipf law, as shown in equation (11), for representing the titles popularity in a VoD system. This law can only represent title popularity at a given time, and does not comprehend the evolution of the popularity. Carsten et al. proposed a model based on a time-evolving Zipf distribution, and on the date of apparition of a title in the network. The initial popularity distribution is computed using a Zipf law (11).

$$p_i = \frac{1}{i} * \sum_{k=0}^{M} k \tag{34}$$

where p_i is the popularity of title *i*, and M the number of titles. Then, the evolution of the popularity of each title is represented by a highly granular expression. This popularity model captures popularity shifts, including the effect of adding a new title to a large content library and making it available to users for the first time – a process that is part of the VoD system operation. As new titles are introduced and receive users' attention, the popularity gets redistributed among all titles of the library. Figure 30 illustrates how new titles become increasingly popular following their introduction before progressively receiving less and less requests – this phenomenon has been shown to be the main driver of popularity trend shifts over time in systems with Long Tail–type offerings (e.g., YouTube, Netflix, Amazon Video, etc.). As it can be observed from Figure 30, different titles may command different popularity patterns with different popularity intensity, and different duration of the peak, different popularity declining curve.

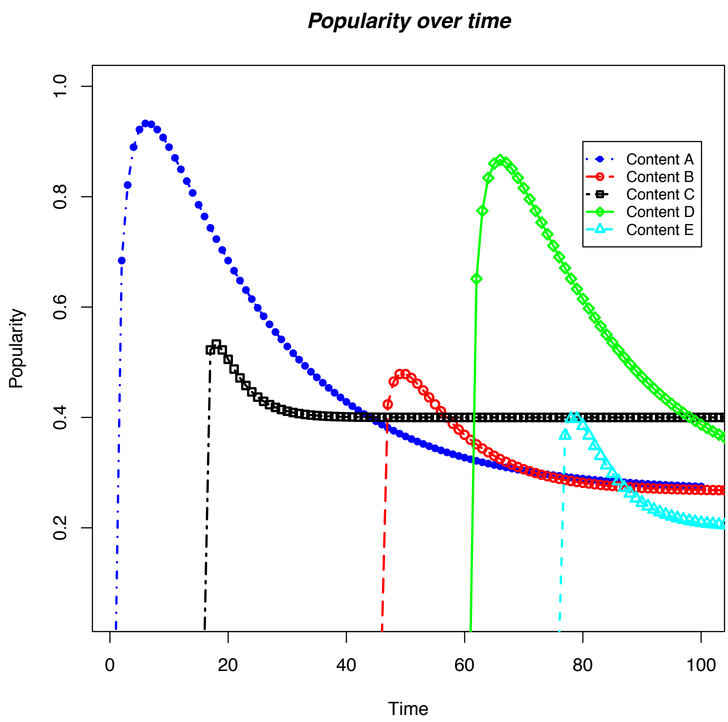


Figure 30: Popularity evolution of multiple titles added at different times in the network.

In our emulations, we use the above-mentioned Carsten et al. work to model the overall content popularity changes. Titles are injected in the network of peers at the beginning of the emulations, but are available in the network only after a period of time, defined for each title. Thus, the content library constantly grows over time.

It is important that the popularity growth and the absolute popularity of new titles are different from a title to another. As illustrated in Figure 30 (Content A-E), while the popularity model for each title is the same, there are still differences in the intensity/sustainability of the initial popularity spike.

Our simulations rely on real content popularity data from a Youtube, a large consumer digital media offering. This data was originally collected from YouTube by Xu Cheng et al., and it is publically available online. Youtube service reach is broad enough and can thus be considered a fairly good proxy for consumers' expected demand for a large VoD library. Furthermore, we integrated the Youtube content popularity traces with Carsten et al.'s approach to content demand evolution in order to better reproduce real-world content popularity shifts.

In our content popularity model, each title has its own popularity fluctuation dynamics that evolve as described above. We believe that the increasing of a real content popularity with an additional popularity variation (at title level) dimension produces a very reasonable content popularity model. The number of VoD requests per hour varies significantly in a typical day, with rises of the demand in the morning, and the evening. Besides the temporal variation in demand intensity, the popularity pattern can vary with different titles receiving more requests than other depending of the time of the day. In order to assess the ability of the system to react in the case of multiple demand scenarios, we vary the intensity of the demand. This intensity evolves every 6 hours, to better reflect the fundamentally different content popularity patterns associated for different times of the day. While the 6-hours figure was obtained from the discussions in *Little et. al*, one should recognize that this a rather arbitrary figure that should be varied by the service provider to meet the particular context of the deployment. In our study, we varied this figure between 2h and 12h, and there was no material impact on the system performance.

2. Resource Allocation Strategies evaluation

In this section, we present the results obtained when using HUF, LCS, LPS, and the LB-RA algorithm.

2.1. Dynamically Selected Strategies by LB-RA

Figure 31 shows the strategies selected over time by LB-RA algorithm based on the Bayesian statistical analysis. HUF and LCS are the only strategies used throughout the emulation time. First, LB-RA starts the first time period with HUF before switching to LPS for the following time period, then uses LCS, and then finally switches back to HF for the last time period.

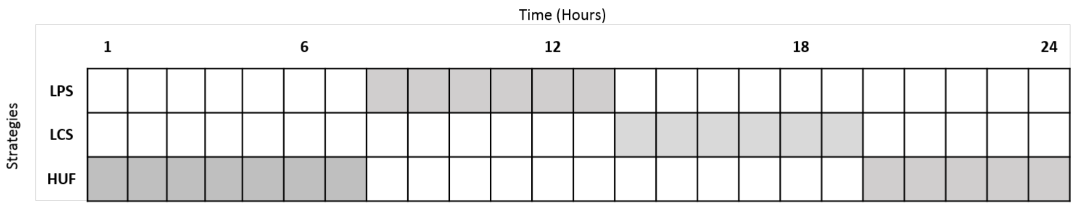


Figure 31: Strategies selected by LB-RA, over time.

It is important to recall that the main drivers of LB-RA strategies selection are: (i) most likely popularity pattern and intensity in the following time period. This is estimated based on the recently observed popularity deviations from the long-run popularity pattern-intensity; (ii) the current content availability in the network which is the consequence of past RA decisions.

2.2. Peer Saturation

Figure 32 presents the number of saturated peers, measured each hour. A saturated peer is a peer that has no more available bandwidth to contribute in any VoD session, meaning the contents contained in a saturated peer become unavailable. Each peer contains certain content parts, and, therefore, some peers are more “critical” than others. It is important to note that a title is considered available only if all of its 5 parts are available from peers in the network.

It is crucial to maximize the number of contents available at any time in order to maximize the number of VoD sessions. Although it’s not a completely inversely proportional relationship between rejection rates and the number of saturated peers, this latter metric is a relatively reliable leading indicator of the VoD rejection rate. Figure 32 shows that LB-RA manages to limit the number of saturated peers by switching between different RA strategies and by maintaining a superior content availability for most of the emulation time (see Figure 33).

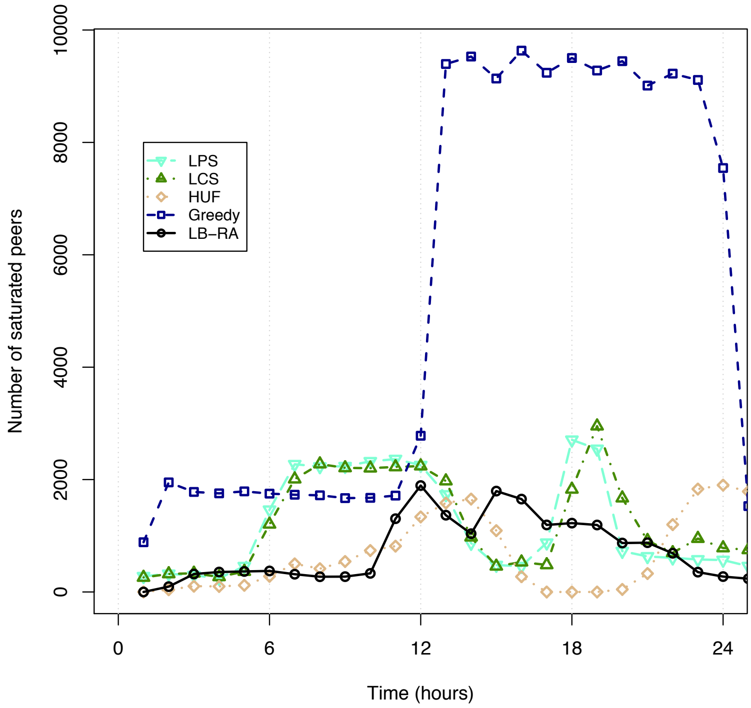


Figure 32: Number of saturated peers over time.

Figure 33 illustrates the content availability in the system throughout the emulation time, and for every RA strategy evaluated. This performance metric essentially tracks how many unique titles are currently available for streaming (i.e., there are bandwidth resources to satisfy a VoD requests for titles). Naturally, at the start of the emulation at t=0h, all 50,000 titles are available. However, as both the popularity intensity and pattern change, the different RA strategies experience different content availability throughout the emulation. Again, one can notice that LB-RA and HUF result in the same content availability evolution for the first time period as LB-RA was effectively using HUF during that period – note that both started with the same full content availability at t=0h.

One can clearly observe from Figure 33 that LB-RA tends to maximize the number of available contents at any time. During the first time period, LB-RA selected HUF, thus ensuring that all contents are available by depleting peers resources in an even manner. In this first time period, network resources are plenty and content availability is high, which means that the ultimate consequences (rejection rate) of resource allocation decisions is not yet felt. LCS and LPS are passive RA strategies so they tend to prioritize the use of the same peers, which lead to their early saturation and subsequent VoD rejections due to reduced content availability. When the high content demand starts to take its tall, LB-RA switches to LPS and thus manages to maintain a content availability lead over other RA strategies.

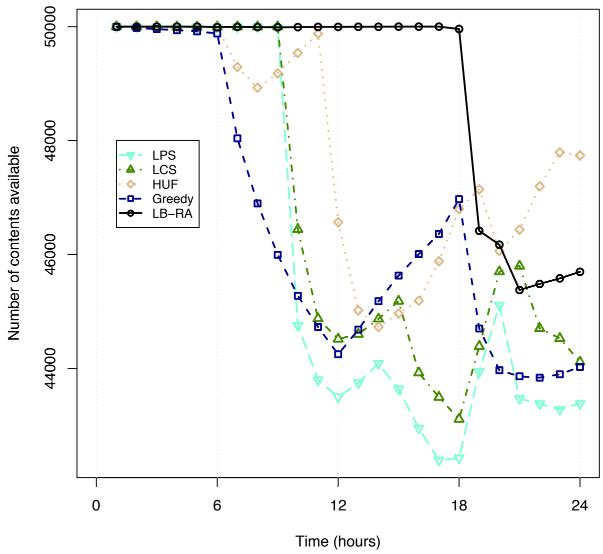


Figure 33: Overall content availability over time (number of unique title available over time).

A final observation here is that the Bayesian approach helps LB-RA to focus on objective statistical outcomes that takes into account current conditions and expected popularity trends, instead of just applying the same rule regardless of the system conditions. For instance, from t=14h to t=18h, one can observe that LCS and LPS maintain a lower number of saturated peers compared to LB-RA. However, this lower number of saturated peers does not translate into a higher content availability, which means that many of the unsaturated peers here contain somehow similar content and thus don't help satisfying the incoming content demand in its diversity.

2.3. Peer Participation

Figure 34 shows how each resource allocation strategy performed in terms of entropy. Obviously both HUF and the Greedy Algorithm outperform all other approaches in terms of balancing the load over the active peers. This is due to the fact that their respective resource allocation algorithms are designed to mainly preserve fairness in terms of peers' bandwidth usage. This ensures that peers resources are consumed "*in unison*", which preserves a high level of content availability as the demand increases.

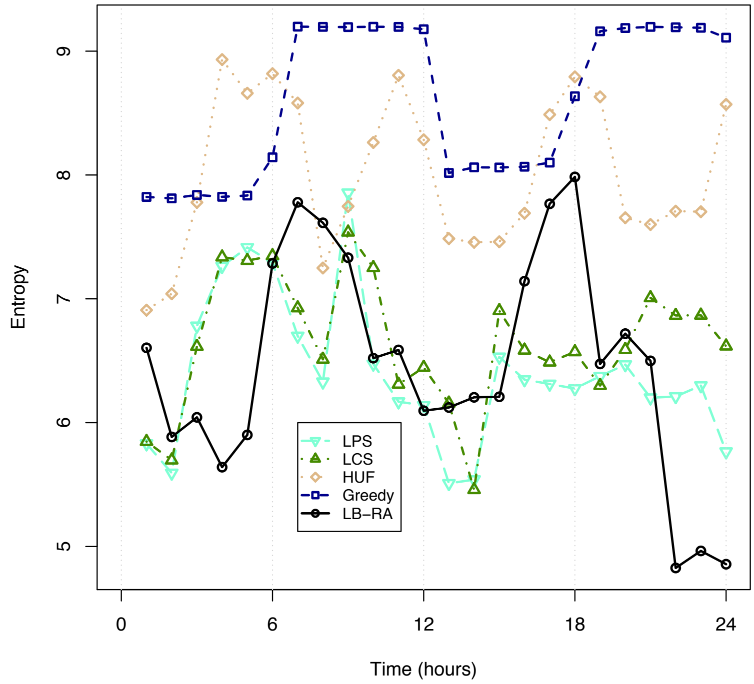


Figure 34 : Evolution of the peer participation entropy, averaged every hour, over time.

Table 1 summarizes the overall peer participation entropy in the system. LB-RA performs similarly to LCS and LPS while HUF and Greedy outperform all other RA strategies. As discussed earlier, those two approaches are designed with a focus on maximize the entropy, so it is not surprising that they outperform all other allocation strategies when it comes to participation level. Therefore, HUF and the Greedy approach perform better than LB-RA, with a higher mean and maximal entropy values.

Table 1 : Entropy evaluation for each strategy.

Strategy	Mean	Max
LCS	6.4	7.9
LPS	6.6	7.5
HUF	8.0	8.9
Greedy	**8.6**	**9.1**
LB-RA	**6.5**	**7.9**

The participation results only deliver one part of the picture as it does not indicate the actual overall P2P network utilization. The participation entropy is indeed an indicator of fairness and how even is the system utilization. It is not an indicator of the overall system utilization, per se. This is rather captured by the VoD requests rejection rate. The VoD requests rejection rate is in many cases more important to VoD services provider. In fact, the entropy metric should be rather used as a guide to improve resource allocation algorithms, while the rejection rate is the ultimate metric to measure the performance of a resource allocation algorithm and the overall yield of the system.

2.4. Rejection Rate

Figure 35 presents the evolution of the rejection rate over time, for each RA strategy. It is important to note that the rejection rate is very negatively correlated with the content availability shown in Figure 33. During the first 6 hours, LB-RA used the same parameters as the HUF strategy, thus resulting in similarly good results. Then, as content demand evolved and exhibited a specific popularity pattern, LB-RA's underlying statistical performance prediction model caused it to switch from HUF to LPS at t=6h. By switching between strategies, LB-RA managed to avoid the rejection peak in the period between t=6h and t=12h, when the demand is at its peak. While other strategies experienced a rejection rate of nearly 60% during peak hours, LB-RA manages to achieve a rejection rate of nearly 0% during this time period. The reason for this outperformance resides in the fact LB-RA enters the peak hour period with a much better network configuration in terms of available resources.

As clearly shown in Figure 35, at t=12h, LB-RA has a higher content availability than any other resource allocation strategy. This situation allows LB-RA to better storm the surge in demand. In other words, LB-RA provisioned the first active sessions using contributing peers that are not the most critical in case of intense content demand. By strategically changing resource allocation strategies from a time period to another, LB-RA will result in a different network configuration in terms of which peers get saturated first and which titles become unavailable. This fundamental difference in terms of network saturation pattern is essentially behind the performance difference in terms of experienced VoD rejection rates. LB-RA's strength resides in its better management of peers' resources (uplink capacity) for the long run.

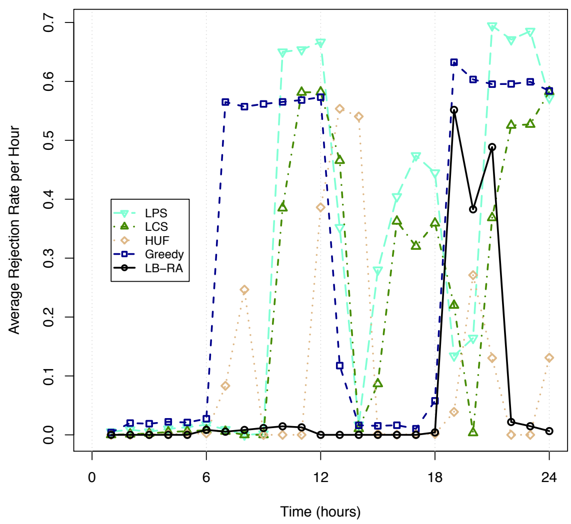


Figure 35: Evolution of the Rejection Rate, averaged every hour, over time.

Although both HUF and LB-RA entered the second time period with relatively similar network resources availability, LB-RA progressively outperformed HUF due to the use of LPS. Clearly, passive resource allocation strategies such as LPS and LCS tend to perform relatively well during peak demand hours as they preserve the most demanded/valuable resources in anticipation of high demand. A critical reason for this outperformance is that both LCS/LPS would assign the same level of priority to peers with somehow similar content, which means there is higher probability that the preserved peers would have complementary content that can be used to satisfy VoD requests. On the other hand, HUF is only concerned with the fairness in terms of peers' bandwidth usage, regardless if the spared peers can be used together to satisfy VoD requests. While HUF depletes peers resources in a fair and arbitrary manner, it does so without regard to what nature of content contained in these peers. This key difference causes passive RA strategies to slightly outperform during extreme demand pressures. Figure 36 shows clearly the gain obtained by using LB-RA in terms of rejection rate.

Between t=12h and t=18h, LCS and LPS present a peak of the VoD requests rejection rate of about 40%, while LB-RA reduces it down to 0% and sustains that level rejection rate till the end to the third time period. LB-RA remarkably keeps the VoD requests rejection rate at a minimum as the low-demand time period starts at t=12h. Note that during the third time period (t=12h to t=18h) LB-RA kept using the LCS strategy. By doing so, LB-RA maintains a higher content availability throughout the third time period. Note that the VoD request rejection rate declines as the content availability increases with the availability of a wider range of titles.

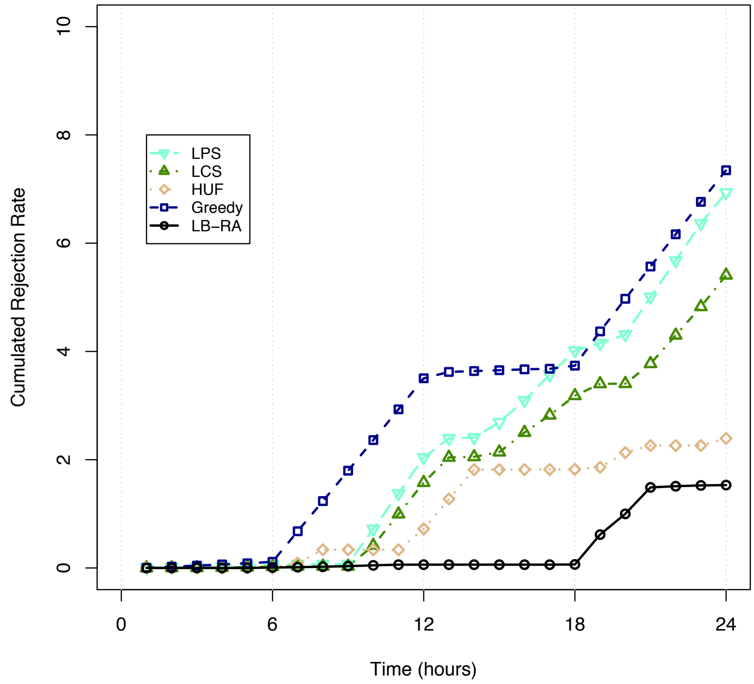


Figure 36: Cumulated Rejection Rate

At t=18h, HUF rejection rate is down to near 0%, while LB-RA reaches a 50% rejection rate (Figure 35). HUF achieves lower minimum rejection rates, because its content availability rises (see Figure 33). In fact, as LB-RA used LCS in the two time periods preceding t=16h, it focused on using the resources of peers with most popular content while protecting the ones with less popular content. Note that over the entire third time period LB-RA still managed to achieve a very low VoD rejection rate (less than 5%).

Finally between t=18h and t=24h, all RA strategies performed relatively in the same manner. An intense content demand combined with more contents appearing in the library resulted in a very high VoD rejection rate across the board.

Table 2 shows the overall performance of each strategy according to the rejection rate metric. By combining the best strategy on each period of time, LB-RA clearly reduced the rejection rate. This helps summarize the performances measured over the simulations. LB-RA shows clear outperformance in terms of total VoD sessions delivered throughout the operation time. As explained earlier, this outperformance is mainly driven by a better management of the inventory of available resources by opportunistically switching to appropriate resource allocation strategy as the demand profile changes. To put the results in context, LB-RA rejected 15,527 less VoD requests than HUF, the second best performer, over only 24 hours. These remarkable improvements represents an opportunity for service providers to increase productivity of their underlying resources, and ultimately generate greater revenues. By combining multiple strategies, and adapting over time, LB-RA managed to reduce the rejection rate efficiently.

Table 2 : Rejection Rate evaluation for each strategy.

Strategy	Mean	Max
LCS	29%	69%
LPS	23%	58%
HUF	10%	55%
Greedy	30%	63%
LB-RA	**6%**	**55%**

3. Dempster-Schafer Theory applied to LB-RA

In this section, we present the results obtained with ARA using the Evidence Theory, labelled "ARA-Evidence". We compare those results to the ones obtained with static strategies, and with the ones obtained with ARA using a Bayesian Approach, labelled "ARA-Bayes".

3.1. Rejection Rate

Figure 37 presents the evolution of the rejection rate over time. It appears clearly that both dynamic strategies greatly reduce the rejection rate. Indeed, when all static strategies reach a higher than 0.5 rejection rate at 13 hours, both dynamic approaches maintain a close to 0 rejection.

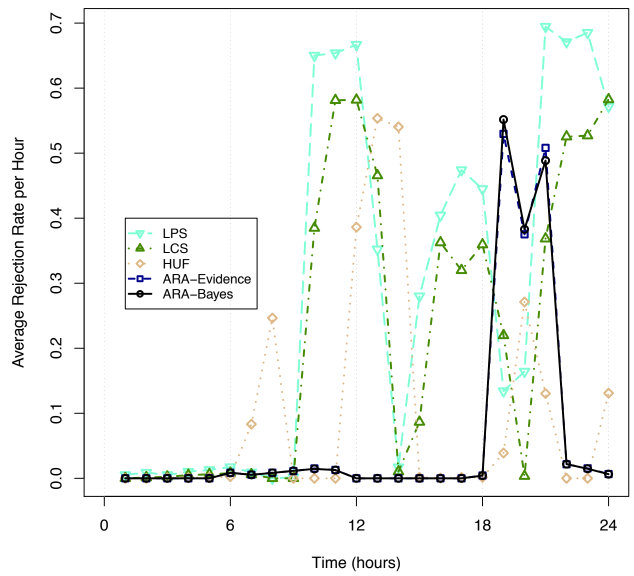


Figure 37: Rejection rate over time for each approach.

This can also be oserved in Figure 38 which displays the total (meaning the sum of all rejection rates, from the start, up to time t) rejection rate over time. All three static RA rise quickly to 2 and higher, while the dynamic approaches manage to stay near 0 rejection, rising up to 1.5 towards the end of the simulation.

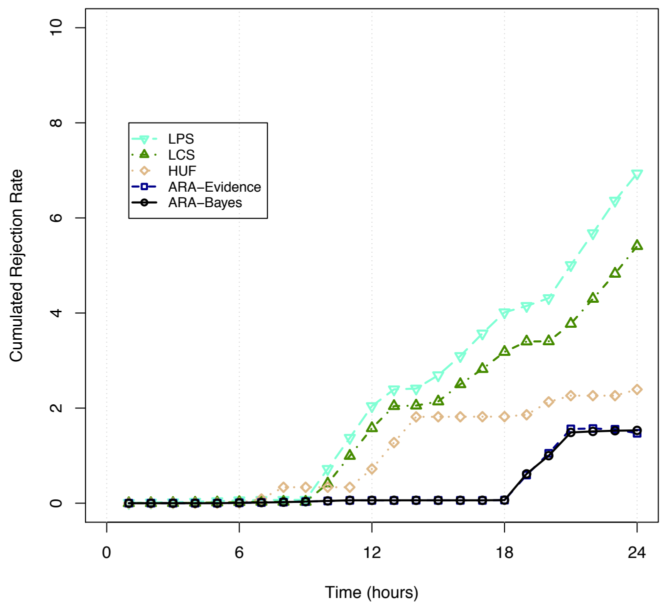


Figure 38: Cumulated Rejection Rate over time for each approach.

3.2. Entropy

Figure 39 presents the evolution of the entropy when using each strategy. Here, HUF is clearly the most efficient. This is due to its design : HUF automatically selects the least participating peers, and, therefore, tends to spread the participations to all peers. On the other hand, the dynamic strategies aim to maintain a low rejection rate, while maintaining a good peer participation entropy.

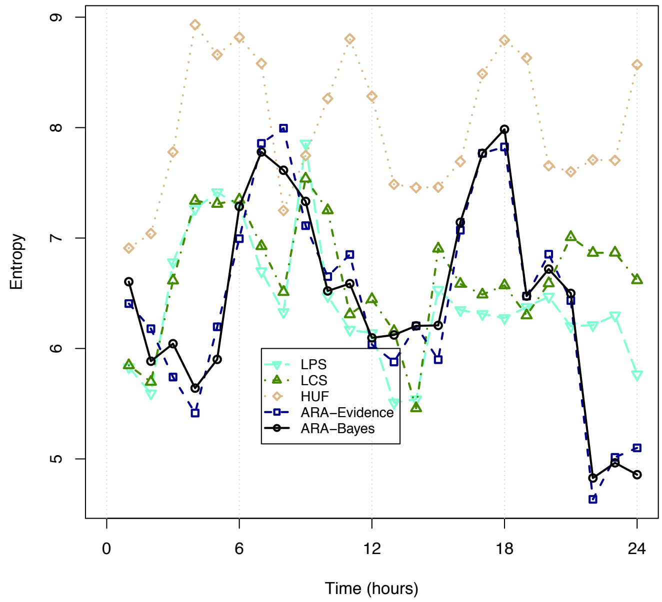


Figure 39: Peer participation entropy evolution over time for each approach.

3.3. Overall results and discussion

Table 3 presents the overall results for all strategies. Both ARA strategies tend to perform similarly, greatly reducing the overall rejection rate.

Table 3 : Overall evaluation for each strategy.

Strategy	Rejection Rate	Entropy
LCS	29%	6.4
LPS	23%	6.6
HUF	10%	8.0
Greedy	30%	**8.6**
LB-RA Bayes	**6%**	**6.5**
LB-RA Evidence	**6%**	**6.5**

Because those two strategies dynamically select the best choices, and switch accordingly, they both obtain very similar results. However, this approach does not require posterior probability estimation, whereas the Bayesian approach is based on it. Therefore, the bayesian approach is a more complex solution to strategy evaluation. By using the evidence theory, we managed to greatly reduce this complexity, while maintaining efficient RA selection.

4. Summary

While LB-RA, using Bayes and Evidence, has outperformed other RA strategies by most metrics, many more fundamental lessons can be drawn from the above performance evaluation and analysis. A key learning is that it is important to let the LB-RA switch strategy more frequently based on popularity predictions and expected statistical performance of the different alternatives. Although LB-RA has shown better performances in most time periods, the level of outperformance tends to weaken or reverse in any given time period, which means that another strategy switch could have helped. Thus, additional RA strategy switching flexibility for LB-RA would be more appropriate in realistic VoD system deployment settings. Furthermore, important content popularity trends shifts tend to last a random time period (not exactly 6h), which further justifies moving LB-RA to a free RA switching policy, i.e., having continuous assessment of future performances.

Another key observation from the experimental results analysis is that the content availability does actually drives the observed VoD requests rejections. Obviously, the resource allocation algorithm play an important role in determining content availability evolution as it allocates specific resources to incoming VoD requests. As seen with LB-RA performances, an element of popularity prediction does actually help maintain a high content availability and thus a low rejection rate. In our performance evaluation, we specifically randomly varied the intensity of popularity and the actual popularity pattern in order reproduce an environment that is as close as possible to real settings. While LB-RA's underlying Bayesian approach will never be able to precisely predict the specifics of future content demand, statistical performance predictions show a significant potential to improve the streaming performances of the system. Although the VoD service provider can further improve RA strategies and add new ones (as discussed below), the very Bayesian approach proposed in this paper will be invaluable in guiding the system for an optimal switching between RA strategies. Finally, the popularity prediction model can also be adjusted and improved to better meet the specific behavior of the VoD users' base.

In this paper, we arbitrarily used three different RA strategies, two passive ones, and one active ones. Our comprehensive performance analysis shows that each one of the RA strategies has its advantages and shortcomings. First, passive RA strategies have the advantage of being content-aware and thus tend to deplete peers' resources in a manner that is more effective during extreme content demand hours. The preserved peers tend to contain complementary content, which would allow the system to provision more VoD sessions when resources are extremely rare. However, passive RA strategies are extremely ineffective at managing system resources during non-peak hours, and tend to shut down a significant part of the peers pool very early on. Second, an active approach like HUF is very effective at maintaining a high content availability before peak hours, but it fails to apply any form of positive discrimination during the peer selection process to rather preserve peers that can be used together. Clearly, our analysis points to the fact that a service provider can certainly devise additional RA strategies to effectively deal with very specific conditions during the system operation. Also, the network and demand specificities should be important inputs in the process of designing new RA strategies. Realistically, a VoD service provider will end up with a number of RA strategies that can cover all possible situations, which further highlights the importance of a proactive strategy like LB-RA.

V – CONCLUSION AND FUTURE WORK

Section 1 – Conclusion

4. Summary of Results and Contributions

The most significant contribution of this work has been to show that hybrid P2P-CDN systems, when using dynamic resource allocation, are reliable for VoD, and provide a scalable alternative to CDN approaches. Most of our research focused uniquely on P2P streaming system - the weak point in a CDN-P2P hybrid architecture -, although the most realistic scenario for using P2P streaming systems in a commercial broadband network would involve the use of CDNs as backup.

In this thesis, we addressed the performances optimization of a P2P-VoD system from three different angles in order to increase system efficiency and ultimately maximize the revenues for the service provider.

Content Dispatching in P2P streaming systems:
First, we thoroughly studied different content fragments dispatching strategies along with their impact on the overall system performances. Two important properties that should be targeted as much as possible when designing content dispatching algorithm are: (i) the peers should be equally popular with the overall cumulated popularity of all fragments contained in any STB[1] (set-top-box) comparable to any other STB – this will favors balancing the streaming load among the different STBs; and (ii) content fragments from any given title should be as spread as possible in the network of STBs in order to improve the overall availability of this title and make it less sensitive to transient STB saturation events.
Taking into account these two performance factors, we present a new dispatching algorithm to equalize the participation of peers in the sessions provided by a content delivery network associated to set-top boxes (peers). To this end, the new algorithm ensures that the fragments of any title are as spread as possible to reduce the probability of having all the peers containing a fragment of a specific title being saturated quickly. In this thesis we show that this new

[1] We use STB and peer interchangeably. The peer in our P2P streaming system will be located in the set-top-box installed at the customers premises by the broadband operator.

algorithm considerably improve the STBs resources during peer demand hours, and it is therefore able to maximize the number of VOD sessions delivered during the peak hours.

Resources Allocation in P2P Streaming Systems:
Second, this thesis presents low-complexity scalable resource allocation strategies that can be deployed to optimize system resources in P2P streaming systems. In fact, with every P2P streaming session served the system use the uplink bandwidth of other peers (STBs) and during peak hours most of peers uplink is saturated which leads to system saturation.

This thesis investigated many passive and active resource allocation strategies and derived new performance metrics to quantify the performance gains and identify the factors behind these gains. At the highest level, it has been found that the best resource allocation strategies are the one that best handle network resources saturations as they gradually build up. More specifically, it is important to design a resource allocation strategy that delays the saturation events as much as possible, while making sure that the streaming load is evenly distributed among STBs at any point of time during the operation phase. A key performance factor for resource allocation strategies is their ability to preserve the system resources needed the most for handling peak demand hours: this requires the resource allocation strategy to be aware of the content popularity cycles and trends.

Learning-Based Resources Allocation (LB-RA) in P2P Streaming Systems:
Third, one of the most important contribution of this thesis is a new approach for a resource allocation strategy in P2P-VoD systems. Instead of designing a new sophisticated RA strategy that can perform relatively well across a wide range situations, LB-RA offers a framework that allows the service provider to combined multiple strategies that can individually outperform in specific conditions. LB-RA uses a Bayesian approach to statistically evaluate the different RA strategies against the most probable content demand changes, and ultimately select the best strategy for the next time cycle. The result is an RA strategy that outperforms in most situations in terms of popularity pattern trends and network saturation levels. LB-RA gives the VoD service providers means to enforce the most effective RA strategy that meets the requirement of their specific situations and their formulated performance objectives.

The Bayesian fusion approach is used to combine together different performance objectives for the resource allocation algorithm, without losing effectiveness. At the core of our RA algorithm is our ability to forecast future content popularity patterns based on observed trends – popularity forecasts are continuously adjusted as more evidence become available. The different RA strategies are evaluated by using these forecasted popularity trends and judged based on their performances in respect to the combined performance objectives.

5. Future Work

In the following, we present some possible scientific directions our future works might explore:

- **Dynamic content re-dispatching** – This thesis proposed a popularity-based approach (PWCD) to inject content in the P2P streaming system. While this content injection algorithm perform relatively well when it comes to balancing the streaming load among all peers and maximizing the system resources, it is not suited for situations where content popularity constantly shifts. Designing a content injection algorithm that dynamically and optimally (minim traffic overhead) responds to popularity shifts in the network.

- **Prediction filters for LB-RA** – The efficiency of the LB-RA approach is in large part dependent on its ability to accurately predict content demand patterns at aggregate level. A possible future research avenue would be to investigate the use of better prediction.

- **Genetic Algorithm for large sets of metrics/strategies** – LB-RA computes the best strategy from a relatively small set of strategies, according to a fusion of a small set of metrics. Admittedly, ideally a service provider would develop a large number of strategies to choose from in order to cover a wide range of possible network conditions. This will require the use of scalable algorithm able to converge within a reasonable time frame. Genetic algorithms can be a good candidate here.

References

[1] Little, T. D C; Venkatesh, D., "Client-server metadata management for the delivery of movies in a video-on-demand system," *Distributed and Networked Environments, 1994. Proceedings., First International Workshop on Services*, p.11-18, 27-28 June 1994

[2] Nossenson, R.; Yudilevich, O.; Markowitz, O., "Client-Server Architecture and Algorithms for Ubiquitous Video Service," *Parallel and Distributed Processing with Applications (ISPA), 2012 IEEE 10th International Symposium on* , p.762-769, 10-13 July 2012

[3] Tobagi, F.A.; Long, J.E., "Client-server challenges for digital video," *Compcon Spring '92. Thirty-Seventh IEEE Computer Society International Conference, Digest of Papers.* , p.88-91, 24-28 Feb. 1992

[4] Apostolopoulos, J.; Wong, T.; Wai-tian Tan; Wee, S., "On multiple description streaming with content delivery networks," *INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE* , vol.3, p.1736-1745, 2002

[5] Carbunar, B.; Pearce, M.; Vasudevan, V.; Needham, M., "Predictive Caching for Video on Demand CDNs," *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE* , p.1-5, 5-9 Dec. 2011

[6] Torres, R.; Finamore, A.; Jin Ryong Kim; Mellia, M.; Munafo, M.M.; Sanjay Rao, "Dissecting Video Server Selection Strategies in the YouTube CDN, "*Distributed Computing Systems (ICDCS), 2011 31st International Conference on* , p.248,257, 20-24 June 2011

[7] Adhikari, V.K.; Yang Guo; Fang Hao; Hilt, V.; Zhi-Li Zhang, "A tale of three CDNs: An active measurement study of Hulu and its CDNs," *Computer Communications Workshops (INFOCOM WKSHPS), 2012 IEEE Conference on* , p.7-12, 25-30 March 2012

[8] Adhikari, V.K.; Yang Guo; Fang Hao; Varvello, M.; Hilt, V.; Steiner, M.; Zhi-Li Zhang, "Unreeling netflix: Understanding and improving multi-CDN movie delivery," *INFOCOM, 2012 Proceedings IEEE* , p.1620,1628, 25-30 March 2012

[9] Moo-Kyung Sung; Chi-Moon Han, "A study on architecture of CDN (Content Delivering Network) with content re-distribution function," *Advanced Communication Technology, 2009. ICACT 2009. 11th International Conference on* , vol.01, p.772,777, 15-18 Feb. 2009

[10] Setton, E.; Baccichet, P.; Girod, B., "Peer-to-Peer Live Multicast: A Video Perspective," *Proceedings of the IEEE* , vol.96, no.1, p.25-38, Jan. 2008

[11] Quaresma, M.; Arrifano, A.M.; Pereira, M.; Freire, M.M.; Antonini, M., "Robust peer-to-peer video streaming based on multiple description coding," *Broadband Multimedia Systems and Broadcasting (BMSB), 2010 IEEE International Symposium on* , p.1-6, 24-26 March 2010

[12] Amoza, F.R.; Rodriguez-Bocca, P.; Romero, P.; Rostagnol, C., "Stability and capacity of peer-to-peer assisted video-on-demand applications," *Ultra*

Modern Telecommunications and Control Systems and Workshops (ICUMT), 2012 4th International Congress on , p.328,335, 3-5 Oct. 2012

[13] Dongmahn Seo; Hanmin Bang; Nansook Heo; Inbum Jung; Yoon Kim, "Experimental Evaluation for Scalability and QoS in a VoD System," *Computational Science and its Applications, 2007. ICCSA 2007. International Conference on* , p.33,38, 26-29 Aug. 2007

[14] Jian-Guang Luo; Qian Zhang; Tang Yun, Y.; Shi-Qiang Yang, "A Trace-Driven Approach to Evaluate the Scalability of P2P-Based Video-on-Demand Service," *Parallel and Distributed Systems, IEEE Transactions on* , vol.20, no.1, p.59,70, Jan. 2009

[15] Almeroth, K.C.; Ammar, M.H., "The use of multicast delivery to provide a scalable and interactive video-on-demand service," *Selected Areas in Communications, IEEE Journal on* , vol.14, no.6, p.1110,1122, Aug 1996

[16] Almeroth, K.C.; Ammar, M.H., "On the performance of a multicast delivery video-on-demand service with discontinuous VCR actions," *Communications, 1995. ICC '95 Seattle, 'Gateway to Globalization', 1995 IEEE International Conference on* , vol.3, p.1631,1635 vol.3, 18-22 Jun 1995

[17] Ying, Lihang and Basu, Anup. pcVoD: Internet Peer-to-Peer Video-On-Demand with Storage Caching on Peers, *Knowledge Systems Institute*, p.218-223, 2005.

[18] Jiayin Qi; Hongli Zhang; ZhenzHou Ji; Liu Yun, "Analyzing BitTorrent Traffic Across Large Network," *Cyberworlds, 2008 International Conference on* , p.759,764, 22-24 Sept. 2008

[19] Rougeau, Brad; Wang, Mea, "Uncover the peer distribution in BitTorrent," *Tsinghua Science and Technology* , vol.17, no.1, p.17,28, Feb. 2012

[20] Kryczka, M.; Cuevas, R.; Guerrero, C.; Azcorra, A.; Cuevas, A., "Measuring the bittorrent ecosystem: Techniques, tips, and tricks," *Communications Magazine, IEEE* , vol.49, no.9, p.144-152, September 2011

[21] Dana, C.; Danjue Li; Harrison, D.; Chuah, Chen-Nee, "BASS: BitTorrent Assisted Streaming System for Video-on-Demand," *Multimedia Signal Processing, 2005 IEEE 7th Workshop on* , p.1-4, Oct. 30 2005-Nov. 2 2005

[22] Guo, Yang, et al. "P2Cast: peer-to-peer patching for video on demand service."*Multimedia Tools and Applications,* vol.33, no.2, p.109-129, 2007

[23] Do, T.T.; Hua, K.A.; Tantaoui, M.A., "P2VoD: providing fault tolerant video-on-demand streaming in peer-to-peer environment," *Communications, 2004 IEEE International Conference on* , vol.3, p.1467-1472 Vol.3, 20-24 June 2004

[24] Minglong Zhang; BoQin Feng, "A P2P VoD system using dynamic priority," *Communications (MICC), 2009 IEEE 9th Malaysia International Conference on* , p.518-523, 15-17 Dec. 2009

[25] Yiu, W.-P.K.; Xing Jin; Chan, S.-H.G., "Distributed Storage to Support User Interactivity in Peer-to-Peer Video Streaming," *Communications, 2006. ICC '06. IEEE International Conference on* , vol.1, p.55,60, June 2006

[26] Harmer, T.J.; McCabe, J.; Donachy, P.; Perrott, R.H.; Chambers, C.; Craig, S.; Lewis, R.; Mallon, B.; Sluman, L., "Gridcast: a grid and Web service broadcast infrastructure," *Services Computing, 2005 IEEE International Conference on* , vol.1, p.35-42, 11-15 July 2005

[27] Dr. Terry Harmer, "Gridcast - a Next Generation Broadcasting Infrastructure?," *Challenges of Large Applications in Distributed Environments, 2006 IEEE* , p.1-2, 19 June 2006

[28] Wenbin Jiang; Chong Huang; Hai Jin; Xiaofei Liao, "A New Proxy Scheme for Large-Scale P2P VoD System," *Embedded and Ubiquitous Computing, 2008. EUC '08. IEEE/IFIP International Conference on* , vol.1, p.512,518, 17-20 Dec. 2008

[29] Xiaofei Liao; Hai Jin, "OCTOPUS: A Hybrid Scheduling Strategy for P2P VoD Services," *Grid and Cooperative Computing, 2007. GCC 2007. Sixth International Conference on* , p.26,33, 16-18 Aug. 2007

[30] Hefeeda, Mohamed, et al. "PROMISE: peer-to-peer media streaming using CollectCast." *Proceedings of the eleventh ACM international conference on Multimedia*. ACM, 15-18 Aug. 2003, Berkeley

[31] Habib, A.; Chuang, J., "Incentive mechanism for peer-to-peer media streaming," *Quality of Service, 2004. IWQOS 2004. Twelfth IEEE International Workshop on* , p.171,180, 7-9 June 2004

[32] Yin, Hao, et al. "Livesky: Enhancing cdn with p2p." *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMCCAP)* vol.6, no.3, p.16-18, 2010.

[33] Yin, Hao, et al. "Design and deployment of a hybrid CDN-P2P system for live video streaming: experiences with LiveSky." *Proceedings of the 17th ACM international conference on Multimedia*, Oct. 19-24, 2009, Beijing.

[34] L. Zou, E. Zegura, M. Ammar, The effect of peer selection and buffering strategies on the performance of peer-to-peer file sharing systems. In *Proceedings of the tenth IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunications Systems* (MASCOTS '02), p. 63-70, August 07-09, 2002, Arlington.

[35] Satsiou, A.; Tassiulas, L., "Reputation-Based Resource Allocation in P2P Systems of Rational Users," *Parallel and Distributed Systems, IEEE Transactions on* , vol.21, no.4, p.466,479, April 2010

[36] Iosifidis, G.; Koutsopoulos, I., "Reputation-Assisted Utility Maximization Algorithms for Peer-to-Peer Networks," *Quality of Service, 2008. IWQoS 2008. 16th International Workshop on* , p.20,29, 2-4 June 2008

[37] Lei Yang; Zhiguang Qin; Hao Yang; Can Wang; ChaoSheng Feng, "A reputation-based method for controlling free-ride in P2P networks," *Communications, Circuits and Systems (ICCCAS), 2010 International Conference on* , p.249,253, 28-30 July 2010

[38] Papaioannou, T.G.; Stamoulis, G.D., "Effective use of reputation in peer-to-peer environments," *Cluster Computing and the Grid, 2004. CCGrid 2004. IEEE International Symposium on* , p.259,268, 19-22 April 2004

[39] Seungwon Shin; Guofei Gu; Reddy, N.; Lee, C.P., "A Large-Scale Empirical Study of Conficker," *Information Forensics and Security, IEEE Transactions on* , vol.7, no.2, p.676,690, April 2012

[40] Li Xiong; Ling Liu, "PeerTrust: supporting reputation-based trust for peer-to-peer electronic communities," *Knowledge and Data Engineering, IEEE Transactions on* , vol.16, no.7, p.843,857, July 2004

[41] Haiying Shen; Yingwu Zhu, "FairTrust: Toward secure and high performance P2P networks," *Parallel and Distributed Systems, 2007 International Conference on* , vol.2, p.1,8, 5-7 Dec. 2007, Hsinchu

[42] S. Koo, L. George, K. Karthik, "A genetic-algorithm-based neighbor-selection strategy for hybrid peer-to-peer networks," *Proceedings of the thirteenth International Conference on Computer Communications and Networks* (ICCCN '04), p. 469-474, October 11-13, 2004, Chicago

[43] M. Fouda, Z. Fadlullah, M. Guizani, N. Kato, A Novel, "P2P-VoD Streaming Technique Integrating Localization and Congestion Awareness Strategies," In *Mobile Networks and Applications,* vol. 17, no. 5, p. 594-603, 2012.

[44] Y. Huang, T. Fu, D. Chiu, J. Lui, and C. Huang, "Challenges, design and analysis of a large-scale p2p-vod system", *Proceedings of the ACM SIGCOMM 2008 conference on Data communication* (SIGCOMM '08), p.375-388, 17-22 Aug., 2008, New York

[45] A. Nafaa, S. Murphy, L. Murphy, "Analysis of a large-scale VoD architecture for broadband operators: a P2P-based solution," *IEEE Communications Magazine,* vol.46, no.12, p.47-55, 2008.

[46] L. Rong, I. Burnett, "BitTorrent in a dynamic resource adapting peer-to-peer network," *Proceedings of the first International Conference on Automated Production of Cross Media Content for Multi-Channel Distribution* (AXMEDIS '05), p.1-4, November 30-December 02, 2005, Florence

[47] Suh, Kyoungwon, et al. "Push-to-peer video-on-demand system: Design and evaluation." *Selected Areas in Communications, IEEE Journal on*, vol.25. no.9 p.706-1716, 2007

[48] Raftopoulou, Paraskevi. "Fair resource allocation in P2P systems: theoretical and experimental results," *Doctoral Dissertation, Technical University of Crete*, 2003.

[49] C. Shannon, W. Weaver, "A mathematical theory of communication," *Bell System Technical Journa*l, vol. 27, p. 379-423, 1948.

[50] J. Gomes, P. Inacio, M. Freire, M. Pereira, P. Monteiro, "Analysis of peer-to-peer traffic using a behavioural method based on entropy," *Proceedings of the International Conference on Performance, Computing and Communications (IPCCC '08)*, p. 201-208, Dec. 07-09 2008, Austin

[51] Gramatikov, Sasho, et al. "P2P Assisted Streaming for Low Popularity VoD," *ICT Innovations*, vol.207, p.23-33, 2013

[52] Yang, Mao, et al. "An empirical study of free-riding behavior in the Maze P2P file-sharing system.", Springer *Peer-to-Peer Systems*, vol. 4, p. 182-192, 2005

[53] Yu, Hongliang, et al. "Understanding user behavior in large-scale video-on-demand systems," *ACM SIGOPS Operating Systems Review*, vol. 40., no. 4., p.333-344, 2006

[54] Dan, A., Sitaram, D., Sahababuddin, P. "Scheduling policies for an on-demand video server with batching," *Proc. of ACM Multimedia*, p.15-23, Oct. 1994

[55] Breslau, L., Cao, P., Fan, L., Phillips, G., Shenker, S. "Web caching and zipf-like distributions: Evidence and implications," *Proc. of INFOCOM, vol.1, p.126-134,* 17-19 March 1999, New York

[56] Acharya, S., Smith, B., Parne, P. "Characterizing user access to videos on the world wide web," *Proc. of ACM/SPIE Multimedia Computing and Networking,* 30-31 Jan., 2000, San Jose

[57] Cherkasova L., Gupta, M. "Characterizing locality, evolution, and life span of accesses in enterprise media server workloads," *Proc. of NOSSDAV*, 12-14 May 2002, Miami

[58] Yoshihiro K., et al., "RFC3016: RTP payload format for MPEG-4 Audio/Visual streams", *Internet Engineering Task Force*, Nov. 2000.

[59] Collette, Y., Siarry, P.,. "Multiobjective optimization : Principles and Case studies", *Springer*, 2004.

[60] El-Ghazali, Talbi.. "Metaheuristics: from design to implementation," *Jonh Wiley and Sons Publishers., Chichester*, 2009.

[61] Dréo, J., Pétrowski, A., Siarry, P., Taillard, E., "Metaheuristics for hard optimization", *Springer Publishers*, 2006.

[62] Boychuk, L.M. Ovchinnikov, V.O., "Principal Methods of Solution Multicriteria Optimization Problems (survey)", *Soviet Automatic Control*, vol. 6, p. 1-4, 1973

[63] M. Hamada et al, " A fully Bayesian approach for combining multilevel failure information in fault tree quantification and optimal follow-on resource allocation", *Reliability Engineering & System Safety*, vol. 86, no. 3, p. 297-305, 2004

[64] Cao, R. B.. "Web Caching and Zipf-like Distributions: Evidence and Implication," *Proc. IEEE INFOCOM: INFOCOMM*, vol. 1, p.126-134, 1999, New York

[65] Carsten Griwodz, and al., "Long-term movie popularity models in video-on-demand systems: or the life of an on-demand movie," *Proceedings of the fifth ACM international conference on Multimedia MULTIMEDIA '97*, p.349-357, Nov. 9-13, 1997, Seattle

[66] Meeyoung Cha and al., "I tube, you tube, everybody tubes: analyzing the world's largest user generated title video system," *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement (IMC '07)*, Oct. 24-26 2007, San Diego

[67] Avramova and al. (2009), "Analysis and Modeling of Video Popularity Evolution in Various Online Video Title Systems: Power-Law versus Exponential

Decay," *First International Conference on Evolving Internet. INTERNET '09*, p.95-100, 2009, Washington.

[68] Chris Anderson, "The Long Tail: Why the Future of Business Is Selling Less of More", *New York: Hyperion*, 2006

[69] Walley, Peter. "Measures of uncertainty in expert systems.", *Artificial intelligence, vol.*83, no.1, p.1-58, 1996

[70] Baraldi, Piero, Irina Crenguta Popescu, and Enrico Zio, "Methods of Uncertainty Analysis in prognostics," *International Journal of Performability Engineering*, vol.6, no.4, p.303-330, 2010

[71] SNAPE simulator, https://bitbucket.org/trohmer/snape