Universidad Politécnica de Madrid

Escuela Técnica Superior de Ingenieros de Telecomunicación

Architectures for Video-on-Demand content distribution in managed networks

Tesis Doctoral

Sasho Gramatikov

Master in electrical engineering

2013

Departamento de Señales, Sistemas y Radiocomunicaciones

Escuela Técnica Superior de Ingenieros de Telecomunicación

Architectures for Video-on-Demand content distribution in managed networks

Autor:

Sasho Gramatikov

Master in electrical engineering

Director:

Fernando Jaureguizar Núñez

Doctor Ingeniero de Telecomunicación

2013

Tesis Doctoral

Architectures for Video-on-Demand content distribution in managed networks

Autor: Sasho Gramatikov

Director: Fernando Jaureguizar Núñez

Tribunal nombrado por el Mfgco. y Excmo. Sr. Rector de la Universidad Politécnica de Madrid, el día ..... de ..... de 2013.

Presidente: D.....

Vocal: D.....

Vocal: D.....

Vocal: D.....

Secretario: D.....

Realizado el acto de defensa y lectura de la tesis el día ..... de ..... de 2013

en .....

Calificación .....

EL PRESIDENTE LOS VOCALES

EL SECRETARIO

"It would take an individual over 5 million years to watch the amount of video that will cross global IP networks each month in 2017. Every second, nearly a million minutes of video content will cross the network in 2017."

Cisco Visual Networking Index: Forecast and Methodology 2012-2017

To my parents, to my brother.

Acknowledgments

This PhD thesis and the years of research dedicated to it has formed an important part of my life. Apart from the professional experience, it also brought to me priceless life experience by giving me the opportunity to meet a lot of people who contributed to its realization and to whom I am especially thankful.

First of all, I would like to thank my supervisor, Professor Fernando Jaureguizar, for all his dedication to my work, for his great efforts to get the things right in the right time, for all the read and corrected papers and for being more than a supervisor during my stay in Spain. I would also like to give my sincere gratitude to Professor Narciso García for giving me the possibility to become part of the *Grupo de Tratamiento de Imágnes* group, for being always helpful and supportive. Thanks to the professors Francisco, Julián and Luís.

I would like to thank to the *Universidad Politécnica de Madrid* for providing me a scholarship for my PhD studies and stay in Spain.

For all the nice time spent during the working hours, the coffee breaks, and out of the working time, I want to give my thankfulness to the most cheerful and friendliest group of people ever: Carlos Roberto, Raúl, Massimo, Filippo, Gianluca, César, Jesús, Maykel, Esther and the rest of the members of GTI. Thanks for your positive energy, for your friendship and for making enjoyable every single moment I spent with you.

Many thanks for the friendship and the nice moments spent during my studies to Perun, Katica, Ana, Irena and all the people who were part of my life in Spain.

I would like to give my sincere gratitude to my friend Ivan for his exceptional support, positive attitude and encouraging advices in the most difficult moments. Thanks to my friends in Macedonia: Sanja, Vlatko, Aleksandra, Verche, Nikola, Dushko, Kostadin, Nadica, Aneta, Hari ...

I would especially like to thank my colleagues and friends Sonja and Igor for the support and for the useful talks that helped me to get the right direction towards the goal.

Finally, I would like to thank to those who mean a lot to me, those who had to share all my hard moments, and the bright ones, those who are the guidance in my life, my support. Thanks to my parents Ivan and Mirjana, and my brother Vladimir. I would also like to thank my brother's wife Snezhana and especially to their son Stefan for his ability to pass all his joy and happiness to me.

Resumen

La demanda de contenidos de vídeo ha aumentado rápidamente en los últimos años como resultado del gran despliegue de la TV sobre IP (IPTV) y la variedad de servicios ofrecidos por los operadores de red. Uno de los servicios que se ha vuelto especialmente atractivo para los clientes es el vídeo bajo demanda (VoD) en tiempo real, ya que ofrece una transmisión (streaming) inmediata de gran variedad de contenidos de vídeo. El precio que los operadores tienen que pagar por este servicio es el aumento del tráfico en las redes, que están cada vez más congestionadas debido a la mayor demanda de contenidos de VoD y al aumento de la calidad de los propios contenidos de vídeo. Así, uno de los principales objetivos de esta tesis es encontrar soluciones que reduzcan el tráfico en el núcleo de la red, manteniendo la calidad del servicio en el nivel adecuado y reduciendo el coste del tráfico.

La tesis propone un sistema jerárquico de servidores de streaming en el que se ejecuta un algoritmo para la ubicación óptima de los contenidos de acuerdo con el comportamiento de los usuarios y el estado de la red. Debido a que cualquier algoritmo óptimo de distribución de contenidos alcanza un límite en el que no se puede llegar a nuevas mejoras, la inclusión de los propios clientes del servicio (los peers) en el proceso de streaming puede reducir aún más el tráfico de red. Este proceso se logra aprovechando el control que el operador tiene en las redes de gestión privada sobre los equipos receptores (Set-Top Box) ubicados en las instalaciones de los clientes. El operador se reserva cierta capacidad de almacenamiento y streaming de los peers para almacenar los contenidos de vídeo y para transmitirlos a otros clientes con el fin de aliviar a los servidores de streaming.

Debido a la incapacidad de los peers para sustituir completamente a los servidores de streaming, la tesis propone un sistema de streaming asistido por peers. Algunas de las cuestiones importantes que se abordan en la tesis son saber cómo los parámetros del sistema y las distintas distribuciones de los contenidos de vídeo en los peers afectan al rendimiento general del sistema. Para dar respuesta a estas preguntas, la tesis propone un modelo estocástico preciso y flexible que tiene en cuenta parámetros como las capacidades de enlace de subida y de almacenamiento de los peers, el número de peers, el tamaño de la biblioteca de contenidos de vídeo, el tamaño de los contenidos y el esquema de distribución de contenidos para estimar los beneficios del streaming asistido por los peers. El trabajo también propone una versión extendida del modelo matemático mediante la inclusión de la probabilidad de fallo de los peers y su tiempo de recuperación en el conjunto de parámetros del modelo. Estos modelos se utilizan como una herramienta para la realización de exhaustivos análisis del sistema de streaming de VoD asistido por los peers para la amplia gama de parámetros definidos en los modelos.

Abstract

The demand of video contents has rapidly increased in the past years as a result of the wide deployment of IPTV and the variety of services offered by the network operators. One of the services that has especially become attractive to the customers is real-time Video on Demand (VoD) because it offers an immediate streaming of a large variety of video contents. The price that the operators have to pay for this convenience is the increased traffic in the networks, which are becoming more congested due to the higher demand for VoD contents and the increased quality of the videos. Therefore, one of the main objectives of this thesis is finding solutions that would reduce the traffic in the core of the network, keeping the quality of service on satisfactory level and reducing the traffic cost.

The thesis proposes a system of hierarchical structure of streaming servers that runs an algorithm for optimal placement of the contents according to the users' behavior and the state of the network. Since any algorithm for optimal content distribution reaches a limit upon which no further improvements can be made, including service customers themselves (the peers) in the streaming process can further reduce the network traffic. This process is achieved by taking advantage of the control that the operator has in the privately managed networks over the Set-Top Boxes placed at the clients' premises. The operator reserves certain storage and streaming capacity on the peers to store the video contents and to stream them to the other clients in order to alleviate the streaming servers.

Because of the inability of the peers to completely substitute the streaming servers, the thesis proposes a system for peer-assisted streaming. Some of the important questions addressed in the thesis are how the system parameters and the various distributions of the video contents on the peers would impact the overall system performance. In order to give answers to these questions, the thesis proposes a precise and flexible stochastic model that takes into consideration parameters like uplink and storage capacity of the peers, number of peers, size of the video content library, size of contents and content distribution scheme to estimate the benefits of the peer-assisted streaming. The work also proposes an extended version of the mathematical model by including the failure probability of the peers and their recovery time in the set of parameters. These models are used as tools for conducting thorough analyses of the peer-assisted system for VoD streaming for the wide range of defined parameters.

Contents

List of Figures

List of Tables

Nomenclature

ACM	Automatic Content Movement
ADSL	Asymmetric Digital Subscriber Line
BS	Branch Server
CDN	Content Delivery Network
CO	Central Office
CR	Central Repository
CS	Central Server
DHT	Distributed Hash Tables
DSLAM	Digital Subscriber Line Access Multiplexer
ES	Edge Server
HD	High Definition
IPTV	Internet Protocol Television
NBPR	Network-Based Personal Recording
P2P	Peer-to-Peer
QoS	Quality of Service
RNG	Random Number Generator
SD	Standard Definition
SHE	Super Head End
SS	Service Selection

STB	Set-Top Box
TSTV	Time Shifted TV
VHO	Video Hub Office
VoD	Video-on-Demand
VSO	Video Source Office
ZM	Zipf-Mandelbrot

Chapter 1

Introduction

From its very beginning, the TV has been constantly evolving in the technology it uses and the services it offers. The TV, as we know, started as a video service, free of charge which consisted in a broadcast of analog video signal through radio waves. All users were equal and everybody could see the contents that were sent by the provider in strictly defined schedule. The users had the freedom to choose the channels, but not the contents they watched. It then evolved to cable TV which digitized the contents and connected the users in a network that enabled both video and data delivery. This service was not free of charge because it required a more expensive infrastructure for bringing it to the homes of the customers. The price that the clients were paying was justified by the provided higher quality of the videos and the higher variety of offered channels. The rapid advances of the network topologies brought the TV to the managed IP networks and introduced the Internet Protocol TV (IPTV). Instead of simultaneous broadcast of all the channels, the IPTV took advantage of the key concept of IP networking, the *multicast*, to enable simultaneous delivery of the same video stream to multiple users in the IP network. The new IPTV technology required a special device that would reside in the users' homes, called Set-Top Box (STB) with capability of converting the IP packet stream into an uninterrupted high quality video stream and storing some of the videos. The new technology also meant higher expectation from the clients, and therefore, apart from the higher quality of the videos, the IPTV providers started to introduce new, more personalized TV services, such as the Network-Based Personal Recording (NBPR) which enabled scheduled recording of a favorite program within a time period chosen by the client and, later, the Time Shifted TV (TSTV) which enabled the users to watch the past contents of any of the channels within a range of few days. These services, however, required certain storage buffer on the streaming servers and a dedicated *unicast* stream for every request, which significantly increased the traffic in the networks.

The growing popularity of these services and the wide spread of the IPTV made a solid

ground for the introduction of a real-time Video-on-Demand (VoD) service, which marked the entering in the ultimate stage of evolution of the personalized video experience. Unlike the NBPR and TSTV, which offered only the contents from the TV channels, the VoD gave the clients the freedom to choose the contents they want to watch from a large video library. The fact that it enabled them to watch any video at any time made it a significantly popular service which has been continuously gaining on number of users.

All these personalized services, however, require extra storage space of the streaming servers, and also a dedicated unicast flow of data for every request, which significantly increases the traffic in the network. The tendency of rapid growth of the popularity of the VoD, its high bandwidth demand and the clients' appetite for higher-quality videos has congested the networks and made them a weak point in the realization of the service. The operators are also faced with high storage demands of the streaming servers for storage of the growing number of video items. Therefore, the network architecture and the strategies for placement of the video contents have become a considerable challenge for many telecommunication operators. This challenge is the main motivation for the research work presented in this text.

Considering that the IPTV network infrastructure consists of hierarchy of several levels of servers, one of the most important goals is to place the VoD contents in such a manner that the overall traffic in the network is minimized, and at the same time, the traffic required from the servers is balanced. The main criterion for this process is the popularity of the contents which is measured according to the traffic generated in the network for each of the videos. It is desired to place the more frequently requested contents closer to the clients, but under the constraints of the limited storage and streaming capacity of the servers. Another challenge is the dynamic nature of the contents which tends to change their popularity in time. An optimal distribution at one moment might not be the best choice in other moment because each video has its own popularity life cycle. The system also has to adjust to the users' intensity of requests. Therefore, designing an automated IPTV-based system for distribution of VoD contents that would respond to all these requirements is one of the goals of this work.

Despite the contribution of the managed networks to the reduction of the load in the system, any of the solutions comes to a point of no further improvement because of the limited resources of the system. No matter how well the contents might be distributed, when the streaming and storage capacities get completely exploited, the system cannot perform any better. Therefore, another level of improvement of the system's performance is including the clients in the process of distribution of the contents by taking advantage of their streaming and storage capacity. This concept is already widely spread and well known in the whole Internet community as P2P (Peer-to-Peer) and enables sharing files between peers connected in a network, where all thee peers that have the content participate in the distribution, without any help of dedicated servers. Although it has shown tremendous success in file

sharing as a result of the non real-time character of the contents and the free of charge shared resources, its main disadvantage in the delivery of real-time video streams is the poor reliability, which is largely dependent on the will of the peers to cooperate, and the reliability of the Internet. Therefore, the quality of service for delivery of VoD contents in real-time cannot be guaranteed in such environment.

These issues, however, are overcome in the IPTV managed networks because the VoD service provider can configure its own network capacity according to the number of the subscribed users and can provide reliable connections between the participants. Moreover, the IPTV provider can reduce the human factor of participation in the sharing of the video contents because it has the possession of the STBs and can easily reserve part of their storage and uplink capacity for streaming purposes. Another convenience of the STBs is that, with the falling tendency of the prices of the storage, they are becoming cheap storage devices. The storage capacity is used by the VoD service provider to place more videos and to increase the availability of the contents on the peers. These facts make the IPTV networks a suitable environment for implementation of P2P for distribution of video contents. The only disadvantage of the IPTV networks is that with the current Asymmetric Digital Subscriber Line (ADSL) technology used for the last mile in most of the networks, the portion of the bandwidth dedicated to the uplink is not enough for streaming an entire video content. Therefore, in the system that will be considered in this work, the streaming servers will remain the foundation of the VoD service that will provide the required quality of service, and the peers will be used only to alleviate their load and the overall load of the network. Since the peers only partially participate in the streaming process, the service is called peer-assisted VoD streaming.

The implementation of such a hybrid system, which employs both the classical client-server streaming and the P2Pstreaming, rises many questions that are of great interest for the providers of the VoD service. The most important questions are how the peer-assisted streaming will improve the service and to what scale it will reduce the traffic in the network. The answers to these questions are not simple because there is a large range of parameters that have a direct influence on the portion of the traffic streamed by the peers. Some of these parameters are the uplink capacity of the STBs designated for streaming purposes, the storage capacity of the STBs, the intensity of the requests for VoD contents, the number of peers, the size of the contents, the size of the offered video library, the popularity of the contents, the streaming capacity of the servers, the storage capacity of the servers and other network based parameters. Assuming that the operator implements an optimal distribution of the contents on the servers, it is particularly important to know how the distribution of the contents on the peers will contribute to improvements of the performances of the system.

Although the VoD service provider can reserve resources for the streaming process, it cannot provide full reliability of the peers. The clients might, intentionally or unintentionally, turn

off their STBs after watching the video that they have required. This action of the clients, not only excludes their resources from the sharing community, but also interrupts the streams that they were distributing before their failure. Thus, the failures of the peers in the system give a more realistic picture of the VoD service, but it also introduces extra complexity because the servers have to compensate all the streams interrupted by the failures. This additional feature raises yet another series of questions like, how will the failure-related parameters influence the portion of the traffic streamed by the peers or what would be the additional amount of traffic that would be redirected to the servers for compensating the interrupted streams and the reduction of the traffic on the servers due to the reduced number of peers in the system? The answers of these questions depend largely on the probability that the clients will turn their STB off and the time that they will spend until they turn it on again.

The objective of the thesis is to propose different models for streaming VoD contents that would consist of hierarchically organized streaming servers and peers that would take part in the streaming process. Using the models for VoD streaming, some of the goals are to find a strategy that would optimally place the contents in the servers according to the request pattern and the state of the systems, so that it is obtained improved quality of service and reduced traffic costs. Other objectives are to find how the distribution of the contents on the peers helps to alleviate the streaming servers and to further reduce the traffic price and improve the quality of service. The key objective of this work is to develop a mathematical tool that would model the proposed system for peer-assisted VoD streaming and that would be used for a thorough analysis of the system for a wide range of parameters.

The structure of the thesis is organized as follows:

Chapter 2 gives an overview of the literature on the topic.

Chapter 3 proposes an automated system for delivery of VoD contents which redistributes the contents in a hierarchy of servers with the purpose to concentrate the traffic closer to the clients and, thus, to minimize the traffic cost and balance the load. For that purpose, it proposes a redirection strategy and a redistribution algorithm which brings replication or deletion decisions based on the state of the system and the users' behavior.

Chapter 4 extends the model proposed in the previous chapter by including the peers in the process of streaming. Then, by means of simulations gives analysis on how the capacity of streaming of the peers and the distribution of the video contents on the peers affect the quality of service, the localization of the traffic and the reduction of the traffic for streaming the contents, assuming that the contents are optimally distributed on the servers. This chapter proposes popularity based distribution schemes of the contents on the peers and analyzes the influence of these schemes on the system's performance.

Chapter 5 defines a precise mathematical model of the system proposed in Chapter 4 based

on the queuing theory. After the verification of the accuracy of the results obtained with the mathematical model, it is used as a tool for a thorough analysis on the influence of the system's parameters and the content distribution on the overall performance.

Chapter 6 extends the mathematical model from Chapter 5 by introducing the possibility that the peers fail and re-join the community after a certain time. After the validation, it focuses on analysis on how the failures and the other system parameters impact the system's performance.

Chapter 7 gives a summary of the conclusions of the work and the possible directions for future work.

Chapter 2

Related Work

The distribution of the video contents in the networks is one of the most appealing challenges for a large community of researches because the videos are the type of contents that occupy a major part of the overall traffic in the Internet. Their popularity has been constantly increasing together with the requirement of higher quality videos, which threaten to congest the distribution networks. The prediction of the growth of the video traffic in the future is such that by the year 2017 it will occupy 80-90% of the global consumer traffic worldwide (Cisco Systems, 2013). More precisely, the equivalent of the generated traffic for VoD delivery will be 6 billion DVDs per month. Therefore, the main challenge for the research community is to reduce the traffic that is generated for delivery of the videos in the networks and, at the same time, to provide a satisfactory Quality of Service (QoS) without big changes of the existing infrastructure and costs.

One of the initial approaches that has been considered for solving the problem of the network congestion is placing proxy servers in various points of the network. The key feature of the cache servers is that they store the previously requested content and use them for future requests for the same contents. When a client requests a content which is already cached, it is delivered by the cache server, instead from the central server, in order to save on bandwidth in the network core. This approach was originally implemented in the distribution of web contents in various network architectures (Rodriguez *et al.*, 2001) and was successfully adjusted for video delivery. It partially decentralizes the content distribution because instead of sending a request to a central server, the clients send their requests to the nearest cache server. If the cache server does not contain the required content, it requests it from the central server and then distributes it to the client. However, because of the large size of the video contents, the number of contents that can be stored in the proxy servers is limited, and therefore, various segmentation techniques are implemented to enable the proxy servers to store fractions of the videos so that more videos can be cached (Jin *et al.*, 2002;

Chae *et al.*, 2002; Wu *et al.*, 2004; Liu, 2004; Chen *et al.*, 2003). These techniques divide the contents into segments and cache them according to their importance. Thus, whenever a client requests a video, its cached parts are immediately delivered and the proxy server simultaneously requests the rest of the video from the central servers.

The drawback of the caching techniques is that they do not contribute to the reduction of the traffic when the videos are requested for a first time. An architecture that is commonly accepted, where the contents are pre-distributed so that they are closer to the users are the Content Delivery Networks (CDNs) (Buyya *et al.*, 2008; Bartolini *et al.*, 2003). A CDN is a specific architecture which is aimed to solve the scalability problem of the Internet content distribution. It was originally introduced for satisfying the increased demand of web requests for famous web sites and dealing with the "flash-crowd" situations (Jung *et al.*, 2002), but very soon they were largely accepted for delivery of video contents. The CDNs are based on a large-scale, geographically distributed replica servers located closer to the edges of the Internet for efficient delivery of the contents. Most of the CDNs are owned by companies that offer their services to companies that need reliable and fast content delivery. The leading commercial CDNs are Akamai (Nygren *et al.*, 2010) and Limelight Networks (Limelight, 2013). There are also academic CDNs which provide content delivery for free but can offer limited storage capacity. Some of the well-known academic CDNs are Globule (Pierre & van Steen, 2006), CoralCDN (CORAL CDN, 2013) and CoDeeN (CoDeeN, 2013). Both the commercial and the academic CDNs have the same main goal: to minimize the network impact on the content delivery, as well as, to overcome the server overload problem that is a serious threat for busy sites serving popular contents.

As the videos started to dominate in the Internet traffic, the CDNs became a convenient solution for hosting such contents and many of the approaches for distribution of web contents emerged as an acceptable solution. The research in the CDNs was mainly concentrated in solving the replication problem which consists of determining the number of replicas that have to be made for a given video and the servers where to be placed in order to optimize certain cost and QoS. The replica placement problem has been extensively investigated in the past years. The most general form of definition of replica problem is to choose K replicas servers among N potential servers such that some objective function is optimized, also known as a k-median problem (Garey & Johnson, 1979). The most common values that are optimized with the k-median problem are the latency, the total bandwidth consumption, or any specific cost assigned to the links. In (Qiu *et al.*, 2001), the replication is set as a k-median problem of storing replicas in a manner that a certain cost is optimized. It analyzes a static case of user requests pattern and offers a variety of algorithms for replica placement. This solution, however, does not limit the number of request that can be served by one server and does not consider the traffic that is generated while the replicas are being distributed.

Since the k-median problem is NP hard, there are several approximation algorithms that try to find the optimal solution. The random algorithm (Qiu *et al.*, 2001; Kangasharju *et al.*, 2002) is the simplest approximation of the given problem. It randomly chooses K replicas among N potential servers with a uniform distribution. The algorithm is executed several times and the solution that yields the lowest cost is chosen. The hot spot algorithm (Qiu *et al.*, 2001; Karlsson & Mahalingam, 2002) attempts to place replicas near the clients that are generating the greatest load. It sorts the N potential servers according to the amount of traffic generated within their vicinity. It places the replicas at the top M servers that generate the largest amount of traffic. Fan-Out (Radoslavov *et al.*, 2002) places contents at the servers with the highest number of outgoing connections, irrespective of the actual cost function. The idea behind this method is that the servers with large out-going connections are closest to all other servers and are therefore, a better choice for replica placement. More complex algorithm is the greedy algorithm (Qiu *et al.*, 2001). The basic idea of the greedy algorithm is as follows: in order to choose K replicas among N potential servers, only one replica at a time is selected. In the first iteration, each of the N potential servers is individually evaluated to determine the server with the lowest cost. The algorithm goes on until K replicas are chosen. An improvement of the last algorithm that takes in consideration the limited storage capacity is the robust greedy algorithm (Sugavanam, 2007). First, the contents are sorted in a decreasing order of popularity. Then, the most popular content is replicated at each of the CDN servers where there is enough storage space. The algorithm then identifies a replica that if dropped from the server will result in a maximal reduction of the cost function. The algorithm drops this replica and substitutes it by the replica of the currently considered content. The process continues until only K replicas of the content remain within the CDN. After placing K replicas for each content, the algorithm further replicates those contents that can potentially have a significant effect on the cost due to server failures. The performance of the greedy algorithm is further improved in (Yang & Fei, 2003) where instead of placing a predetermined number of replicas, an optimal number of replicas K is calculated, and then the greedy algorithm is executed K times. Unlike these algorithms that focus on improving some average performance measure of the entire community, (Wang *et al.*, 2006) focuses on placing the replicas such that they meet some QoS requirements with the objective of minimizing the replication cost.

A major limit of all these solutions is that they do not consider the natural dynamics of the users' requests. When the conditions in the network change in a way that different placement is cheaper or more satisfying for the users, the only possible solution is to re-apply the placement algorithm from scratch. The problem with such approach is that it may react slowly to the system changes so that the new placement of the replicas is not the best one for the current users' request pattern. Moreover, the replica placement is done

without considering where the replicas were previously placed. Solutions for dynamic replica placement are proposed in (Presti *et al.*, 2007). These solutions are dynamic in the sense that replicas are added and removed from the CDN servers according to the dynamically changing users' request pattern. The redirection of the requests is also adjusted to achieve load balancing among the different replicas. Each site autonomously decides on whether some of the replicas it stores should be replicated or removed. This decision is based on local information such as the number of contents stored in the server, the load of the servers and the number of requests they are currently serving. The authors of (Presti *et al.*, 2007) address the dynamic replica placement and the users' requests redirection in the CDNs and propose a fully distributed solution that minimizes the costs for replica placement and maintenance, but at the same time, satisfies all the requests for videos and keeps the replicas above a target level of utilization. Another replica placement strategy is proposed in (Huang *et al.*, 2005), where some of the crucial factors for placing the contents are the latency and the load balancing. The main goal of this strategy is to place replicas so that latency bound requirements are met, the workload of all the servers is balanced and the replication cost is low. A more specific formulation of the problem where the QoS is considered is given in (Fu *et al.*, 2008).

Apart from the CDNs, there are many other Internet based architectures. Nevertheless, the high traffic demands of the VoD service cannot be always satisfied by these architectures because of the uncontrollable and unpredictable character of the Internet. Therefore, many VoD solutions move towards development of new architectures in the managed networks. These networks are a convenient solution because their size and capacity can be adjusted according to the number of the subscribed users, and the traffic can be controlled and spread over the network using technologies that are not possible in the Internet. The IPTV (Simpson & Greenfield, 2007) emerged as one of the most suitable platform for delivery of VoD contents. Apart from the main service of multicast linear TV, the IPTV offers more personalized services like TSTV and NPVR (Stockhammer & Heiles, 2009), which require dedicated data flow for each request. Because of their growing popularity, the problem of the optimal network utilization and the provision of an improved QoS is a main concern for many research works. Following this direction, the authors in (Verhoeyen *et al.*, 2008) propose an algorithm for optimal placement of video contents for various IPTV services based on the popularity of the contents without considering the state of the network. Another IPTV-based architecture and algorithm for optimal placement of video contents is presented in (Borst *et al.*, 2009). Few different algorithms for replication and placement of VoD contents within a cluster of media servers based on the users' request pattern are proposed in (Zhou & Xu, 2007; Dukes & Jones, 2002).

No matter how good these solutions might be, they all reach a point from where no fur-

ther improvements can be done due to the limitations of the available streaming and storage resources. Therefore, the P2P concept has emerged as a widely accepted solution for overcoming the problem of high intensity traffic for distribution of the video contents (Li, 2008). Guided by the positive experiences from the file sharing on the Internet, this concept has encouraged many researchers to consider its implementation in the distribution of video contents for overcoming the resource problems. The P2P approach was first used for delivery of live video streams over the Internet in real-time, where the peers that watch certain channel participate in the delivery of the same content to the other peers that later join the system (Liu *et al.*, 2008). The advantage of the live streaming is that the peers are watching the same contents at same time, unlike the VoD systems where the clients request videos asynchronously, which makes them more complex to develop. Depending on the overlay formed by the peers that join in the system, the existing P2P solutions are categorized as tree-based or mash-based. The tree-based architectures are chronologically the first P2P live streaming solutions where the joining peers form a tree overlay structure with a channel server as a root, from which the stream is multicasted. Since not all the network environments allow implementation of network-layer multicast, these architectures use application-layer multicast to deliver the stream from the root to the leaves. Depending on the number of trees, the solutions in the literature are divided into single and multiple-tree solutions. The representatives of the single-tree solutions are ESM (Rao & Seshan, 2002), NICE (Banerjee *et al.*, 2002), Zigzag (Tran *et al.*, 2003), CoopNet (Padmanabhan *et al.*, 2002) and SrpreadIt (Deshpande *et al.*, 2001). The main issue of these single-tree streaming systems is that the leaves of the tree remain unused, and thus, the peers as a whole are not efficiently used for reducing the video traffic. Another issue is that they generate large control overhead traffic. This problem is partially solved in the multiple-trees systems SplitStream (Castro *et al.*, 2003), Bullet (Kostić *et al.*, 2003), AnySee (Liao *et al.*, 2006) and DirectStream (Guo *et al.*, 2003a) which divide the videos into sub-streams and form multiple-tree streaming overlays. Although they improve the utilization of the peers, they are not resistant to the peer churn (process when the peers join or leave the community) since when some of the peers leave the system, the tree overlay might be broken and a new overlay tree has to be established. The tree overlays are also used in P2P systems for distribution of VoD. The basic idea behind these systems is multicasting the content by forming a tree from the peers that watch the same video within a given time frame. Some of the well-known tree-based systems in the literature are P2Cast (Guo *et al.*, 2003b), P2VoD (Do *et al.*, 2004) and P-chaining (Kim & Yeom, 2007).

Although there is numerous literature on the tree-based systems, they did not achieve the commercial success. The systems that did achieve their commercial highlight are the mash-based P2P solutions for both live and VoD streaming. In these systems, the peers form mesh-like overlay structure (Hei *et al.*, 2008) and avoid the churn problem and the low utilization

of some of the peers, typical for the tree-based systems. In the mesh-based systems, the peers establish connections dynamically and maintain relationship information with their neighbor peers. When some of these neighbors leave the system, new relationships are established with other peers so that a mesh network with certain number of connection per node is maintained. The mesh overlay is usually formed by using protocols similar to Bittorent (BitTorrent, 2013), the widely accepted protocol for P2P file sharing. The system collects information related to every peer that joins the community via a node called tracker. Whenever a new peer needs some contents, it contacts the tracker, which sends it a list of peers that have the desired content and establishes a connection with selected peers from the list.

The main reason for the commercial success of the mash-based systems is that they provide low cost video delivery and support a large number of simultaneous users. The first commercially successful representative of the large-scale mesh-based systems for live video streaming is CoolStreaming (PPlive, 2013a; Xie *et al.*, 2007) winch was followed by the equally successful PPLive (PPlive, 2013b), PPStream (PPstream, 2013; Jia *et al.*, 2007), SopCast (SopCast, 2013), UUSee (UUSee, 2013) and TVAnts (TVAnts, 2013). However, the extensive analysis and performance evaluations in (Li & Yin, 2007; Hei *et al.*, 2007; Xie *et al.*, 2007; Li *et al.*, 2008; Spoto *et al.*, 2009; Fallica *et al.*, 2008) show that these systems still suffer from issues like long start-up delay, moderate streaming rates, difficulty to reach all the peers in the Internet community, load imbalance on the peers and other issues related to the dynamics of the systems.

The mesh-based approach for P2P delivery of VoD contents is used in the system BiToS (Vlavianos *et al.*, 2006). Since the BitTorrent fetches pieces of the file out of order, which makes it unsuitable for real-time applications, the BiToS system introduces modifications of the BitTorrent in order to give a priority to those portions of the video which are close in time to the currently watched part of the video, thus, providing uninterrupted service. In (Kawarasaki & Suzuki, 2009), the authors extend the BitTorrent to meet the requirements of the real-time VoD delivery and to overcome the server dependence and fairness issues of the protocol. Another system for P2P VoD is VMesh (Yiu & Chan, 2007). Unlike the BitTorrent protocol, this system uses Distributed Hash Tables (DHT) to locate the contents, i.e., for every pair of peer ID and content ID, a hash key is generated which is distributed among the peers. To locate the contents among the peers, the system uses a special DHT protocol.

There is also a significant work in the recent years related to hybrid solutions for distribution of VoD contents which combine the classical client-server approach and the P2P. In these hybrid systems, the requests for videos are initially checked to be served by the peers, and if that is not possible, they are served by a dedicated server. As the videos are divided into strips, in most of these systems, the strips of the same video are delivered by both the peers and the servers. Therefore, this kind of streaming is also referred to as peer-assisted

streaming. The aim of the hybrid systems is to increase the reliability of the P2P service by providing a guaranteed delivery from the servers, but also to reduce the load on the streaming servers by taking advantage of the assistance of the peers in the streaming.

The systems for peer-assisted VoD streaming Toast (Choe *et al.*, 2007), BASS (Dana *et al.*, 2005) and JOOST (JOOST, 2013) achieve improved QoS, increased scalability and a significant reduction of the server traffic by introducing modifications of the BitTorrent. These systems also use a dedicated server to compensate for the parts of the videos that cannot be provided by the peers in real-time, because they are either unavailable or arrive out of order. In (Carlsson *et al.*, 2009), the authors further increase the reduction of the server traffic in BitTorrent based system by proposing incentive policies that motivate the peers to participate in the streaming. In (Huang *et al.*, 2007), a peer-assisted VoD system is proposed, where only the peers that are currently watching a video can assist in the streaming process. The system implements a different set of pre-fetching policies for optimal use of the peers' uplink capacity. These policies are used depending on the demand, the resources that the peers can supply and the state of the streaming servers. In (Kozat *et al.*, 2009), the reduction of the server load is achieved by the introduction of a control server that is fully aware of any information related to the peers that are part of the system, which implements various caching policies for placement of the already watched videos on the peers. In PROP (Guo *et al.*, 2004), the VoD streaming is assisted by the peers that self-organize in a mash overlay that supports DHT for locating the requested video chunks. Instead of a dedicated index server, the tracking of the peers is performed by the very peers which have additional role as index servers.

In all the previously considered Internet based solutions, the reliability issue of the P2P remains the main concern because the QoS on the Internet cannot be guaranteed, and the participation of the peers depends largely on the will of the users to cooperate. These issues are overcome in the IPTV managed networks where the QoS can be provided, and the human factor of participation in sharing of the contents can be significantly reduced. The operators have the control over the users' STBs and can easily reserve part of their storage and uplink capacity for streaming purposes. Another convenience of the STBs is that nowadays they are becoming cheap storage devices with capacity to store large amounts of data. These facts made the IPTV networks a suitable environment for implementation of the P2P in the distribution of video contents and a field of extensive research. The conditions under which the P2P concept becomes beneficial for providing IPTV services are studied in (Chen *et al.*, 2007, 2009; Cha *et al.*, 2008). A P2P model for streaming of live IPTV channels in managed networks is presented in (Wu *et al.*, 2010).

Since the live streaming in the IPTV platform is already optimized by using multicast, the VoD streaming caused a greater interest in the research community. A system for peer-

assisted streaming of VoD contents in managed environments is presented in (Kerpez *et al.*, 2010) where the P2P sharing is localized within the same access network. The analysis of the system shows that the peer-assisted steaming can substitute the streaming traffic of the VoD servers in the Passive Optical Networks (PONs). The authors of (Jayasundara *et al.*, 2011) go one step further by proposing an algorithm for optimal placement of the contents with bandwidth and storage constraints. However, this scenario is not universally applicable because not all the users have an optical link to their homes.

A typical example of system for peer-assisted VoD contents is (Janardhan & Schulzrinne, 2007). It uses a BitTorrent-like protocol and DHT for locating the contents. The system also pre-fetches the popular contents to reduce the server traffic. In the same context, solutions for peer-assisted streaming in managed environments where the contents are pre-fetched on the peers in the low-activity hours are proposed in (Chen *et al.*, 2009; Suh *et al.*, 2007). In (Chen *et al.*, 2010), there is a strategy for placement of popular videos, while the authors in (Brosh *et al.*, 2009) propose distribution strategies that prove to be highly efficient when the not so popular videos are stored in the peers. The authors of (Dyaberi *et al.*, 2010), apart from the BitTorrent-based peer-assisted VoD system, propose a distribution algorithm, which decides on the optimal number of replicas of the videos that have to be stored in the peers according to the popularity of the contents and the storage and streaming constrains of the peers. Another work that uses the contents popularity to determine the optimal number of replicas on the peers is the model for peer-assisted streaming in (Muñoz Gea *et al.*, 2012) where the authors also propose a dispatching strategy for optimal utilization of the available uplink capacity of the peers. The authors in (Loeser *et al.*, 2005), also focus on distribution schemes of the contents for fair load balancing, but additionally, propose a method for predicting the access probability of the videos.

The authors of (Lee *et al.*, 2009) achieve to improve the performance of the IPTV service by proposing a performance-based content sharing network and a replication scheme based on the users' interest to view certain type of contents to replicate the contents, increase their availability and optimize the sharing of data among the IPTV users. The peer-assisted system for VoD in (Korosi *et al.*, 2009) focuses on the received QoS. It not only achieves the guaranteed QoS in case of unreliable upload capacity of the peers, but also significantly alleviates the streaming servers. Another system for peer-assisted VoD streaming which achieves a guaranteed QoS is proposed in (Zhu *et al.*, 2010).

Unlike these systems that use only the P2P and the unicast methods for distribution of the VoD contents, the adaptive hybrid architectures HySAC (Kulatunga *et al.*, 2011) and CPM (Gopalakrishnan *et al.*, 2009) achieve reduction of the overall traffic and a better QoS by dynamically changing the use of the unicast, multicast, P2P and pre-fetching delivering methods, depending on the popularity of the contents and the state of the network.

The systems for P2P delivery of both live and VoD contents presented so far prove to give satisfactory results by means of simulations or log data obtained from real implementations. There is also a significant amount of work dedicated to mathematical models of the P2P systems. Such an example is (Ge *et al.*, 2003), where a mathematical model of a framework for file sharing applicable to various P2P architectures is proposed. A general model for different P2P systems using the queuing theory is presented in (Li *et al.*, 2009; Ramachandran & Sikdar, 2005). In (Fan *et al.*, 2006; Qiu & Srikant, 2004), the stochastic fluid theory is used to mathematically model BitTorrent-like P2P systems. Another model for content delivery in multi-P2P networks, based on structured and unstructured P2P networks is proposed in (Lu *et al.*, 2005), where the authors define a relation between the probability of finding a content on the peers and the size of the network, the size of the interconnected P2P clusters and the users' request rate. The stochastic fluid theory is also used by the authors of (Kumar *et al.*, 2007) to model P2P streaming systems including the churn of the peers, their heterogeneous steaming capacities, the buffering and the delay. The authors in (Wu *et al.*, 2010, 2009), study the performance of multichannel live streaming systems using closed and open queuing network models. An analytical model of mash-based unstructured P2P system for VoD which enables observing how different system parameters affect the system's performance is proposed in (Lu *et al.*, 2008).

Mathematical model for a more reliable hybrid P2P systems where both peers and streaming servers are participating in the streaming is presented in (Rimac *et al.*, 2008), with the accent on modeling the server capacity for providing a satisfactory QoS to the clients. The authors in (Tu *et al.*, 2005), also propose a hybrid P2P model with a focus on the capacity growth in the system and the effects of various factors on the growth. These models, however, are not very applicable in the controlled closed environments, where the number of subscribers is not highly dynamic and the operator can take advantage of their streaming and storage resources, despite the fact that in a given moment they may not all participate in the streaming. Therefore, these models cannot answer the questions of type how some crucial parameters like storage and streaming capacity can alleviate the streaming servers. The closest work that addresses these questions is the statistical model for P2P VoD streaming in (Zhou *et al.*, 2011). This model is applicable in networks where the clients have enough uplink capacity to stream entire videos and focuses on how the proposed distribution strategies contribute to the performances. Although the proposed models accurately describe the P2P systems, they do not take into consideration how different distribution schemes affect the system performance. Besides they assume that the peers have uplink capacity which is higher than the playback rate of the videos and do not provide deeper analysis on how the failure of the peers will affect the server traffic.

Although there is enormous work related to distribution of VoD contents in the networks,

there are still some open issues. There are already frameworks and algorithms for optimal distribution of VoD contents in privately managed networks which introduce bandwidth savings in the network, however, they do not treat the additional traffic for redistribution the copies of the videos among the servers, the QoS received by the clients and the responsiveness of the system in the time as a result of the implementation of the algorithms. These issues are addressed in Chapter 3, where a framework for VoD delivery is proposed which implements a redistribution algorithm for optimal placement of the contents according to the users' behavior and the state of the system. Since any algorithm for distribution of VoD contents cannot reduce the bandwidth demand more than certain optimal value, in Chapter 4 the peers are given the capability to stream part of the videos so that the streaming servers are alleviated. What makes this chapter different from the work presented in the literature is that it treats the issue of the dependence of the distribution of the contents on the QoS and the cost reduction. In order to study the behavior of the system for peer-assisted VoD streaming more thoroughly, without the necessity of the time-consuming simulations for every change of the input parameters, in Chapter 5, a stochastic model of the system is proposed. The model is unique in the research community because it takes into consideration a large variety of network parameters, including the distribution scheme of the contents, to estimate the system's performance. The work presented in Chapter 6 goes one step further in the realistic modeling of the system for peer-assisted VoD streaming by including the failures of the peers as a part of the streaming process. Thus, it proposes an extended mathematical model which is used for studying the traffic demands in the streaming servers as a result of the failures and the recovery of the peers.

Because of the specific nature of the dependence of the states comprising the mathematical models presented in the Chapter 5 and Chapter 7, the mathematical methods from the literature used for theoretical calculation of some important probability functions in the models are inconvenient for use in practice because of the large size of the real networks that they model. Therefore, Appendix A proposes numerical extrapolation method that solves the probabilistic functions independently on the size of the network. Despite this acceleration, the mathematical models include a huge set of states that cannot be solved with the current computation technology. In order to solve this issue, in Appendix B, a sampling method is proposed that reduces the size of the models and thus, enables computation of the results in decent time.

Chapter 3

Content Delivery Network for VoD Streaming

The VoD service is becoming a dominant service in the telecommunication market due to the personalized experience it offers to the clients regarding the choice of content items and their independent viewing time. However, this convenience also implies high server storage and streaming demands because of the large variety of content items and the high amount of traffic generated for serving the requests of the growing number of users. The price that the operators have to pay for this service is the increased traffic in the networks, which are becoming more congested due to the higher demand for VoD contents and the increased quality of the videos.

As a solution, this chapter proposes a hierarchical network system for VoD content delivery in managed networks, which implements redistribution algorithm for optimal content distribution and a redirection strategy. The system monitors the state of the network and the behavior of the users to estimate the demand for the content items. When the system detects that some of the servers are overloaded, or the traffic is not equally balanced within the network because of the popularity change of the contents, it executes a redistribution algorithm which, based on the clients behavior and the network state, takes decision on new positions of the replicas of the contents in the network. The system's objectives are to distribute replicas of the content items in the network in a way that the most demanded contents will have replicas closer to the clients so that it will optimize the network utilization and will improve the users' experience. It also balances the load between the servers concentrating the traffic to the edges of the network.

The proposed system is modeled in a simulation environment and the efficiency of the algorithm is tested for various simulation scenarios. The goals of the simulations are to prove

that the system is adjustable to the clients' request pattern and it is highly responsive to the changes of the contents popularity and the intensity of the requests.

3.1. System architecture

One of the most common approaches for serving a large community of clients is the placement of cache servers close to the clients that will store the most demanded contents. This approach saves valuable resources in the core of the network, since once a content is requested from the core servers, it is placed in the cache servers and each future request for the same content is redirected to the cache server. A similar approach is used in this work, however, for the purpose of optimal video delivery, other features like monitoring, load balancing and redistribution are added.

The proposed system for optimal VoD content delivery is designed for private networks owned and managed by a company that offers VoD streaming and other TV and data related services. The provider has the overall control of the network and can dimension it according to the number of subscribed users.

The model consists of three logical entities: clients, streaming servers and management servers which are responsible for the automatic content distribution and service selection (Figure 3.1).

The streaming servers have a double role: they serve the clients and they deliver content items to other servers in the network. These servers have limited storage and streaming capacities and therefore, can host only a part of the entire video library and can serve only a part of the clients. The streaming servers are connected with high-capacity links that are able to accept the high-intensity streaming traffic directed to the clients and the traffic generated by content distribution to the servers. The servers store the content items in compressed format and stream the packets with data rate that is sufficient for the client to get uninterrupted video sequence. They provide true streaming, i.e., they deliver the packets in real-time and have the capacity to simultaneously serve large number of clients. They are placed at different points of the network, organized as an n-tier hierarchical architecture. Starting from the top of the hierarchy, every level downwards is closer to the clients. Although the hierarchy may contain arbitrary number of levels, this work considers three-tier hierarchy composed of Central servers (CS), Branch servers (BS) and Edges servers (ES) (Figure 3.2). From an IPTV architecture point of view, the ES are actually placed in a Central Office (CO) and form local communities with the clients that are connected to the same Digital Subscriber Line Access Multiplexer (DSLAM) or access network. The BSs are placed in the Video Source Offices (VSOs), the CSs are in the Video Hub Offices (VHOs) and the Central Repository

(CR) server, which is the storage server that contains the entire video library, is in the Super Head End (SHE). The CSs represent the highest level of the hierarchy. They possess high storage capacity because they have to store most of the offered contents. Since the tendency of the system is to store the popular contents in the lower layers of the architecture, the CSs do not necessarily have to serve large number of requests. The BSs are an intermediate level containing replicas with moderate popularity that are not popular enough to be placed in the ESs. The last level, which is closest to the clients, is represented by the ESs. They are placed at the periphery of the network and serve a community of geographically close clients. The ESs host the most popular contents, i.e., they serve most of the clients' requests and therefore, have high streaming capacity.

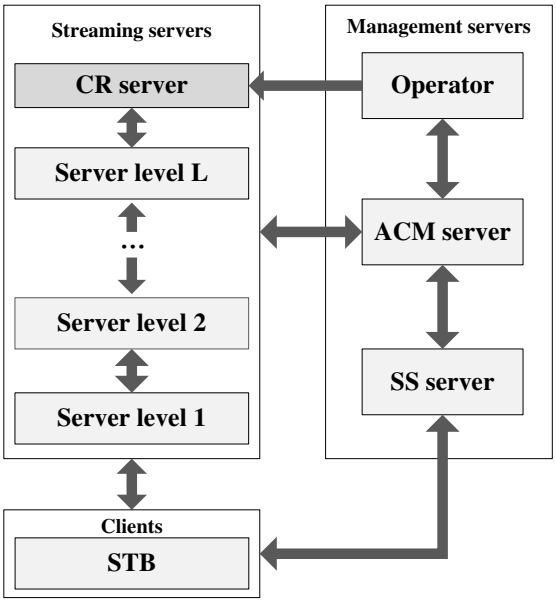


Figure 3.1.: Block model of the system for VoD service.

At the top of the hierarchy resides the Central Repository (CR) server. It contains all the contents that are offered by the operator, and, unlike the rest of the streaming servers, it does not serve clients. This server is as an entry point for new videos introduced in the system and an origin point for distribution of replicas to the streaming servers down the hierarchy. It is accessed by the streaming servers for distribution of contents that are not available on any of the servers.

The clients are placed at the bottom of the architecture and their role in the system is to make requests for contents and to process the received streams from the streaming servers. They are connected to the managed network and subscribed to the VoD service. In order to be able to use the VoD service, the clients use a dedicated device called Set-Top Box (STB), which is owned by the operator. These devices have internal buffer that stores the received

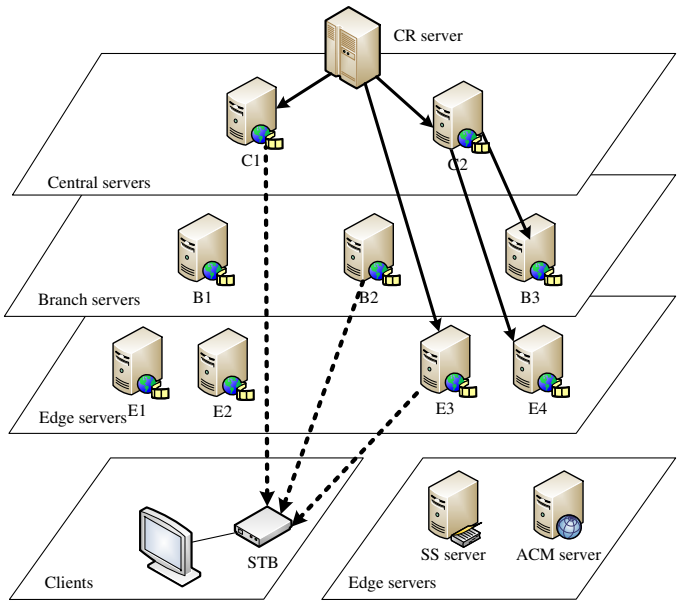


Figure 3.2.: Three-tier system architecture.

packets for decoding and prevents interruptions when there is network congestion.

The management servers are represented by the Operator, the Automatic Content Movement (ACM) server and the Service Selection (SS) server.

The Operator represents an entity of the system with functionality to introduce new contents, remove unpopular contents and to configure parameters relevant for the redistribution algorithm.

The ACM server has a central role in the entire system. It communicates with all the servers, monitors the system, takes redistribution decisions and issues commands to the servers. The ACM server monitors the state of the network by periodically issuing commands to the streaming servers. Upon reception of the state information from all the streaming servers, it forwards it to the SS server for redirection purposes. Whenever it detects that there are overloaded servers or that the users' behavior has changed, it runs an algorithm for content redistribution. Using popularity data of the contents in the recent past, previously obtained from the SS server, the algorithm decides whether a replica of a content item should be moved to another server, removed or left as it is. The execution of the algorithm results with a new distribution of the content items in the system, which is deployed by execution of a set of removal and replication commands issued by the ACM server. Along with issuing the commands, the ACM server sends the new availability of the contents to the SS server.

The SS server is responsible to accept the clients' requests and to redirect them to the most appropriate streaming server. It implements a redirection strategy that uses the current state

of the system and the availability of the contents to take a decision on where to redirect the clients. The SS has the capacity of accepting huge number of request, because it is the entry point of every request in the system.

The logical organization and the communication paths between the basic functional entities are shown in Figure 3.1. The clients exchange messages with the SS server and any of the streaming servers. The SS server, apart from the clients, is communicated by the ACM server which sends the location of every content in the system and the state of each server, which the SS server uses for redirection purposes. The ACM server communicates with all the entities in the system, except with the clients. It constantly interchanges messages containing commands about actions that have to be executed or data about the state of various entities. The operator has one way communication with the CR server which consists of issuing deletion commands or insertion of new contents. Its communication with the ACM server is bidirectional and consists of configuration messages.

3.2. System operations

The functionality of the system is mainly defined by a flow of messages between the clients, the streaming servers or the management servers. These messages can contain both management and video data. Depending on the content of these messages and the state of the entities included in the communication, there are different operations defined for the proposed system.

3.2.1. Content request

The basic function of each client is making request for a VoD content. Every client has access to a list of available contents which is updated whenever a new content is introduced or removed from the system. The client first establishes a connection with the SS server and then sends a request for a content by indicating the unique ID of the desired content. After checking on the content's availability and the system's resources to serve it, the SS server responds with a message with information that may lead to one of the following situations:

- The content is available somewhere in the system and there is a streaming server that can serve it. In this case, the response of the SS server is affirmative and contains the address of the most appropriate server to which the client should direct the request for service. Upon reception of this message, the client closes the connection with the SS server and establishes a new connection with the indicated server. It then sends the same kind of request it previously sent to the SS server and initiates a streaming session with the streaming server.

- The content is not available at the moment, but it is assigned to be distributed when requested. In order to prevent flooding the network with distribution of the contents among the servers in the moments after the execution of the algorithm, when the new distribution of the contents is determined, some of the contents that should be moved to other servers are only assigned to be distributed by reserving the required storage space. When the client requests a content that is only assigned for distribution, (1) in Figure 3.3, the SS server affirmatively responds to the client with the address of the assigned server (2). Then, it issues a command to the ACM to handle the situation of distribution of the content to the assigned server (3). The ACM chooses the CR server as an origin server and sends it a command directing it to push the content to the destination server (4). The CR server then multicasts the content to the destination server and all its children servers that are in the same level as the destination server (5). The ACM also sends a message to the SS server (6) to update the entries in the availability matrix for the new destination servers (they are marked as available, instead of assigned for distribution). In most of the cases, the origin server is the CR server because it contains the entire video library. Afterward, the client requests the content from the designated server (7) and initiates a streaming session (8).
- The content is not available on the streaming servers. Because of its low popularity, the content has been removed from the servers, and therefore, the content is distributed to the server which is highest in the hierarchy, i.e., the CS that is closest to the client that requests the content. The procedure of the distribution of the unavailable content is the same as in the previous case.
- The content is available somewhere in the system, but all streaming servers that contain its replica are overloaded due to the large number of clients being served at the moment. Since the system lacks the necessary streaming resources for serving the client, it is directed to close the connection, wait a certain time and retry to make a new request for the same content. In order to avoid flooding the SS server with requests for contents that cannot be served because of overloaded servers, the system employs a mechanism which determines the exact time a client should wait until the next trial. Whenever a client receives a response that the server is overloaded, it generates random number in the range from 0 to a value assigned to a configurable parameter called inter-request interval. This number is the time in that the client should wait to make a new request for the same content. The client keeps on trying until it receives affirmative message, and depending on the availability of the requested content, follows one of the previous procedures.

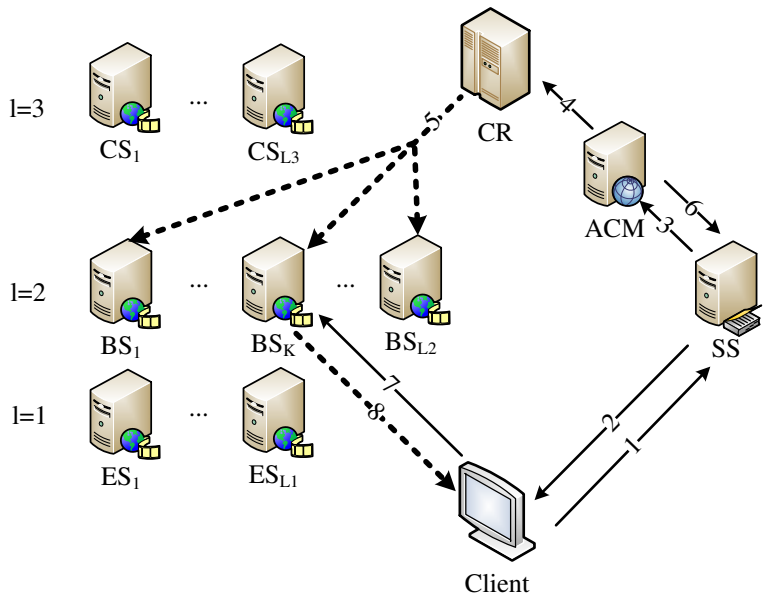


Figure 3.3.: Request handling process of a content that is assigned for distribution.

3.2.2. Service selection

The Service Selection is a service that determines the best server that has to be assigned to any client's request. Apart from its task to satisfy as many requests as possible, the service has to optimize the utilization of the network according to the current placement of replicas and the current state of the network. The optimal network utilization in this contest refers to the maximal utilization of the ESs so that most of the streaming traffic is concentrated closer to the clients. It also refers to keeping the utilization of all the servers well balanced without causing traffic congestion in any part of the network.

The SS gets updated with new information by the ACM sever. Whenever a new content delivery scheme is determined, the ACM server sends the new availability of the contents on every server. This information is used by the SS server to create a list of candidate servers that will serve the clients' requests. Apart from this information, the SS server periodically receives information about the state of the servers. Always when the ACM server receives complete state information from the streaming servers in the system, it immediately forwards it to SS server.

3.2.3. Automatic content movement

The Service Selection service described in the previous subsection has an important role in the system because it balances the load in the network according to the current replica distribution and the servers' utilization. However, it is not able to follow the dynamics of

clients. It tries to optimize the available resources and serve all the clients, but it cannot make any improvement in the service when the current distribution of replicas saturates parts of the network. Therefore, the proposed content delivery system employs an automatic content movement service which monitors the state of the network, and whenever it detects that there is a possibility of overloading of the streaming servers or there is a traffic imbalance, it runs an algorithm to redistribute the replicas so that the servers are capable to handle the future requests. This service is run by the ACM server. The ACM server is constantly monitoring the network state by periodically issuing commands to the servers to request information about their state. The period of issuing these commands is configured by the operator and is called inter-state interval. Upon reception of the state request commands, the streaming servers send the current streaming utilization and the number of currently served streams of each content. Every time the ACM receives server state information it updates an inner state table.

When the ACM receives response from all the servers in the system, it re-sends the same state information to the SS server, which uses it for taking redirection decisions. Having the current state of the system, the ACM server checks whether any of the servers reaches utilization above a defined threshold or whether there is imbalance in the utilization of the servers. In these cases, it starts up a procedure for running the reorganization algorithm. This procedure consists in sending a request for popularity information of the contents from the SS server because this server receives the requests from the clients. The SS server immediately responds with the number of times each replica has been accessed on every server in the interval between the last execution of the algorithm and the current time. The ACM server also sends requests to the streaming servers asking for the current number of streams of each content that they host. When all the necessary data is gathered, the ACM server executes the algorithm to determine whether a content should be moved to another server, deleted or left as it is.

After the execution of the redistribution algorithm, which gives as an output the new availability of the contents, the ACM generates two sets of commands. The first set of commands are deletion commands which indicate to the streaming servers which contents to remove. The second set of commands issued to the streaming servers and the CR server are push commands for instant distribution of the most popular contents. These commands contain the identification number of the contents that have to be pushed and the list of destination streaming servers that will host them. Since it is highly probable that the popular contents assigned to be moved to other locations already have active streaming sessions, the push commands also contain directions for the transfer of all these active sessions to the new servers. This operation achieves almost instant response of the system to the new distribution. Before the ACM server sends the commands to be executed, it updates the SS server with the new

availability so that the future requests can be directed to the new locations of the replicas.

3.3. System parameters

The proposed model for streaming of VoD contents consists of a set of streaming servers S placed in one of the L different levels of a tree hierarchy. The vicinity of the server s related to the clients and the other streaming servers is defined by its level within the hierarchy $l(s)$. A server can have minimum value of the level $l(s) = 1$ if it is an ES or value $l(s) = L$ if it is a server in the last level of the hierarchy (CS). There are G edge servers in the first level, and each server s that belongs to this level serves a group of $N(s)$ clients. The overall number of clients in the system is N.

The state of each server s is defined by the streaming utilization $u(s)$ and memory utilization $m(s)$. The streaming utilization $u(s)$ is defined as the portion of the streaming capacity $U(s)$ that is occupied for serving the requests of the clients and distribution of replicas to other servers. The value of this parameter lets the ACM server determine whether a redistribution algorithm should be run. An important measure tightly coupled to the triggering of the algorithm is the utilization trigger threshold $T(s)$ defined as the maximum value of $u(s)$ which can be tolerated for considering the server as normally loaded. The trigger threshold $T(s)$ is expressed as a percentage of the streaming capacity $U(s)$. Whenever this value is exceeded, the ACM server initiates a procedure for new redistribution in the system. The server storage utilization $m(s)$ is defined as the portion of the storage capacity $M(s)$ occupied for hosting the replicas on server s.

Each server can host one replica from a set of different content items C. Each content item c in the system has size $s(c)$ and streaming rate $r_s(c)$, which defines the duration of the video $d(s) = s(c)/r_s(c)$. The system also maintains information about the replicas of the content items on different streaming servers. This information is kept in an availability matrix of size $S \times C$ where each element $a(s, c)$ has value 1 if a replica of content item c is present on server s or 0, otherwise. The local popularity of the replicas is stored in a popularity matrix of the same size as the availability matrix, where each element $p(s, c)$ represents the number of times a replica of the item c has been requested from server s. The global popularity is the overall popularity of the contents in the system and is calculated as $p(c) = \sum p(s, c)$. The popularity information is gathered by the ACM server before the execution of the algorithm and it refers to the activity of the users during the inter-execution interval ΔT, defined as the time between the previous execution of the algorithm and the current execution. In order to control the frequency of the algorithm execution, a minimum inter-execution interval ΔT_{min} is also defined, which is the minimum time that has to pass between two consecutive

executions, no matter whether the value of the uplink utilization of the servers exceeds the threshold value. The system also keeps the local popularity of the replicas in the previous execution of the algorithm $p'(s,c)$, as well as their global popularity $p'(c) = \sum p'(s,c)$. An overview of the parameters defined for the system is given in Table 3.1.

Table 3.1.: Overview of the system parameters for the CDN for VoD streaming.

Parameter	Description
S	Number of streaming servers
G	Number of ESs
L	Number of levels
$l(s)$	Level of server s
$N(s)$	Number of clients directly served by server s
$u(s)$	Streaming utilization of server s
$U(s)$	Streaming capacity of server s
$m(s)$	Memory utilization of server s
$M(s)$	Memory capacity of server s
$T(s)$	Trigger threshold of server s
C	Size of video content library
$s(c)$	Size of video content c
$r_s(c)$	Streaming rate of video content c
$d(c)$	Duration of video content c
$a(s,c)$	Availability of content c on server s
$p(s,c)$	Number of requests for video content c on server s
$p(c)$	Global number of requests for video content c
$p'(s,c)$	Number of requests for video content c on server s between the previous two executions of the algorithm
$p'(c)$	Global number of requests for video content c between the previous two executions of the algorithm
ΔT	Current inter-execution interval
$\Delta T'$	Previous inter-execution interval
ΔT_{min}	Minimal inter-execution interval
$r(s,c)$	Demand for video content c on server s
$r(c)$	Global demand for video content c
$B(c)$	Total amount of streamed data for content c
$n(c)$	Number of completed streams for content c within ΔT

Continues...

Parameter	Description
$n_c(c)$	Number of initiated and completed streams within the interval ΔT
$n_p(c)$	Number of partially completed streams within the interval ΔT
$n'_p(c)$	Number of partially completed streams within the interval ΔT initiated within $\Delta T'$
λ_c	Arrival rate of requests for content c within ΔT
λ'_c	Arrival rate of requests for content c within $\Delta T'$
t_x	Time of execution of the algorithm
T_i	Inter-arrival time between the requests at t_i and t_{i-1}
$f(i)$	Fraction of streamed content of the i-th request relative to the time $t_x - d(c)$
$q(c)$	Number of interrupted streams within ΔT
$f'(i)$	Fraction of streamed content of the i-th request relative to the time $t_x - \Delta T - d(c)$
$'q(c)$	Number of interrupted streams within $\Delta T'$
$d(s, g)$	Distance from server s to group of clients g
q	Shifting constant of the ZM distribution
α	Skew factor of the ZM distribution

In order to quantify the generated traffic between two executions of the algorithm, a demand parameter $r(c)$ is introduced and is defined as the average load generated on the servers as result of serving requests for the replica of content item c within an inter-execution interval ΔT. The demand is calculated as a ratio between the total amount of streamed data $B(c)$ sent between two consecutive executions of the algorithm and the duration of that interval:

$$r(s, c) = \frac{B(c)}{\Delta T} = \frac{n(c)s(c)}{\Delta T} \tag{3.1}$$

where $n(c)$ is the number of completed streams of content c on the servers during the interval ΔT. This number cannot be easily determined by the global popularity $p(c)$, because it only indicates how many times the content has been requested within ΔT, but it says nothing about how many of these requests have been completed. In order to determine this number, an estimation method based on the popularity data and the duration of the last inter-execution intervals is proposed. This method is using only the redirection data obtained from the SS

server and the streaming utilization obtained from the streaming servers and thus, contributes to the reduction of management data in the network.

In the analysis, it is assumed that the time between two requests for a content is exponential, i.e., the request process is a Poisson process, noted as $N(t)$. Since the requests for a given content item c are independent on the requests for other content items, the main Poisson process $N(t)$ can be presented as a sum of independent Poisson processes $N_c(t)$ with arrival rate λ_c where each one represents the process of requests for content item c. Since the expected number of events for a Poisson process within a time interval ΔT is $\lambda_c \Delta T$, and the actual number of requests within that interval is the global popularity $p(c)$, the arrival rate λ_c can be determined as the ratio:

$$\lambda_c = \frac{p(c)}{\Delta T} \tag{3.2}$$

It is assumed that λ_c has a constant value within the interval ΔT, but its value can change after every execution of the algorithm because there might be a different rate of requests.

An important constraint for the estimation method is the duration of the inter-execution interval ΔT, which has to be longer than the average duration of the videos $d(c)$. Thus, within the time of two consecutive executions of the algorithm, the streams initiated in the first interval will end either in the same interval or in the next interval (Figure 3.4). The condition $d(c) < \Delta T$ also means that some of the initiated $p(c)$ requests will be completed in the same interval, and the rest will finish in the next interval. The objective of the estimation method is to find the overall number of completed streams having as information only the number of requested streams. From Figure 3.4, the number of completed streams $n(c)$ can be expressed as:

$$n(c) = n_c(c) + n_p(c) + n'_p(c) \tag{3.3}$$

where $n_c(c)$ is the number of initiated and completed streams within the interval ΔT, $n_p(c)$ is the number of partially completed streams that were initiated in the interval ΔT, but were interrupted by the execution of the algorithm, and $n'_p(c)$ is the number of partially completed streams that were initiated in the previous interval $\Delta T'$ and were interrupted by the previous execution of the algorithm. The task of the estimation method is to find the value of each one of the members of (3.3).

The initiated and completed streams within ΔT are those that were requested one content item duration $d(c)$ before the algorithm's execution, and therefore, their number $n_c(c)$ is calculated as the expected number of events within the interval $\Delta T - d$:

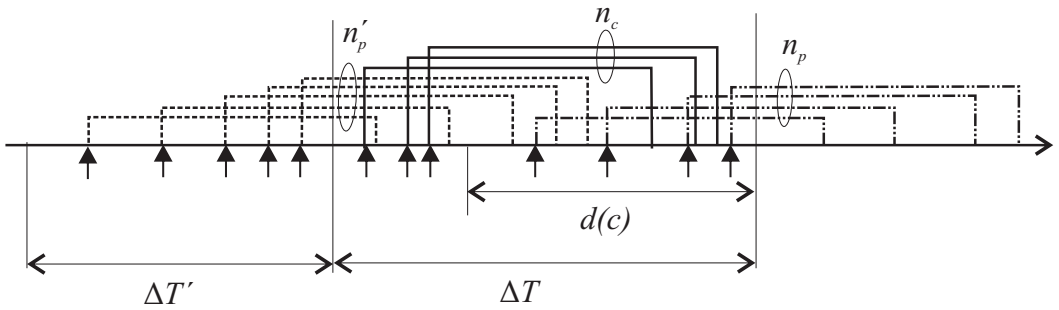


Figure 3.4.: Time-line of different streams for a content item c from streaming server s.

$$n_c(c) = \lambda_c(\Delta T - d) = \left(1 - \frac{d}{\Delta T}\right) p(c) \tag{3.4}$$

where λ_c is the average inter-arrival rate of the requests for content c.

If the time moment when the algorithm is run is marked as t_x, the number of partially completed streams $n_p(c)$ can be calculated by finding the completed fraction $f(i)$ of each stream i initiated within the interval $[t_x{-}d(c), t_x]$ (Figure 3.5), defined as:

$$f(i) = \frac{d(c) - E\left[t_i\right]}{d(c)} \tag{3.5}$$

where $E\left[t_i\right]$ is the expected time of arrival of the i-th request relative to the time moment $t_x{-}d(\mathrm{c})$. This value can be expressed as a sum of exponentially distributed independent inter-arrival times T_i, which leads to:

$$E\left[t_i\right] = E\left[\sum_{k=1}^{i} T_k\right] = \sum_{k=1}^{i} E\left[T_k\right] = \frac{i}{\lambda_c} \tag{3.6}$$

where $E\left[T_k\right]$ is the expected inter-arrival time between the $(k-1)$-th and k-th request and is expressed as $E\left[T_k\right] = \lambda_c^{-1}$.

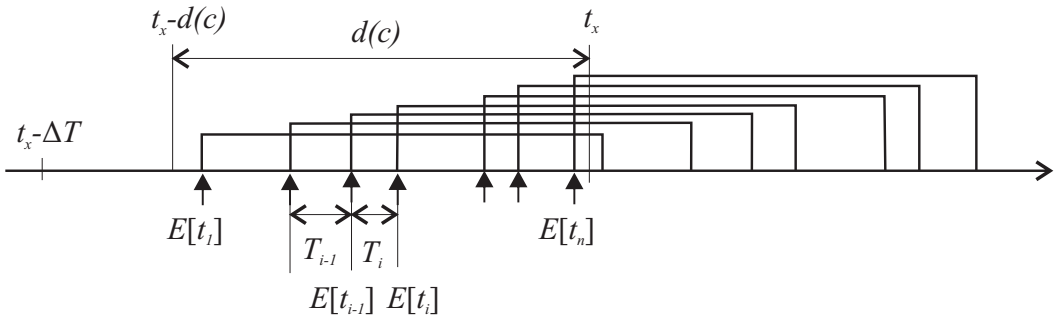


Figure 3.5.: Timeline of the partially completed streams of content c on server s interrupted by the execution of the algorithm.

The total number of partially completed streams $n_p(c)$ will be the sum of all fractions of the interrupted streams at the time t_x. The number of such streams is actually the difference of the total number of requested streams and initiated and completed streams $q(c) = p(c) - n_c(c)$. Thus, the number of partially completed streams will be eventually calculated as:

$$n_p(c) = \sum_{i=1}^{q(c)} f(i) \tag{3.7}$$

If (3.4)-(3.6) are substituted in the last expression, the number of partially completed streams will be eventually calculated as:

$$n_p(c) = \sum_{i=1}^{q(c)} \frac{d(c) - \frac{i}{\lambda_c}}{d(c)} = \frac{1}{2}\left(p(c)\frac{d(c)}{\Delta T} - 1\right) \tag{3.8}$$

The number of completed streams interrupted by the previous execution of the algorithm $n'_p(c)$ can be determined in a similar way, with the difference that the fraction of each completed stream within the interval $[t_x - \Delta T - d(c), t_x - \Delta T]$ is now determined as:

$$f'(i) = \frac{E'[t_i]}{d} = \frac{i}{\lambda'_c d(c)} \tag{3.9}$$

where $E'[t_i]$ is the expected time of arrival of the i–th request in the previous interval $\Delta T'$. In this case, λ'_c is obtained as the ratio $\lambda'_c = p'(c)/\Delta T'$.

Applying (3.9) in the sum of fractions for $q'(s,c) = p'(s,c) - n'_c(s,c)$, it is obtained:

$$n'_p(c) = \sum_{i=1}^{q'(c)} f'(i) = \frac{1}{2}\left(p'(c)\frac{d(c)}{\Delta T'} - 1\right) \tag{3.10}$$

Eventually, if (3.4), (3.9) and (3.10) are substituted in (3.3) and then it is substituted in (3.1), expressing the streaming duration as a ratio between the content size and its rate $d(c) = s(c)/r_s(c)$, the average demand for the content item c on the streaming servers is obtained as:

$$r(c) = \frac{s^2(c)p'(c)}{2\Delta T \Delta T' r_s(c)} + \left(1 - \frac{s(c)}{2\Delta T r_s(c)}\right)\frac{s(c)p(c)}{\Delta T} \tag{3.11}$$

The value of the demand of each content $r(c)$ and the utilization of each server $u(s)$ are used as main parameters in the redistribution algorithm for determining the new positions of the replicas within the server hierarchy.

3.4. Redistribution algorithm

The goal of the redistribution algorithm is to reorganize the system by moving the replicas of the popular contents closer to the clients in order to optimize the network utilization according to the users' behavior. It is run whenever the system detects increase of the streaming utilization of the servers or imbalance of the overall traffic in the network. It consists of three phases. In the first phase, it calculates the optimal position of the contents according to the current demand. This process is initiated in a way that all the contents are "virtually" deleted from the servers, and the servers are then repopulated with contents according to their position within the hierarchy and the popularity of the content. This is also called a phase of desired positions, since the contents are assigned to the best position for optimal usage of the servers, although it might not always be feasible for the operator to redistribute all the contents to the desired positions. In order to prevent big fluctuation of the contents among the servers, in the second phase, the algorithm decides whether it is profitable for the operator to redistribute the contents to their new positions or to leave them in the servers that hosted them prior to the execution of the algorithm. After the algorithm decides on the new final positions of the contents, if there are still streaming and storage resources left on the servers, the algorithm brings back the "virtually" deleted contents on the servers that were not assigned to remain on the servers with the new distribution. In order to increase the responsiveness of the system and alleviate the busy servers, in the third phase, the algorithm transfers the streams of the most popular contents from the overloaded servers to the new servers where these replicas are assigned.

The first phase of the algorithm is the decision on the optimal position of the contents. Since the servers are organized as a tree, where the leaves are the ESs that directly serve one group of users, the aim of this step is to assign a replica of each content to a server that is parent of that group. If the content cannot be placed in the first level, the algorithm goes upwards in the tree and checks the parents of the ES until an appropriate server is found. The steps for this operation are shown in Algorithm 3.1.

First, the contents are sorted in a list D with descending order according to their current demand, and the current storage and streaming utilization of the servers are initialized to value 0. The desired availability of the contents on the servers will be stored in the matrix A_d, which also has initial value 0 for each entry (lines 1-2). Then, starting from the most demanded content (line 3), it is intended to place a replica on every server in the level that is closest to the clients so that all the groups of clients have a direct access to a replica within the hierarchy. Therefore, for each content, the algorithm initializes an empty array where it stores the groups of clients for which there is a replica on a server that is a parent of that group (line 5). Starting from the lowest level, for each server in the considered level

Algorithm 3.1 Calculation of desired placement of the contents

```
1  D=sort(R,'descend')
2  u=0, m=0, Ad=0
3  for each c ∈ D
4      level = 1
5      servedGroups=[]
6      while level<=L
7          levleServers=servers(level)
8          for each s∈levelServers
9              if m(s)+s(c)/M(s)<1 and u(s)<T(s)
10                 u(s)=u(s)+r(c)/U(s)/G*children(s)
11                 m(s)=m(s)+s(c)/M(s)
12                 Ad(s,c)=1
13                 servedGroups.Add(children(s))
14             end
15         end
16         if servedGroups.Size()==G
17             break
18         else
19             level++
20         end
21     end
22 end
```

(lines 8-15), it is checked whether it has available space to store the replica and available streaming capacity left to serve the calculated demand (line 9). If so, the current streaming utilization is increased by the value of the traffic that would be generated if the content is stored in the server, and the storage utilization is increased by the size of the considered content (lines 10-11). The streaming utilization is increased by the amount of the overall demand of the content divided by the number of groups that generate that demand, and the whole expression is multiplied by the number of children groups of that server, which is the number of groups that the server is serving (line 10). The entry for the chosen server and content in the matrix of desired availability A_d is set to value 1 (line 12), and the children of the server, which are the groups that are directly served by the considered server, are added to the array of served groups (line 13). If not all the groups of the system have a provided copy of the content on a server which is their direct parent (line 16), the level is increased (line 19) until this condition is satisfied. The procedure of replica assignment on the servers is repeated for all the contents in the system. It ends when all the servers are depleted or when a replica of all the contents in the library is placed in the servers.

After the calculation of the desired positions of the contents, the algorithm enters in the second phase (Algorithm 3.2) when it decides on whether the desired positions of the contents

are feasible, i.e., whether the traffic cost that they would save for streaming the content from the new position is higher than the cost for distribution of that content to the assigned server. The second phase of the algorithm begins similarly like the first phase. Apart from the initialization of the streaming and storage utilization vectors to value 0, a new availability matrix A_o that will contain the optimal availability of the contents at the end of the algorithm is initialized to values 0. The key difference between this and the previous phase is that, apart from checking the available resources left on the server, the algorithm also checks whether the desired availability of a content is the same with its actual position. If the replica of a content c is desired on server s and is also currently available on the same servers (lines 9-10), then the replica is left on the server, i.e., the values for the utilization and the new availability are simply updated, and the groups served by the considered server is added into the group of clients that have direct access to the replica (lines 11-14). In the case when a replica of the content is desired (line 9), but it is not currently available on the considered serves (line 15), the algorithm calculates the gain of its movement to the currently considered server. In order to calculate the gain, the algorithm first looks for the first servers in a level above the current level that contain the content (line 16). Then, it calculates the traffic cost that would be saved if the content is moved to the desired position. However, the placement of the replica in the new position also requires additional traffic, which reduces the overall gain. Therefore, the gain is calculated by subtracting the distribution cost for movement of the replica from the CR server to the new desired location from the cost savings that would be obtained if the replica is streamed from the new position, compared to its old position (line 17). The replica will be assigned for distribution to the server only if the gain of this operation is positive (lines 20-23). In the opposite case, the replica is not assigned to be stored in the new server and the algorithm continues to check whether the upper levels are more appropriate for hosting a replica of the content (line 18). After checking all the contents or after depletion of all the levels, the algorithm assigns the values of the optimal distribution matrix to the values of the current distribution matrix (line 37).

After the optimal placement, the algorithm checks if there are available unassigned streaming and storage resources on some of the servers, and if that is the case, it reassigns the most demanded contents that were currently available, but were not assigned in the optimal distribution. After this operation, all the remaining contents that were currently available but not assigned in the optimal distribution are assigned for permanent removal from the servers. Afterward, the ACM server sends the obtained availability data to the SS servers, which will use it to redirect the clients according to the new positions of the replicas.

Although some of the replicas are assigned to new locations, they are not all immediately delivered to the new servers. The main reason for this is that the immediate delivery of all the new replicas might generate huge amount of traffic in the core of the network, and

Algorithm 3.2 Calculation of optimal placement of the contents

```
D=sort(R,'descend')
u=0, m=0, Ao=0
for each c∈D
  level = 1
  servedGroups=[]
  while level<L and levelCompleted
    levleServers=servers(level)
    for each s∈ levelServers
      if u(s)<T(s) and Ad(s,c)==1
        if A(s,c)==1
          Ao(s,c)=1
          m(s)=m(s)+s(c)/M(s)
          u(s)=u(s)+r(c)/U(s)/G*children(s)
          servedGroups.Add(children(s))
        else if exists sUp such that l(sUp)>l(s) and A(sUp)==1
            gain=(l(sUp)-l(s))*r(c)-(L-l(sUp))*d/ΔT
            if gain<0
              continue
            end
                    Ao(s,c)=1
                    m(s)=m(s)+s(c)/M(s)
                u(s)=u(s)+r(c)/U(s)/G*children(s)
                servedGroups.Add(children(s))
          end
        end
      else
        continue
      end
    end
    if servedGroups.Size()==G
      break
    else
      level++
    end
  end
end
A=Ao
```

some of the not so popular contents that were moved to the higher levels, might not even be requested by the clients. Therefore the algorithm chooses to instantly deliver only the most demanded contents. The rest of the contents are delivered to the servers upon a request from the clients. The SS server already has the availability data, so if a client requests a content that is not physically present on a server, but it is assigned by the algorithm, it issues a push command to the CR server to instantly provide a copy of the content. Since the system for VoD streaming is built on privately managed network, the delivery of the contents to the servers is done using multicast. Whenever a client requests a content that has to be stored in a server on certain level, the same replica is multicasted to all the servers in the level.

The algorithm determines the optimal position of the replicas, however, the predicted state of the system cannot be immediately reached because the servers are currently serving clients for replicas that are assigned a new position. Therefore, the effects of the algorithm could be visible only after the current streams of the repositioned replicas are over. In order to increase the responsiveness of the system to the new distribution, the algorithm also implements a method for transferring some of the current streams of the repositioned replicas that originate from the busy servers. The aim of this transfer is to immediately alleviate the busy servers and to obtain the predicted state of the system in a shorter time. On one hand, this approach has a downside because at the time of the execution of the algorithm, the clients that are watching the same video are actually watching it at a different position relative to the beginning. Therefore, for uninterrupted experience, there are necessary as many delivery streams as there are clients being served. On the other hand, it has the advantage that there have to be streamed only portions of the contents, instead of entire videos. With the use of multicast for delivery of the contents, a portion that is pushed to one server is also pushed to the rest of the servers in the same level. Thus, all the pushed portions can be assembled and used for streaming the remaining parts of the video. An example of transferring of 4 different streams of a replica that has to be repositioned to other servers after the execution of the algorithm is given Figure 3.6. When the streaming is interrupted by the execution of the algorithm (marked with dotted line), all 4 streams have different portions of the video left until the end. The difference between these portions is approximately 1/5 of the entire video, which can be clearly seen in the right part of figure showing the same streams from the left side, shifted in the time as they started in the same moment. The first stream c_1 has 4/5 of the video left and the last stream c_4 has only 1/5 left. If the algorithm decides that the content should be moved to other servers, the same content should be simultaneously redistributed 4 times, starting from the position of interruption in each different case, so that the interrupted streams can continue from the same position on the new locations. However, it is sufficient that only 1/5 of the video is distributed for each stream, starting from the point of interruption. After the simultaneous multicast of each 1/5 of video necessary for uninterrupted streaming of

the 4 streams, the entire content will be available on the new locations. The algorithm also multicasts the first 1/5 of the video that is already delivered to the clients by all 4 streams before the interruption so that the new requests for that content can be served from the beginning. With this approach, the algorithm provides immediate transfer of the current streams to the new locations, and at the same time, it provides immediate distribution in short time. This kind of distribution is a burden for the CR server at the moment of the execution of the algorithm, and therefore, the number of different contents that will be immediately transferred will depend on the streaming capacity of the CR server and on the capacity of the links that inter-connect the CR server with the streaming servers.

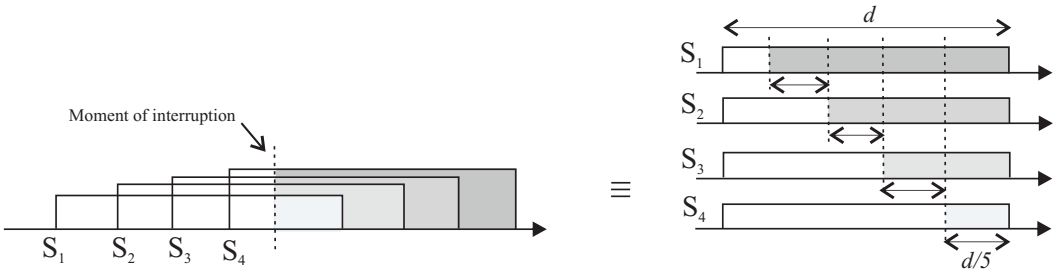


Figure 3.6.: Time representation of 4 streams of the same content in the moment of interruption by the algorithm and representation of the same strips shifted in the time as they started in the same moment.

3.5. Redirection strategy

The objective of the redirection strategy is to redirect each request to the appropriate server so that the streaming traffic is concentrated in the lower levels of the network and the traffic between the servers in a same level is well balanced. Upon a request, if there is more than one server that contains the required content, the SS server calculates a redirection metric for each server s that contains a replica of the content item c and chooses the one that minimizes the metric. If the client belongs to a group g, the redirection metric $m_{ss}(s, c, g)$ is calculated as:

$$m_{ss}(s, c, g) = d(s, g)u(s)\frac{p(s, c)}{p(c)} \tag{3.12}$$

where $d(s, g)$ is the distance between the candidate server s and the group g the client belongs to. The distance of the server from the client is important for redirecting the clients to the servers that are closer to them. Another important value for the redirection metric is the utilization of the candidate server $u(s)$ which, in the given expression, gives preference to the less utilized servers. In order to balance the load generated by the replicas of the same

contents between the servers, the metric includes the percentage of served streams of a single replica by a server relative to the total number of served streams for that content item.

3.6. Simulations and results

This section presents the experimental results obtained from the implementation of the redistribution algorithm in the proposed network model for distribution of VoD contents. For that purpose, a simulation model was developed in the discrete event simulator OMNeT++ (Varga & Hornig, 2008), using the implementation of the network protocols defined in the INET library (Varga, 2013). In the simulation model, each entity is presented as a node in the network that communicates with the other entities by exchange of messages. The video streams are also simulated as exchange of data packets. The messages are sent over bidirectional links with capacities defined by the operator.

The simulation model consists of $S = 13$ streaming servers that are serving $n = 4000$ clients. Following the architecture of a network for IPTV services (Simsarian & Duelk, 2007; Stockhammer & Heiles, 2009), the servers are organized in tree hierarchy of $L = 3$ levels where each level $l = 1$, 2 and 3, contains 10, 2 and 1 servers, respectively. The streaming capacities of the servers, in the same order of levels, are $U(s) = 250$, 500 and 1000 Mbps, accordingly. The trigger threshold of each server s is $T(s) = 90\%$, which determines the utilization of the uplink capacity above which the server is considered busy. The links that interconnect the servers have enough capacity to support the maximum streaming load of all the servers. The streaming servers host $C = 1500$ Standard Definition (SD) quality contents with playback rate $r_s = 2$ Mbps and average duration $\overline{d} = 90$ min. The streaming servers have also limited storage capacity which, in order of levels, is $M(s) = 25\%$, 35% and 40% of the entire video library. These values are obtained according to the popularity of the contents and the streaming capacity, assuming that the most popular contents will be stored in the lowest level, so that both streaming and storage resources are optimally used. The process of generating requests for videos is a Poisson process with average inter-arrival time $w = 1/\lambda$ =20 min.

The popularity of the video contents is modeled according to the Zipf-Mandelbrot (ZM) distribution (Krogfoss *et al.*, 2008). This distribution is a modification of the Zipf distribution, commonly used for modeling popularity of web pages (Breslau *et al.*, 1999), obtained by the introduction of a shifting constant q:

$$P(v) = \frac{(v+q)^{-\alpha}}{\sum_{k=1}^{c}(k+q)^{-\alpha}} \tag{3.13}$$

This expression gives the relative frequency of the v-th ranked video out of C videos in the library, where α is a skew factor which indicates the dispersion between the popular and not popular contents. When the skew factor is $\alpha = 0$, the distribution becomes uniform. The ZM distribution is used to better describe the specific behavior of the clients when they request video contents: client that already watched one video will not repeat a request for the same video. This causes the steepness of the Zipf distribution curve to reduce for the popular videos, which is achieved by adding the shifting constant q. For values of $q = 0$, the ZM distribution becomes a Zipf distribution.

In the literature, the value of the skew factor obtained from trace data of systems with different sizes of the content library varies between 0.2 and 1 (Yang & Fei, 2003; Yu *et al.*, 2006; Chesire *et al.*, 2001). In the simulations, it will be considered that the skew factor is $\alpha = 0.8$ and the shifting constant is $q = 10$ (Borst *et al.*, 2009). Using this distribution, a Random Number Generator (RNG) is designed to generate an integer number within a range defined by the number of video contents, every time a client schedules request for a video. The relative frequency of appearance of the videos obtained as a result of the RNG after running the simulations sufficiently long time is shown in Figure 3.7. The figure shows that only a small portion of the first most popular contents of the entire library is responsible for the highest number of requests.

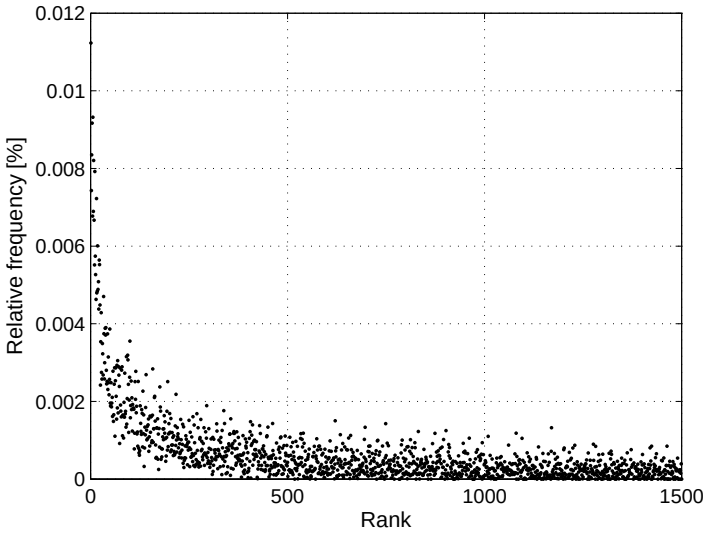


Figure 3.7.: Relative frequency obtained from a RNG according to a ZM distribution with $q = 10$ and $\alpha = 0.8$.

The first simulation scenario consists in running the algorithm when the video contents are randomly distributed on the servers. Then, at the time $t = 500$ min, the algorithm is run, and the contents are redistributed in the servers according to their popularity and according to the available resources of the servers. The time that the clients have to wait for initiating

a stream from some of the servers during the simulation is shown in Figure 3.8. From the figure, it can be seen that before the algorithm is run, there is a considerate delay in the service. The reason for such a poor QoS is the non-optimal distribution of the contents on the servers. Some of the servers are overloaded, while other servers are underloaded, and therefore, although there are sufficient resources for serving all the requests, a considerate number of the clients are denied for immediate service. After the algorithm is run, the contents are placed in the servers according to their request patterns. Hence, there are no denials for immediate service.

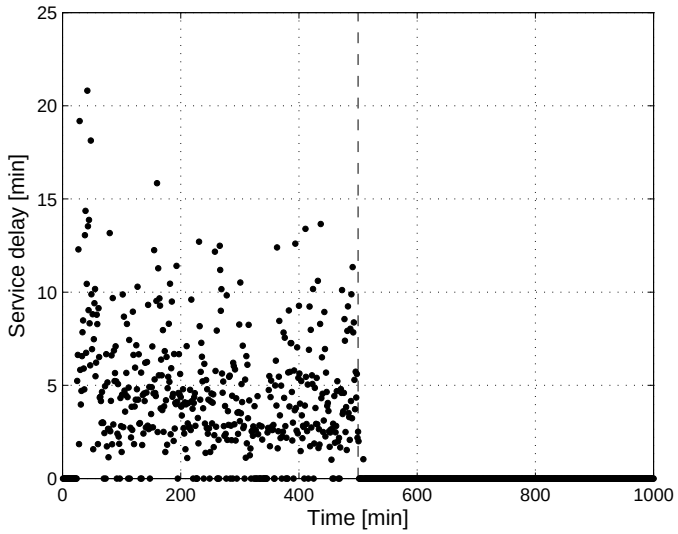


Figure 3.8.: Service delay of the requests before and after running the redistribution algorithm at time $t = 500$ min with random initial distribution of the contents.

Figure 3.9 gives an inside view of the system by presenting the streaming traffic that originates from each of the levels in the hierarchy before and after the execution of the algorithm. The traffic is presented as a percentage of the overall streaming capacity of the system. In the initial phase of the simulation, the servers in the higher levels are overloaded, which is the reason for the high number of denied requests for immediate service. Since each of the levels $l = 2$ and 3 have 25% participation in the streaming traffic, and the trigger threshold of their servers is 90%, in the initial phase they are used up to this level. The level $l = 1$ has a participation of 50% and is very poorly used before the execution of the algorithm. From the figure, it can be seen that after the execution of the algorithm, the traffic of the poorly-utilized servers in the first level immediately begins to ascend, which alleviates the overloaded servers in the higher levels. Although the branch servers in the level $l = 2$ are alleviated, their traffic is reduced only to the point from which there will be no denials for service, keeping them maximally utilized. As a result of the optimal utilization of the streaming capacity of the first two levels, the traffic in the highest level considerably reduces. This

behavior shows that the system tends to maximally utilize the levels that are closer to the clients and to reduce the traffic in the core of the network.

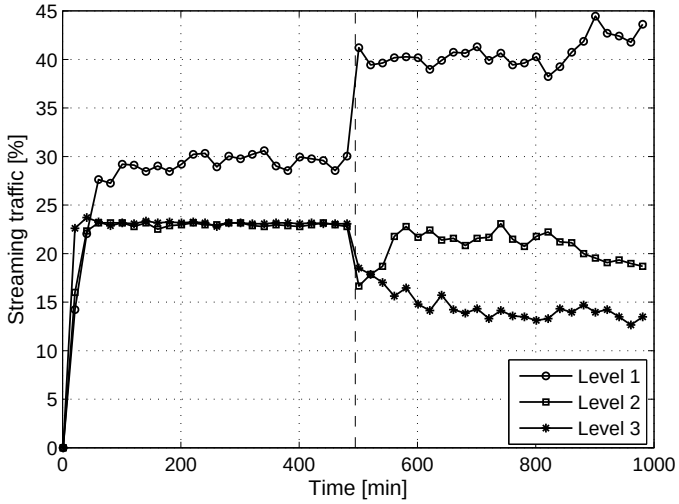


Figure 3.9.: Percentage of the overall streaming traffic streamed by the servers in the levels $l = 1$, 2 and 3 before and after running the redistribution algorithm at time $t = 500$ min for random initial distribution of the contents.

An important measure defined for the estimation of the performance of the redistribution algorithm is the transport cost of the streaming traffic from the streaming servers to the clients. It is based on the distance of the servers from the clients and their current load and it is expressed as:

$$\text{Cost} = \sum_{s=1}^{S} l(s)u(s)U(s) \tag{3.14}$$

where $l(s)$ represents the level of the server, which is actually the distance of server s from the clients it is serving, and $u(s)U(s)$ is the current streaming rate of the server.

In order to compare the current cost of the system to an optimal value, towards which it tends to converge, a reference minimum cost function is defined. The value of this measure depends on the current traffic demand in the system and is calculated as the minimum cost that has to be paid for streaming the demanded traffic in a given time. The calculation of this value is shown in Algorithm 3.3. In the algorithm, first, the overall streaming traffic is calculated as a sum of the traffic of each server in the system (lines 3-5). Then, starting from the servers in the lowest levels, the maximum desired occupancy of each server is calculated as the amount of the threshold traffic on the server since the goal is to utilize the lower servers as close to their threshold as possible (line 7). This value is then multiplied by the distance of the server from the clients and is added to the minimum cost (line 8). If possible, the

remaining traffic is reduced by the maximum traffic added to the minimum cost (line 9) and the procedure is repeated until all the streaming traffic is depleted.

Algorithm 3.3 Algorithm for calculation of minimum traffic cost.

```
1  MinCost(t)=0
2  traffic(t)=0
3  for s=1;s≤S;s++
4      traffic(t)=traffic(t)+u(s,t)*U(s)
5  end
6  for s=1;s≤S;s++
7      maxTraffic(t)=T(s)*U(s)
8      MinCost(t)=MinCost(t)+d(s)*min(maxTraffic(t),traffic(t))
9      traffic=max(0,traffic−maxTraffic)
10 end
```

Figure 3.10 shows the percentage of additional traffic cost relative to the minimum cost that has to be paid for streaming the video contents throughout the simulations. At the beginning, the cost is higher because the contents are not optimally distributed in the servers, however, after the algorithm is executed, the cost reaches values close to the minimum cost. The results indicate that the algorithm contributes to optimal content distribution according to their popularity, which leads to considerably lower traffic costs.

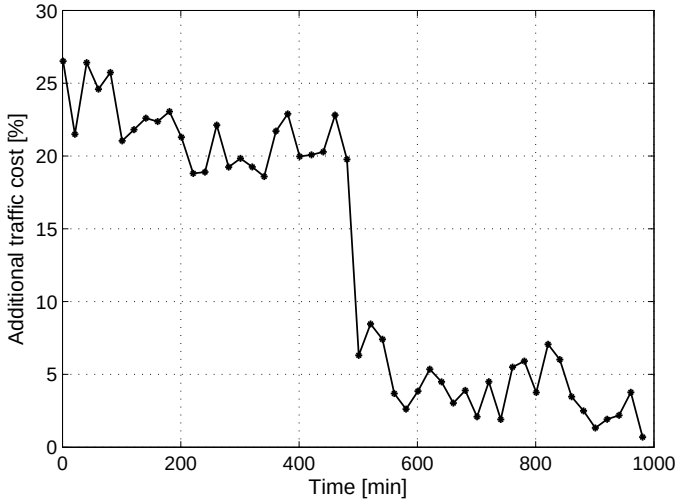


Figure 3.10.: Additional traffic cost relative to the minimum cost before and after the execution of the algorithm at time $t = 500$ min with random initial distribution of the contents.

Although the algorithm contributes to many improvements in the system, like QoS and reduced streaming cost, its main downside is the additional distribution traffic in the core of the network which is generated for placement of the replicas in the servers. The distribution

traffic expressed as a percentage of the overall streaming capacity of the system is presented in Figure 3.11. The peak value of the additional traffic at the beginning of the simulation is a result of the fact that in the random distribution of the contents, not all of the contents are available on the servers. Therefore, when a request for unavailable content is made, the CR server pushes a replica of the required content in the server in the highest level, which then streams the video to the clients. The traffic of the CR server rapidly reduces because, as the time passes, most of the unavailable contents are already pushed to the streaming servers.

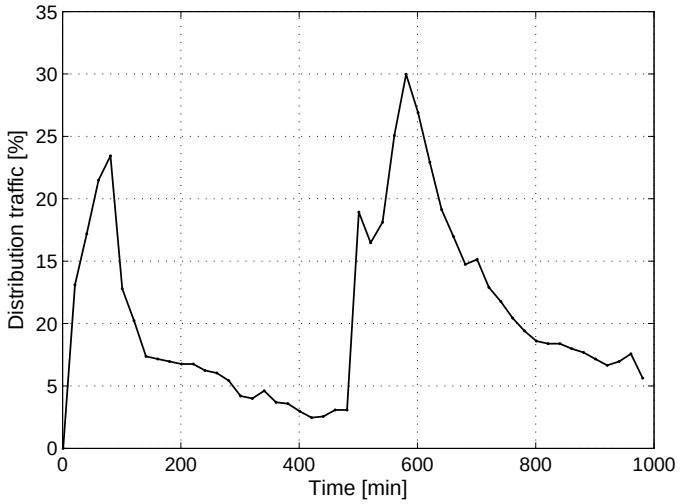


Figure 3.11.: Percentage of distribution traffic in the network relative to the overall streaming capacity before and after the execution of the algorithm at time $t = 500$ min for random initial distribution of the contents.

Another peak of the distribution traffic is present immediately after the execution of the algorithm at $t = 500$ min. The distribution traffic immediately increases because the streams of the most popular contents are transferred to their new locations so that the busy servers are immediately alleviated. Although the number of these popular videos is small (in the simulations, this number is 50), each content requires as many multicast streams as there are current streams at the moment of the execution of the algorithm. The second peak that follows shortly after the discussed peak for multicast of the contents is a result of the requests for replicas that are assigned to be placed on new locations. In order to avoid instant distribution of all the replicas with new location, the system only allocates resources for storing the replicas, which will be physically stored only when a client makes a request for them. From the figure, it is clear that the distribution traffic reaches considerably high levels, but one should bear in mind that, initially, the contents are randomly distributed without considering the popularity of the contents. Although the random distribution is not common in the real VoD systems, which consider distribution based on the previous history of the clients' behavior, the aim of these simulations is only to show the responsiveness of the

algorithm and its capability of fast achieving optimal placement of the contents.

Another proof of the efficiency of the algorithm is the next simulation scenario where the contents are already optimally distributed according to the past behavior of the clients, however, at a certain moment of the time, the popularity of the contents changes. In the simulations, the popularity rapidly changes in the moment $t = 300$ min. This popularity change is effectuated by swapping the first 100 videos with the videos that are 300 positions less popular. Then, at time $t = 600$, the algorithm is run to respond to the popularity change and redistribute the contents according to their current popularity. Figure 3.12 shows the time the clients have to wait for service throughout the simulations. At the beginning, the clients receive immediate service due to the optimal distribution of the contents. However, as the popularity changes, the servers that contain less popular contents become overloaded because the contents they are hosting become more popular, and therefore, they reject some of the requests until some of the current streams are released and more streaming resources are provided. After the algorithm is run, it can be noted that there are no denials, i.e., all of the following request are immediately served.

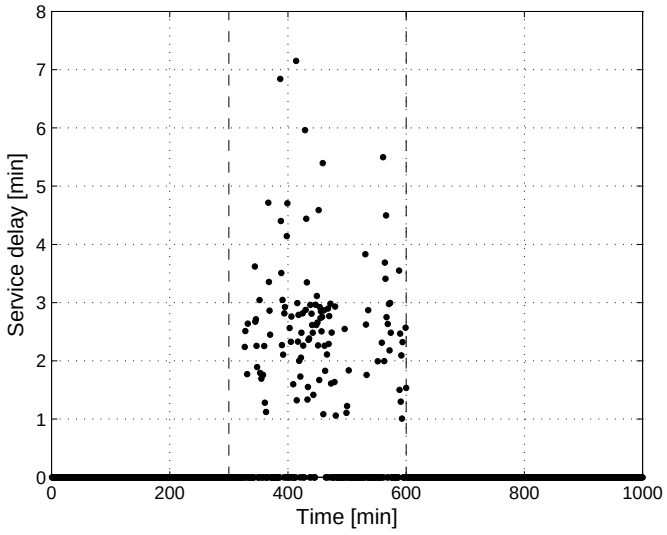


Figure 3.12.: Service delay of the requests for optimal initial content distribution with popularity change ($t = 300$ min) before and after running the redistribution algorithm ($t = 600$ min)

A better explanation of these previous results can be found in Figure 3.13, which shows the streaming traffic in all the levels of the hierarchy. Initially, the largest part of the traffic is streamed by the first two layers, but, as the popularity of the contents changes, the servers in level 1 get less utilized, and consequently, the traffic that originates from the servers in the remaining levels increases. The high number of denials for immediate service is a result of the fact that with the optimal initial distribution, the branch servers ($l = 2$) get almost

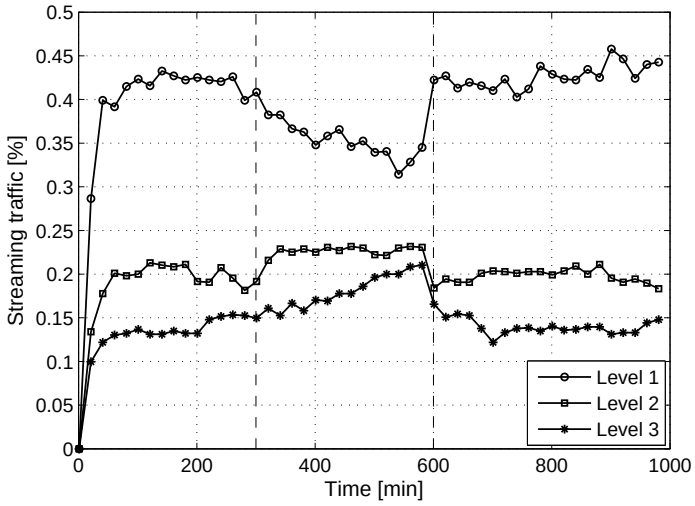


Figure 3.13.: Percentage of the overall streaming traffic streamed by the servers with popularity change before and after running the redistribution algorithm for optimal initial distribution of the contents.

completely utilized, and the change of the popularity redirects more request for the videos they host. Unable to respond to the new increased demand, the servers reject a considerate part of the requests. After the execution of the redistribution algorithm, which responds to the current state of the system and the popularity of the contents, the traffic in all the servers converges to the previous optimal levels that the servers had before the change of the popularity of the contents. The figure also shows the high responsiveness of the system, since the servers immediately reach the desired optimal levels.

It is important to mention that although there are visible changes after the first execution, the algorithm is executed whenever the threshold of some of the servers is reached. In this particular case, there are no significant changes in the popularity, and therefore, there are no changes in the distribution obtained with the first execution.

Figure 3.14 shows the effects of the new popularity on the streaming traffic cost and the contribution of the algorithm to the reduction of this cost to values close to the minimal cost. The change of the popularity, implies a significant increment of the additional streaming cost relative to the minimal cost. After the execution of the algorithm, the traffic is optimally distributed on the streaming servers, which consequently reduces the traffic cost.

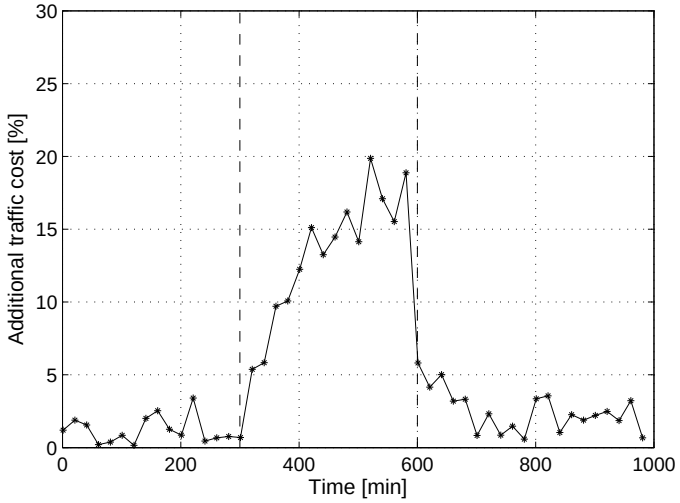


Figure 3.14.: Additional traffic cost relative to the minimum cost with popularity change before and after the execution of the algorithm with optimal initial distribution of the contents.

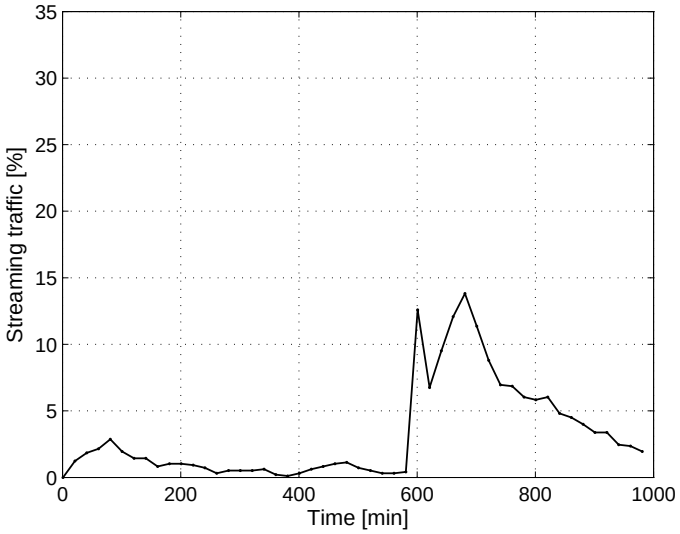


Figure 3.15.: Percentage of distribution traffic in the network relative to the overall streaming capacity with popularity change before and after the execution of the algorithm for optimal initial distribution of the contents.

The additional traffic that is generated in the network for distribution of the replicas to their new locations is shown in Figure 3.15. Initially, there is only a negligible amount of distribution traffic for replicas of contents that are not stored in any of the streaming servers. These are the replicas that might have been assigned by the ACM server to be stored in some of the servers, however, none of the clients made a request for them. The popularity change has no influence on this traffic because all the contents that are affected by this change

are already stored in the servers in the levels close to the clients. The distribution traffic significantly increases after the new distribution of the contents is obtained. In order to get immediate response of the system to the new distribution, the first most popular videos are first transferred to the new servers closer to the clients, and then, the rest of the contents assigned to new locations are distributed as they are requested by the clients. This additional traffic is the cost that has to be paid for the immediate service the clients obtain, despite of the change of the popularity of the contents.

Another feature of the algorithm is its responsiveness to the change of the intensity of the traffic. In the following simulation scenarios, the inter-arrival time of the clients is set to value $w = 1/\lambda$=50 min at the beginning of the simulations, and then, at time $t = 300$ min, the inter-arrival time is decreased to $w = 1/\lambda$=15 min which implies higher traffic demands in the system. At time $t = 700$ min, the inter-arrival time is set to the initial value again. The contents are previously optimally distributed on the servers. In order to see the influence of the increased intensity of the requests on the system's performance, the simulation is first run without execution of the algorithm. The service delay of the rejected clients of this simulation scenario is shown in Figure 3.16. As the number of requests increases, the servers become overloaded and cannot serve all the clients with the current distribution of the contents. Running the same simulations with execution of the algorithm showed that there are no denials in the system at all.

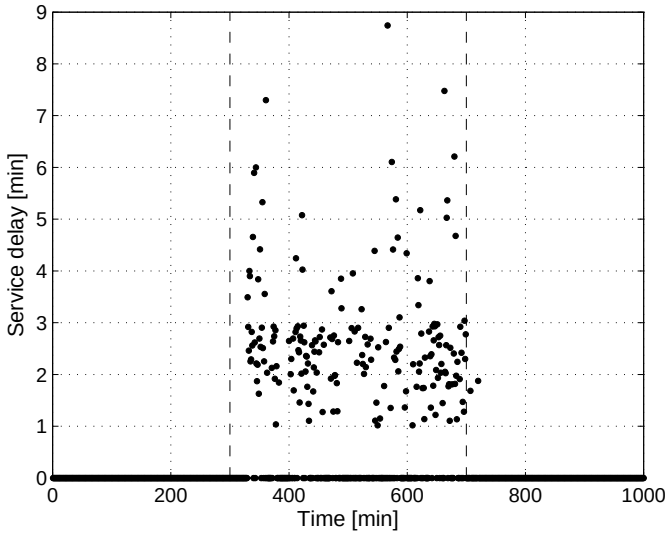


Figure 3.16.: Service delay of the requests for increased intensity of the requests without running the redistribution algorithm.

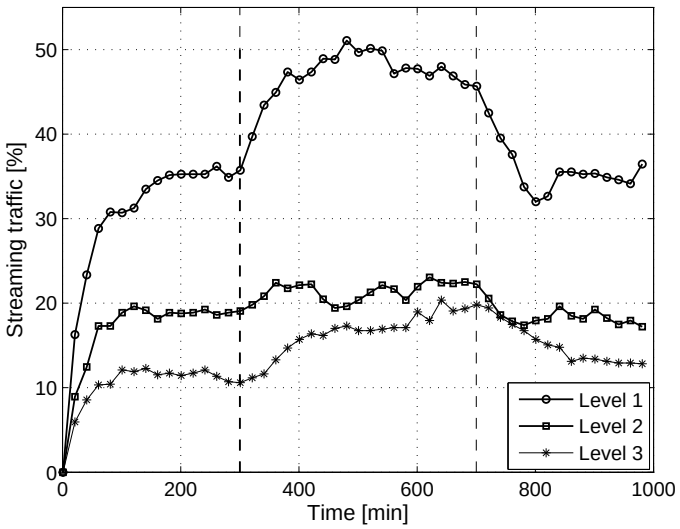


Figure 3.17.: Percentage of the overall streaming traffic streamed by the servers in the levels $l = 1$, 2 and 3 with change of the intensity of the requests as a result of the execution of the redistribution algorithm.

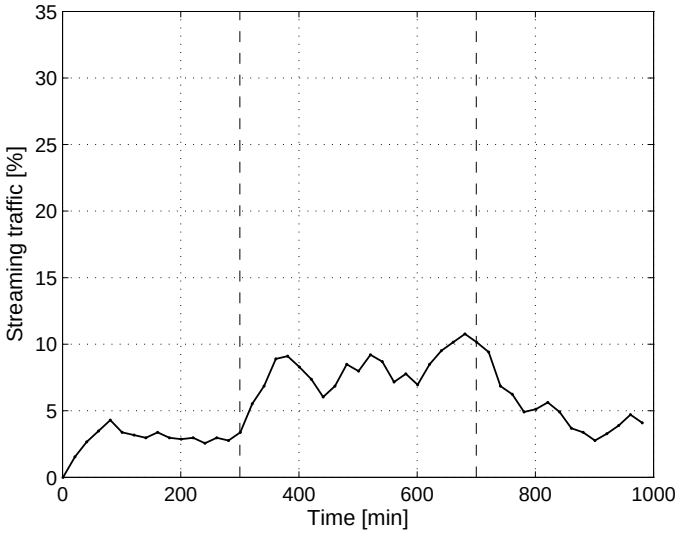


Figure 3.18.: Percentage of distribution traffic in the network relative to the overall streaming for distribution of contents with change of the intensity of the requests.

Figure 3.17 shows the traffic in each of the levels of the hierarchy in the scenario when the redistribution algorithm is executed. It can be seen that after the intensity change of the requests, the first two levels get maximally utilized, but as the intensity of the requests reaches such a level that more streaming resources are necessary, the traffic of the last level also begins to increase. In the case of increased request rate, the algorithm reserves more

resources for the popular contents, and whenever the streaming capacity is exploited, the less popular contents that were previously in this level are moved upwards in the hierarchy.

The additional traffic introduced in the network for redistribution of the contents due to the increased intensity of the traffic is shown in Figure 3.18. From the figure, it can be seen that within the interval of more intensive requests, the additional traffic increases which is a result of the redistribution and the transfer of the streams of some of the contents. Although the execution of the algorithm introduces more additional traffic, it is acceptable price that has to be paid for the clients to get immediate service.

3.7. Conclusions

In this chapter, a hierarchical system for optimal streaming and distribution of VoD contents in managed networks is proposed. The system implements a redistribution algorithm that uses the current demand of the content items and the state of the network to take distribution decisions that optimize the network utilization and improve the QoS received by the clients. The system additionally implements a redirection strategy which keeps the servers balanced and the traffic to the edges of the network. It is also proposed an estimation method for determining the streaming demand for the replicas in the system that reduces the management traffic in the network.

The experimental results proved that the proposed system reaches the defined objectives for improved QoS and optimal network utilization. It redistributes the content items according to the request pattern and the state of the server, and thus, starting from a random distribution of the content items, it achieves a fast convergence to an optimal distribution. The system is also responsive to popularity change of the contents, and by means of the algorithm, it redistributes the contents according to the new popularity. The advantages of the outcome of the redistribution algorithm and the redirection strategy are numerous: an immediate service of the clients is achieved, the traffic is concentrated in the edges of the network, thus providing less congested network, a better utilization of the network resources and lower traffic cost. The optimal utilization of the resources improves the scalability of the system because it brings the possibility for serving higher number of clients. The price that has to be paid for these improvements is the overhead traffic generated for distribution of the replicas among the streaming servers. However, this traffic occupies only insignificant part of the overall traffic in the network when there are no significant changes in the popularity of the contents.

The proposed solution contributes to better distribution of the streaming traffic among the servers, and any further improvements of the algorithm would bring only insignificant changes in the streaming traffic in the edge of the network. No matter how well one redistribution

algorithm might perform, it cannot reduce the overall streaming traffic, but can only redistribute it from one location of the network to another. The optimal placement of the replicas is an acceptable solution for a network with constant size. However, the growth of the number of customers would also require upgrades in the network, or otherwise the service would be deteriorated. The upgrade of the system consists in installation of new streaming servers and increasing the capacity of the links that interconnect them, which is expensive solution for the operators. Therefore, the next level of improvement of the system is taking advantage of the clients' resources and including them in the streaming process. The implementation of the P2P concept and its contributions to the system for VoD streaming is treated in details in the chapter that follows.

Chapter 4

General Model of Peer-assisted VoD Streaming

One of the solutions for reducing the high amount of traffic that is generated by the VoD service in the core of the managed networks is implementing a strategy for optimal placement of the video contents in the streaming servers. An example of such a solution was presented in the previous chapter, which consists of a specific hierarchical architecture and a redistribution algorithm that takes into account the behavior of the users and the state of the servers. However, any effort for designing better solutions based on placement of the contents, reaches a point from where no further improvements can be made due to the resource limitations of the system. This chapter addresses the issue of including the peers in the distribution of VoD contents. It proposes extension of the model developed in Chapter 3 by taking advantage of the client's unused uplink and storage capacity to serve the requests of other clients. Because of the limited resources of the peers for a reliable streaming, the peers are only partially included in the streaming process, whereas the servers still have the main role. This solution requires that the videos are partitioned into strips, which add more complexity to the system. As the peer-assisted streaming is already an accepted concept which gives satisfactory results, this chapter focuses on the impact of the distribution of the contents on the peers on the reduction of the traffic in the core of the network. Based on the optimal distribution of the contents on the servers, it proposes popularity based schemes for distribution of the contents on the peers. These schemes tend to place both the popular and not popular contents on the peers in such a manner that would localize the traffic in the periphery of the network, keeping the satisfactory level of QoS and traffic cost. The chapter analyzes the impact of some system parameters and the distribution schemes on the QoS, the reduction of the traffic cost and the scalability.

4.1. Model description

One approach that contributes to reducing the server traffic beyond the limits of the resources is the implementation of the P2P concept in the distribution of the VoD contents. This concept can be easily applied to the delivery of VoD contents, since the clients subscribed to the service have an STB. Apart from its basic function to decode the stream and buffer portions of the video for uninterrupted streaming, the STB has also a storage capacity for storing past TV programs or some of the videos already viewed by the clients. This is certainly a great advantage since this space can be used for storage of the videos that can be streamed to the other peers. Another reason for including the peers in the streaming process is that a significant part of their uplink capacity is rarely entirely used because most of the traffic that the clients generate is incoming traffic that makes part of the downlink capacity of the connection between the client and the provider. However, although their uplink capacity can be used for P2P streaming purposes, it is not enough for completely serving an entire video because with the current network technology, the maximum uplink capacity is lower than the playback rate of a Standard Definition (SD) quality video. Therefore, the streaming is done by parallel streaming sessions of multiple STBs in order to compensate for their limited streaming capacity. Unlike many P2P solutions where the peers self-organize themselves, in the proposed model, the peers have a role of passive contributors to the streaming process, having no knowledge of the existence of other peers. They are only capable of serving the videos that they have already stored. All the decisions regarding redirection are taken by the servers at the edge of the network.

Despite of the numerous advantages of the P2P concept, its main issue is the reliability since the peers in the Internet community are not always willing to share their resources. However, this work addresses the privately managed network were the operator has control over the STBs. The control over the STBs enables the provider to avoid the reliability issue by reserving certain amount of storage and streaming resources for streaming of the videos.

The idea of cheap and unused resources is used for building a hybrid model for peer-assisted streaming as an extension of the previously described system for distribution of VoD contents. The purpose of the P2P is not to completely substitute the servers by the peers, but only to reduce the traffic generated in the core of the network. Relying on the previous system where the contents are distributed on the servers according to their popularity, the reduction of the traffic in the core of the network and the overall traffic cost, will largely depend on the distribution of the contents among the peers. Storing copies of the popular contents in the STBs is quite a reasonable solution that could significantly reduce the traffic in the edge of the network, particularly in the busy hours. However, there is a large number of contents that are not in the high popularity range, but still take significant part of the overall

traffic. Since they are stored in more distant servers in the core of the network, the traffic generated for their streaming is a burden for the backbone of the network. The opposite case of distributing the not popular contents on the STBs contributes to reducing the traffic in the core of the network because it concentrates most of the traffic in the periphery of the network: the popular contents are streamed by the servers at the edge, and a great part of the unpopular contents are streamed by the STBs. This is important when one of the objectives is reducing the transport cost in the network. Although both of the distributions bring improvements by reducing the overall traffic, they do not provide improved service for the entire set of contents in the busy hours. When the popular contents are stored in the STBs, the response time for service of the not popular contents is increased because the servers cannot serve all the incoming requests. The same happens when the not popular contents are stored on the STBs with the difference that now, not all the requests for popular contents can be immediately served. Therefore, this chapter proposes a solution for a network with popularity based distribution of contents, both on the streaming servers and STBs, which aims to reduce the traffic in the core of the network, and at the same time, tends to provide immediate service in the cases of high demand scenarios. One of the objectives targeted with the reduction of the traffic in the core is alleviating the backbones from video traffic so that it can be used for other type of traffic and enabling growth of the number of clients subscribed to VoD service without significant additional changes and costs in the core of the network. Although the schemes that are proposed consider all the contents, the accent is mainly put on the low popularity contents by reserving more storage space in the STBs than the popular contents, aiming to provide locally close availability of the entire video library.

The proposed solution is a hybrid system for peer-assisted streaming of VoD contents that unites the advantages of both the IPTV and P2P architectures: the high reliability and scalability of the IPTV architecture and the cheaper and unused storage space and uplink bandwidth of the P2P architecture. The model is extension of the model for VoD streaming in the previous chapter obtained by including the clients as streaming entities. It consists of hierarchical structure of VoD servers, management servers and STBs. The management servers are responsible for monitoring the system and taking decisions about redirection of the requests and the placement of the contents. Since the accent of this chapter is on the peer-assisted streaming, the function of the management servers will not be considered and it will be assumed that the contents on the servers are distributed according to their popularity. The main streaming functionality is provided by the VoD servers. The clients, apart from requesting videos, can also assist in the streaming process taking advantage of the storage space of their STBs and their unused uplink capacity.

The VoD servers are placed at distinct points of the private network, organized in a hierarchical tree-structure (Figure 4.1). These servers have limited storage and streaming capacity,

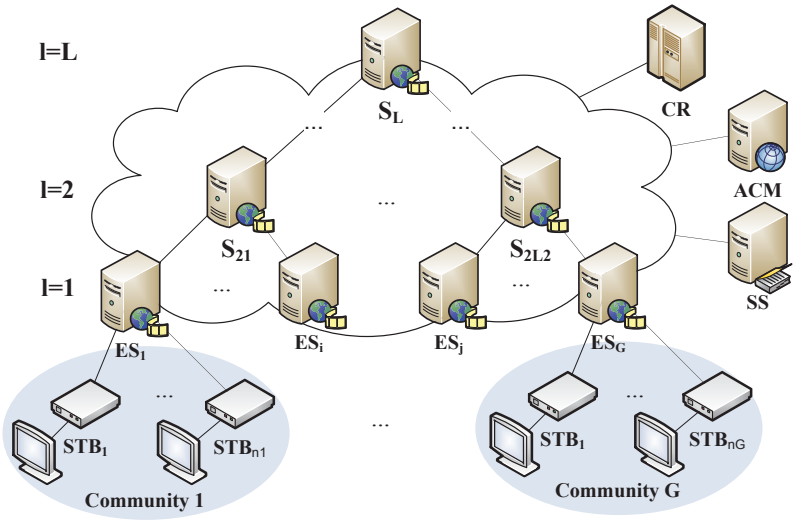


Figure 4.1.: Model architecture with hierarchically placed VoD servers.

so they can host a limited number of contents and can serve a limited number of clients. From all the VoD servers, the servers placed at the edge of the network, also called Edge Servers (ES), have a particular importance in the process of peer-assisted streaming. These servers are placed in a Central Office (CO) and form local communities with the clients that are connected to the same DSLAM or access network. Unlike the system described in the previous chapter, the initial requests from the clients are directed to the ES. The ES receives the requests from the clients in the local community and, depending on the availability of the requested video, decides how the contents will be delivered to the client. The basic functionality of the ES is to stream videos to its local community. It cannot serve clients from other communities because the tree-structure of the architecture implies longer distances between two communities, and thus, more traffic in the inner parts of the network for streaming the videos. Another important functionality of the ES essential for the implementation of peer-assisted streaming is its role as an index server within its local community. It maintains availability data of the contents stored in the peers and data related to the available uplink capacity of the peers. The ES uses this data to redirect the clients whenever there are requests for contents that are already stored in the peers with available streaming resources. The new role of the ES as an index server requires additional computing power which implies hardware upgrades of the ordinary streaming server and additional cost for the operator. The management messages also create traffic originating from the ES that is considered negligible compared to the amount of the video traffic. Another issue addressed by the role of the index servers is the increased response delay which is considerably shorter and of little importance compared to the duration of the video contents.

The clients from one local community cannot assist in the streaming of the clients in other

communities because the cross-community traffic would cause additional burden in the core of the network. The contents are distributed on the peers during the off-peak hours by the operator or they are simply assigned to be stored after the clients watch them. Another important feature that is taken into consideration when distributing the contents on the peers is their volatile nature of the popularity. This property comes as result of the behavior of the users not to repeat a request for the same content. Soon after a video is introduced in the system, it reaches high popularity, but as the time passes, the popularity decays because the clients who already saw the video are unlikely to request it again. Therefore, a content item that is already viewed and stored in the STB of many clients is very likely to be later removed from the ESs as not popular. In such a way, most of the contents with reduced popularity will be already stored in the STBs and available for streaming. This saves a lot of additional traffic for distribution of the contents from the streaming servers to the STBs.

4.1.1. Content division

The division of the contents into smaller strips is inevitable for the implementation of the peer-assisted streaming. The main reasons for the necessity of division are the limited uplink capacity of the STBs and the real-time nature of the VoD service. Since the uplink capacity of the STBs is several times smaller than the play-back rate of the videos, there will be interruptions in the play-back if it is streamed by only one STB. The solution for overcoming this issue is including more STBs in the streaming of the same video and the division of the contents into parallel strips so that each included peer could participate in the streaming with its available capacity. The partitioning of the contents (Chen *et al.*, 2010; Muñoz Gea *et al.*, 2012) is an already accepted practice used for streaming of videos from many sources with limited streaming resources. For immediate and uninterrupted playing, a content item has to be streamed in parallel by as many peers as it is necessary for reaching its playback rate. The partitioning of one video into m parallel strips is shown in Figure 4.2.

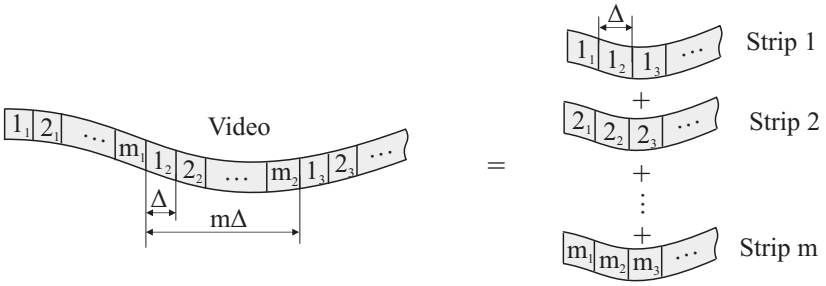


Figure 4.2.: Division of a videos into m parallel strips.

The size of the streamed portion Δ is determined according to the maximum acceptable initial viewing delay. Each strip consists of consequent streaming portions that are on distance $m\Delta$

between each other, where m is the number of strips that compose one video. Depending on its capacity, each peer can stream at least one strip of the content item. When all the strips reach the client, they are assembled and the content is played. Figure 4.3 shows the streaming process of the videos divided into strips and the assembly of the strips on the clients' side. From the entire video, each STB in the figure, streams only the portions that belong to one strip, i.e., only those portions that have distance $m\Delta$ between each other. The rate necessary to stream one strip is also referred to as channel and is the m-th fraction of the play-back rate of the video.

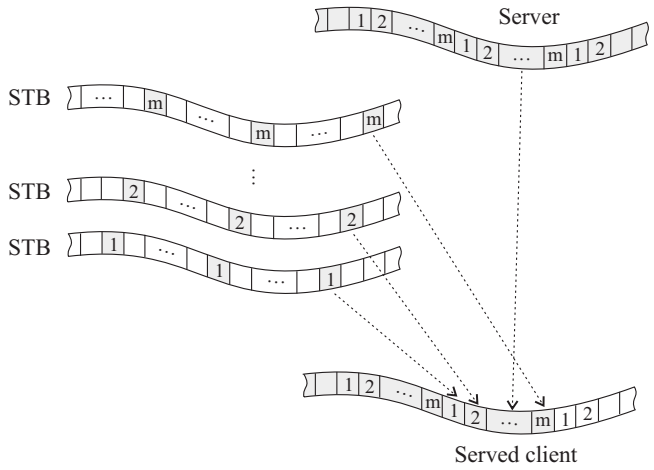


Figure 4.3.: Streaming process of the videos divided into strips.

The division of the contents also contributes to increasing the storage efficiency of the peers and the contents' availability. The fact that each peer is capable of streaming only a portion of the content makes it reasonable to store only those portions that it is capable to stream. Since the strips are m times smaller in size than the original size of the content, each peer can store m times more strips of different content items. All the contents that are stored in the STBs are entirely stored in the servers, so that they can be delivered whenever the STBs are not able to provide any of the strips.

4.1.2. Request handling process

Including the peers as participants in the streaming process adds complexity in the process of serving the requests because before the client is redirected to be served by the servers, the index server has to check whether part of the streaming can be effectuated by the peers. Therefore, the index server is added as an entry point for the requests, unlike the previously considered system for pure server streaming where the requests were directed to the SS server. The entire process of handling a request for a video content is shown in Figure 4.4.

The process of serving a request for a video is initiated by a client that makes a request to

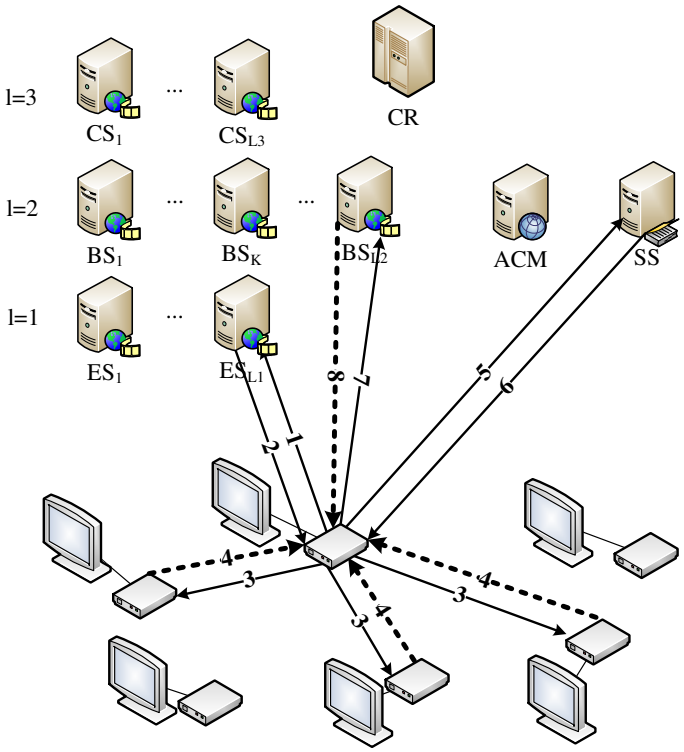


Figure 4.4.: Request handling process of a content that can be partially served by the peers.

its designated ES (1). Each request for one content is treated by the ES as m independent requests for strips since the contents are distributed on the peers as strips and each peer contains different strips of the videos. The ES first checks whether it can find peers with available channels that contain at least one of the requested strips. If there are peers that fulfill these conditions, for each strip that can be served by a peer, the ES adds an entry in an availability list. The entry contains the address of the peer and the number of the strip. If there is not a sufficient number of peers to stream the entire video, the ES then initiates a process for finding the most appropriate server to stream the remaining strips. It first checks whether it contains a copy of the content to serve the request itself, and if so, it adds up its address to the availability list. In the opposite case, the ES has no information about which server could serve the requested item and therefore, it directs the client to repeat the request to the SS server. Consequently it adds the address of the SS server at the end of the availability list. Afterward, the ES sends the availability list to the client (2).

Upon reception of the list, the client makes a request for the designated strip from each peer in the list (3) and initiates a parallel streaming session (4) with the peers. In the case when not all the strips are available on the peers, the peer checks on the address of the last item in the availability list. If the indicated address is the address of the ES, it initiates a streaming session with the ES. The streaming session consists of one stream that contains the video

data from all the missing strips. The streaming rate of the strip is the number of the missing strips multiplied by the streaming rate of one channel. If the ES does not have a copy of the required content, the client makes a request to the SS server (5) which chooses the best server that has available copy of the requested video and sends its address to the client (6). Then, the client repeats the request for the missing strips to the designated server (7) and initiates a streaming session (8). The video is played when the data from all the parallel streams is assembled into one stream.

Another typical case of handling an incoming request is the scenario of congested network, when all the servers that contain a copy of the video are overloaded. In that case, the client's request is denied and it is asked to repeat the same request after a random time within a defined range of values. If there are available strips on the peers but the servers are busy to serve the rest of the strips, the client will not initiate a streaming session with the peers because the video cannot be played until all the strips are provided. In that case, the process of handling the next request is identical to the one when the client requested the content for a first time.

4.2. System parameters and dimensioning

The model for peer-assisted VoD streaming is an extension of the basic, pure-server model that consists of S streaming servers which belong to one of the L levels of a tree-structure. In the system, each server s has a streaming capacity $U(s)$ and a storage capacity $M(s)$ for storing a limited number of C content items. The content items have average size $\overline{s}$ and a play-back rate r_s. The system is serving N clients that belong to one of the G local communities. The average time the clients wait to make a new request after they watched a video is w.

The extension of the model for VoD streaming by including the peers in the streaming process requires defining new parameters in the system that are important for the analysis. The peer-assisted streaming is enabled by taking advantage of the unused streaming and storage capacity. Their size will be crucial for the amount of the traffic that will be served by the peers. In the model, each peer has a storage capacity marked as M_p, which is expressed as the number of entire videos that the peer can store in its STB. Because of the content division, instead of entire contents, the operator assigns only certain strips of the contents for storage in the peers. Another important parameter of the peers is their streaming capacity U_p, which is also expressed as a number of streaming channels. An overview of the parameters defined for the system for peer-assisted VoD streaming is given in Table 4.1.

Table 4.1.: Overview of the system parameters for the system for peer-assisted VoD streaming.

Parameter	Description
S	Number of streaming servers
G	Number of local communities
$G(s)$	Number of local communities served by s
L	Number of levels
$l(s)$	Level of server s
N	Number of clients
$N(s)$	Number of clients directly served by server s
$u(s)$	Streaming utilization of server s
$U(s)$	Streaming capacity of server s
U_p	Streaming capacity of a peer
$M(s)$	Storage capacity of server s
M_p	Storage capacity of a peer
C	Size of video content library
$s(c)$	Size of video content c
$r_s(c)$	Streaming rate of content c
$P(c)$	Request probability of content c
$a(s)$	Rank of the first video stored in server s
$b(s)$	Rank of the last video stored in server s
$\lambda = 1/w$	Arrival rate of requests for contents
d	Average duration of the contents
ρ	Service rate of the system

Another important issue that treats this section is estimating the size of the streaming $U(s)$ and $M(s)$ storage capacity of the servers so that they can comply with the requests of the N clients. This is important because these two capacities are tightly coupled in a way that if one of the capacities is over-dimensioned, it cannot be utilized because the other capacity would be under-dimensioned. The objective is to determine the streaming capacity of the servers given the size of the system and the request pattern of the contents. Because the storage capacity of a server is more easily upgradeable than their streaming capacity and the capacities of the links that interconnect them, first, the streaming capacity of the server will be determined, and then, the storage capacity will be adjusted to store the number of contents that generate the traffic dimensioned with the streaming capacity. It is assumed that all the servers at the edge of the network serve approximately the same number of clients, and therefore, have the same streaming and storage capacities.

The system size is modeled according to the popularity of the content items and according to the position of the servers within the hierarchy. Considering that the distribution algorithm always places the most popular videos in the servers that are closest to the clients, the condition that the streaming capacity $U(s)$ has to fulfill so that all the requests directed to server s can be served is:

$$N(s) \cdot \sum_{c=a(s)}^{b(s)} P(c) r_s(c) \leq U(s) \tag{4.1}$$

where $N(s)$ is the maximum number of clients that server s can simultaneously serve and $P(c)$ is the probability that a request for content c will be generated. For the first level of the tree, $N(s)$ is the number of peers in the local community of server s, and in the upper levels, it is the sum of all clients in the communities that can be directly served by that server, i.e., all clients that are children of the server. The indexes $a(s)$ and $b(s)$ note the ranks of the first and the last content items stored in server s, sorted according to their popularity. The value of the index $a(s)$ is 1 for the ES, while for the rest of the servers it is equal to the value of $b(s)$ of its child server incremented by 1. Assuming that the local communities have the same average size, $N(s)$ can be expressed as:

$$N(s) = \frac{\rho}{1+\rho} \frac{G(s)}{G} N \tag{4.2}$$

where $G(s)$ is the number of local communities served by server s and $\rho = d/w$ is the system's service rate defined as a ratio between the average duration of the videos $d = \overline{s}/r_s$ and the average inter-arrival time of the requests. The ratio $\rho/(1+\rho)$ determines the fraction of the N clients that will be receiving a video for the given service rate ρ (Kleinrock, 1975). Since the request probability of the contents $P(c)$ obeys the ZM distribution, it can be calculated from (3.13). If this expression for the ZM distribution and the expression for the number of simultaneous streams served by the server (4.2) are substituted into (4.1), the condition for the streaming capacity becomes:

$$\sum_{c=a(s)}^{b(s)} (c+q)^{-\alpha} \leq \frac{\rho+1}{\rho} \cdot \frac{G}{G(s)} \cdot \frac{U(s)}{r_s N} \sum_{c=1}^{C} (c+q)^{-\alpha} \tag{4.3}$$

Once the indexes $a(s)$ and $b(s)$ are determined, the optimal storage capacity of the server is determined from the following condition:

$$(b(s) - a(s) + 1)\overline{s} \leq M(s) \tag{4.4}$$

Since $b(s)$ cannot be expressed in a closed form, it is determined by using numerical methods.

4.3. Distribution Schemes

The previous chapter already showed that the placement of the contents in the servers has very important role in the distribution of the traffic among the servers. Since the peers have an active role in the streaming process in the extended model, the distribution of the contents on the peers will also be an important factor for the scale of the benefits obtained with the peers-assisted streaming. This section proposes various mixed schemes for distribution of the contents on the peers obtained by combining simple popularity distributions on different popularity groups of contents.

The schemes are based on the division of the contents in groups according to their popularity. Since most of the requests are destined to a small number of the first most popular videos, the contents are divided into two groups of popularity according to the 80-20 rule (Yu *et al.*, 2006). According to this rule, 80% of the requests are for the first 20% of the most popular contents. Therefore, 20% is the limit that distinguishes the contents as popular or not popular, i.e., the first 20% of the contents of the video library are placed into the group of popular contents and the rest of the videos in the not popular contents. The proposed content distribution schemes include both the popular and not popular content items. Combining these simple distributions enables to take advantage of the contributions of each one of them: the distribution of popular contents makes the network more responsive in highly congested conditions, and the distribution of the not popular contents makes the streaming process locally closer to the clients for all the available contents, and thus, reduces the traffic in the core of the network.

One of the key factors in the definition of the distributions is the portion l of storage capacity on the STB dedicated to the popular contents. If more space for the popular contents is reserved, the uplink capacity of the peers can be better used and the ES will be more alleviated, but it will not contribute to the reduction of the traffic in the core of the network. On the other hand, dedicating more space for the not popular contents will achieve the goal of alleviation of the servers in the core of the network, however, the available uplink capacity of the peers will not be optimally used since the not popular contents generate less traffic. Therefore, the value of the portion of storage space dedicated to the popular contents should be carefully chosen.

Since the main objective of the model is to concentrate the traffic in the periphery of the network, most of the storage capacity will be dedicated to the not popular contents. The reservation of a small portion of the STBs' storage space for the popular contents will provide sufficient alleviation of the ESs in the busy hours and the rest of the storage will enable reduction of the backbone traffic. Although the value of the portion dedicated to the popular contents l largely depends on the system's parameters and the objectives that are to

be achieved, for the sake of the objective to localize the traffic close to the clients, in the simulations, it will take values of 20%. The considered distributions are focused on the not popular contents since they are the most numbered, but the least frequently requested which requires more storage on the peers for alleviating the core servers that serve the requests for these contents. The distributions are based on the contents' popularity and determine the number of strips of each content that will be distributed on the peers. Each distribution consists of two equal distributions applied to the popular and not popular contents. The single distributions are Uniform, Linear and Zipf-Mandelbrot (ZM) distribution. Apart from the mixed distribution, there will also be considered the simple distributions which include distribution of only popular contents (Chen *et al.*, 2010), and only not popular contents. Figure 4.5 shows the content distributions used in the simulations for library of $C = 1500$ content items, where the first 20% of the contents are considered as popular.

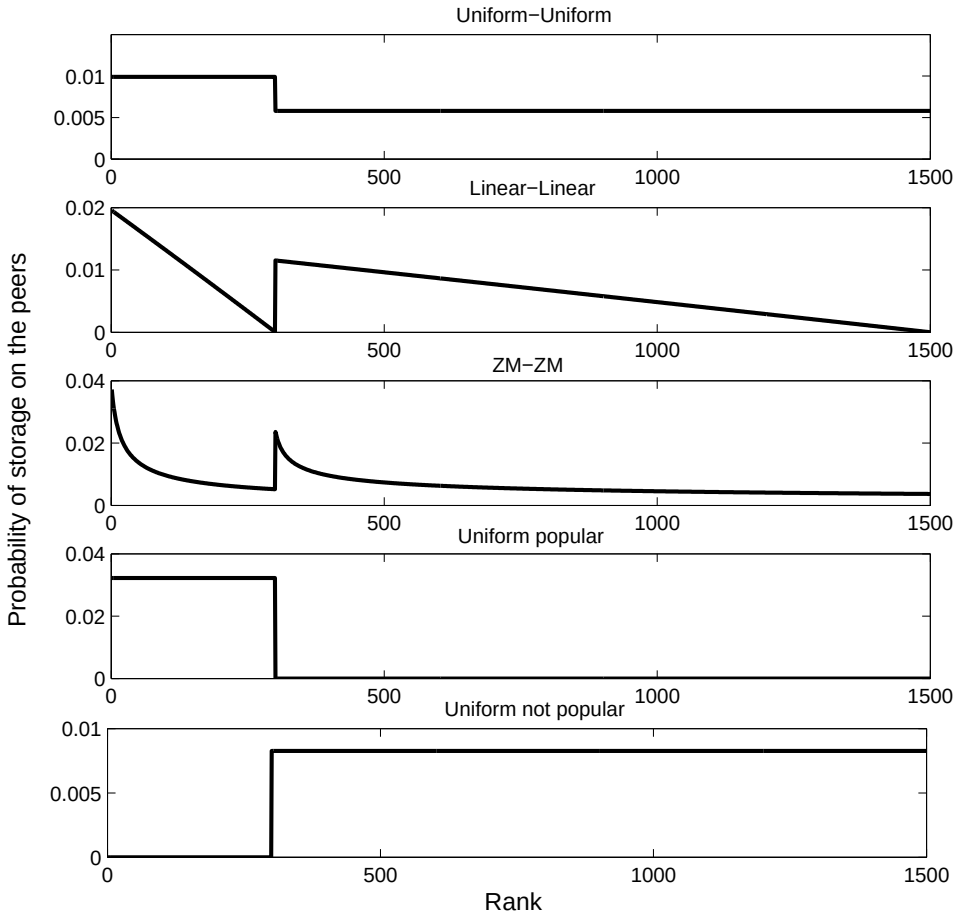


Figure 4.5.: Distribution schemes of the contents on the peers.

4.4. Simulations and results

The simulation environment for testing the behavior of the proposed system for peer-assisted VoD streaming is an extension of the model used in (Chapter 3) with the added new functionality of the peers to participate in the streaming process. In the extended model, the ESs handle the requests of the clients and act as index servers, and the peers serve other clients, just like the servers do. The network considered in the experiments consists of $S = 13$ streaming servers organized in a tree-structure with $L = 3$ levels where each level $l = 1, 2$ and 3, contains 10, 2 and 1 servers, respectively. The streaming capacities of the servers in the same order of levels are $U(s) = 500$, 1000 and 1000 Mbps, accordingly. The links that interconnect the servers have enough capacity to support the maximum streaming load of all the servers. The streaming servers host $C = 1500$ SD videos with play-back rate $r_s = 2$ Mbps and average duration $d = 90$ min. The process of generating requests is a Poisson process with average inter-arrival time of $w = 1/\lambda = 20$ min.

The servers are serving $N = 5000$ clients divided into $R = 10$ communities. Each community is directly served by one ES. The clients possess STBs with capacity to store the entire length of $M_p = 5$ content items. The portion of this storage reserved for strips of the popular contents is $l = 20$ %. The STBs are connected to the network with links that have download capacity higher than the playback rate of the SD video quality. Although the uplink capacity of the peers U_p will have variable value throughout simulations, in the first simulation scenario, it will be $U_p = 200$ kbps (one channel), which is 1/10 of the SD playback rate ($k = 10$).

The popularity of the content items obeys the ZM distribution with shifting coefficient $q = 10$ and $\alpha = 0.8$.

Taking into consideration the defined system's parameters, the storage and the streaming capacities of the servers are dimensioned according to (4.4) and (4.3) in a way that they are optimally used. The contents in the servers and in the peers are previously distributed by the operator in the off-peak hours.

There are several different scenarios considered in the simulations. The first scenario is the reference point for the comparisons and represents the simple case when the streaming process is completely done by the streaming servers (no P2P). The intensity of requests in the system is set to such a value that would keep all the streaming servers constantly overloaded and the same request rate will be used in all the simulation scenarios.

Since the servers are kept in a state of high utilization throughout the simulations, some of the requests directed to the overloaded servers are rejected and the clients are demanded to request the content later within a random period of time within the interval $[1, 3]$ minutes. The percentage of requests that are rejected for immediate service due to overloaded servers

is shown in Figure 4.6. The high denial rate in the scenario with no P2P is a quite expected result since the request rate of the clients is higher than the available resources. Therefore, almost one third of the requests are rejected. The figure shows that the implementation of the peer-assisted streaming with streaming capacity of the peers of one channel ($U_p = 200$ kbps) introduces reduction of the denial rate for any of the considered distributions of the contents on the STBs. Distributing only the popular contents contributes to reducing the number of rejected clients to almost half. This effect is slightly more emphasized in the distribution of the not popular contents. The proposed mixed distributions, however, introduce significantly lower denial rates, among which, the lowest miss rate is obtained for the Uniform-Uniform distribution, followed by the Linear-Linear and ZM-ZM distributions.

The clients which are denied for service because of insufficient streaming resources have to wait certain time and re-request the same video. Figure 4.7 shows the average time that each client has to wait for service in the various simulation scenarios. This value is obtained as a product of the denial rate and the average service delay when a denial happens. As expected from the results from the denial rates, the figure demonstrates that the peer-assisted streaming reduces the time that the rejected clients have to wait for service. However, in the case of distribution of only the popular contents, the service delay does not significantly improve compared to the case of pure server streaming, although this distribution reduces the denial rate to almost half. The other distributions considerably reduce the average service time, especially the Uniform-Uniform distribution which contributes to a remarkable reduction of the average service delay.

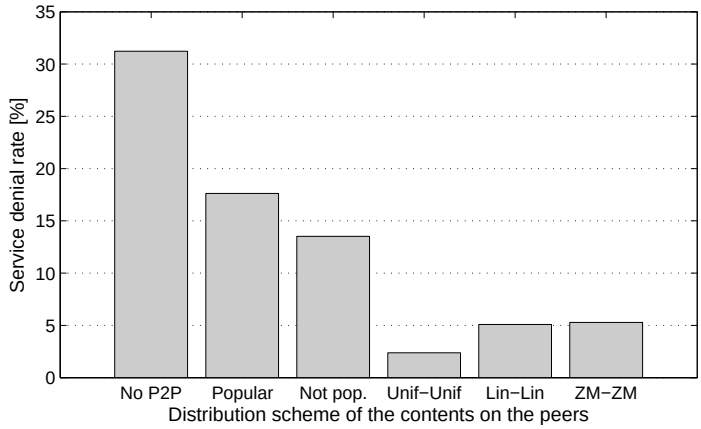


Figure 4.6.: Rate of denial of service for various distribution schemes.

Another measure that is analyzed in order to estimate the contribution of the considered distributions is the transport cost for delivering the video streams from the streaming servers to the clients. This measure is defined in (3.14). The cost is calculated in an identical way as in the previous system for pure-server streaming, i.e., it is a function of the distance from

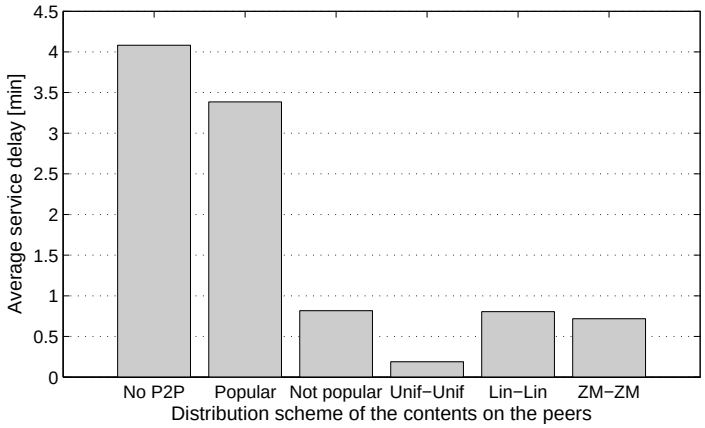


Figure 4.7.: Average service delay for various distribution schemes.

the servers to the clients and their current occupancy. The traffic from the peers is excluded in the cost since the streaming is done over the unused uplink capacity of the clients which is considered to have no transport cost for the operator.

Figure 4.8 shows the average transport cost reduction obtained as a result of the implementation of the various distribution schemes for peer-assisted streaming relative to the case of pure-server streaming. According to the figure, the peer-assisted streaming of the most popular contents and the mixed distribution schemes introduce a modest reduction of the transport cost with almost insignificant difference among each other. On the contrary, the distribution of the not popular contents considerably reduces the traffic in the higher layers of the architecture, and therefore, it reaches the maximum reduction of the transport cost. Although there is reduction of the traffic cost, there could not be expected more significant cuts of the traffic cost since in the simulation scenario the peers have the capacity to stream only $U_p = 200$ kbps, which is only 1/10 of the play-back rate of the videos that they are demanding.

The various distribution schemes also contribute to a different streaming capacity utilization of the STBs. This measure is shown in Figure 4.9. The streaming capacity of the peers is best utilized when they host the most popular contents because it is very probable that any of the strips of these popular contents are requested to be streamed by the peers. The worst utilization is obtained for the distribution of the not popular contents because, although there are many strips of the not popular contents stored in the STBs, there are not always requests for them. The utilization of the mixed schemes lies between the maximum value obtained for the distribution of the popular contents and the minimum value obtained for the distribution of the unpopular contents.

The results show that distributing the contents on the peers according to the mixed Uniform-Uniform schemes would be the optimal choice for best performance under the given system

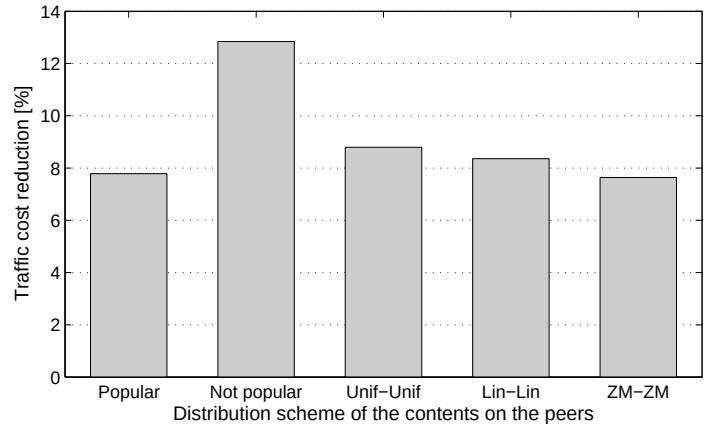


Figure 4.8.: Cost reduction relative to pure server streaming for various distribution schemes.

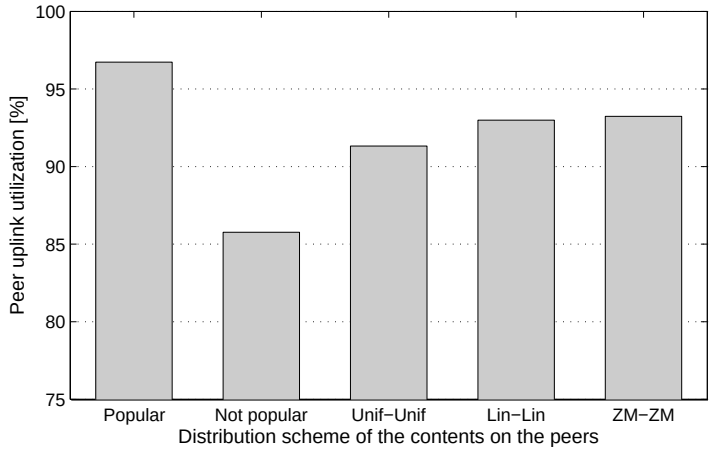


Figure 4.9.: Peer uplink utilization for various distribution schemes.

parameters. Although it does not reach the maximum cost saving and peer utilization of the simple distributions, it is a good compensation for the weak points of both of them. In addition, it significantly improves the number of immediately served clients and the average service delay.

As it was shown in the analysis so far, the distribution of the contents has very important role in the system's performance. The results indicate that even for small streaming capacities of the peers, there are improvements of the performance which are significant compared to the case of pure-server streaming. However, the benefits of the peer-assisted streaming could come to effect if the streaming capacity of the peers is increased. The more uplink capacity a peer has, the more traffic it can serve. The amount of the peer-assisted traffic determines the reduction of the traffic cost in the core of the network which also depends on the distribution of the contents in the peer. The popular contents contribute to the reduction of the traffic

in the edge of the network, while the not popular contents contribute to the reduction of the traffic in the upper levels of the hierarchy. Since one of the goals is to concentrate most of the traffic within the local communities, for sake of the analysis, a value called traffic locality is introduced. This value notes the level of localization of the traffic in the system, and it is defined as percentage of the overall streaming traffic in the system streamed by any member of a local community, i.e., a peer or ES. This value is mostly dependent on the amount of the traffic that the peers are able to serve.

The dependence of the traffic locality on the streaming capacities of the peers for various distributions of the contents on the peers is shown in Figure 4.10. The values of the streaming capacity of each peer vary from $U_p = 0$ kbps, which is equivalent to pure server streaming, to bandwidth $U_p = 2$ Mbps which is the bandwidth for streaming $k = 10$ simultaneous strips or the play rate r_s of the videos. In the figure, it can be seen that when only the popular contents are distributed on the STBs, there is no change of the traffic locality. Moreover, it keeps the same value as the case when no peer-assisted steaming is implemented at all. This result is quite expected because, although the peers serve a significant part of the traffic, they only alleviate the ES servers, and the traffic generated for serving the popular contents by the ES is now passed to the peers. In that case, the traffic of the not popular contents remains the same, and there is no improvement in the locality. On the contrary, the distribution of only the not popular contents considerably improves the traffic locality even for small values of the streaming capacity of the STBs. Initially, as the streaming capacity increases, the traffic locality rapidly grows, and then, from the middle of the range, it slowly, asymptotically moves towards a maximum value close to the entire local coverage of service. A full coverage of local service could be theoretically achieved for infinite number of channels and infinite storage capacity. In the simulation, this value is not achieved for the maximum streaming capacity because, although the peers have available channels for streaming, there are not enough copies of strips of all the requested videos on the peers. The figure also shows that the locality for the mixed distribution schemes is between the previous two simple distributions, with tendency to reach the locality of the not popular contents. The best results are obtained for the Uniform-Uniform scheme and slightly poorer results are obtained for the Linear-Linear and ZM-ZM schemes.

The traffic locality has a big influence on the traffic cost because the more traffic is served locally, the more the cost reduces as a result of the more alleviated servers in the upper levels of the hierarchy. Figure 4.11 shows the average transport cost reduction for peer-assisted streaming for various streaming capacities of the STBs, relative to the case when all the streaming is done by the servers. It can be seen that both the curves of only popular and only not popular contents have a similar behavior. In the lower range of the streaming capacities, the traffic cost reduction increases linearly as the capacity increases, and then, it

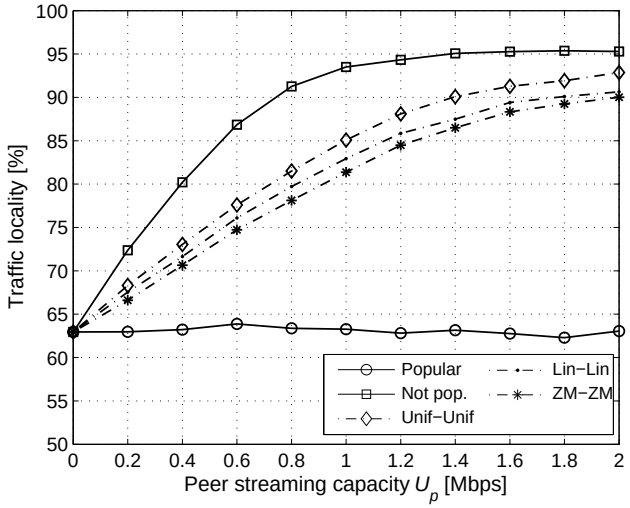


Figure 4.10.: Dependence of the traffic locality on the peers' streaming capacity for various distribution schemes.

asymptotically converges towards a certain limit value. The difference between these two distributions is that distributing the not popular contents contributes to a higher cost reduction. Another common feature of these distributions is that after they reach the maximum reduction, in the middle of the range of streaming capacity, no further increasing of the capacity would cause additional reduction of the cost. The reason for this behavior is that in each of the distributions, there is only reduction of the traffic on the edge of the network or in the higher levels.

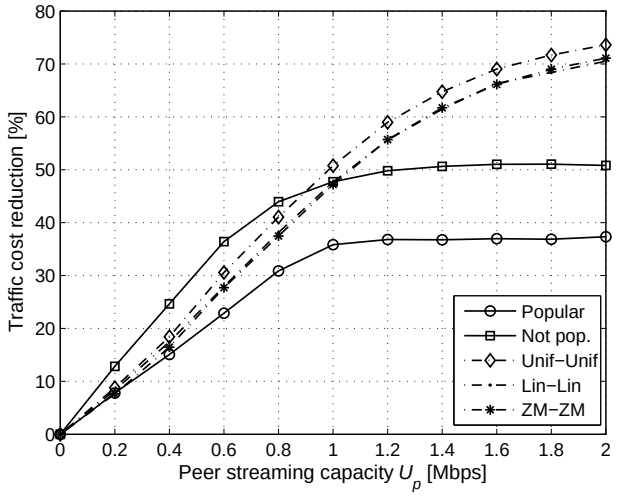


Figure 4.11.: Dependence of the traffic cost relative to pure-server streaming on the peers' streaming capacity for various distribution schemes.

Although for lower values of the capacity range the distribution of the not popular contents

gives the cheapest price for streaming the videos, in the upper part of the range, it is over-performed by the mixed distribution schemes. In these cases, the traffic on each level of the hierarchy reduces because all the contents are placed both in the servers and in the STBs. In the figure, the cost reduction for the mixed distribution schemes and the distribution of the not popular contents intersect in the middle of the range, which is the value of the uplink capacity that is most commonly offered by the operators nowadays with the current ADSL technology. For streaming rates higher than 1 Mbps, the traffic cost will be minimal (the reduction is maximal) if the mixed schemes are used, especially the Uniform-Uniform distribution.

Figure 4.12 shows the dependence of the uplink utilization on the streaming capacity for the various distribution schemes. As it was shown in the previous analysis, for small streaming capacity, the peers are utilized the most when they host only the popular contents and the least when they host the not popular contents. In all the distributions, the peers get less utilized as the streaming capacity grows. The curves for the mixed schemes have smaller inclination and therefore intersect with the curve of the popular content distribution. From the figure, it can be concluded that for best uplink utilization, within most of the range of capacities, the contents should be distributed according to the mixed distribution schemes.

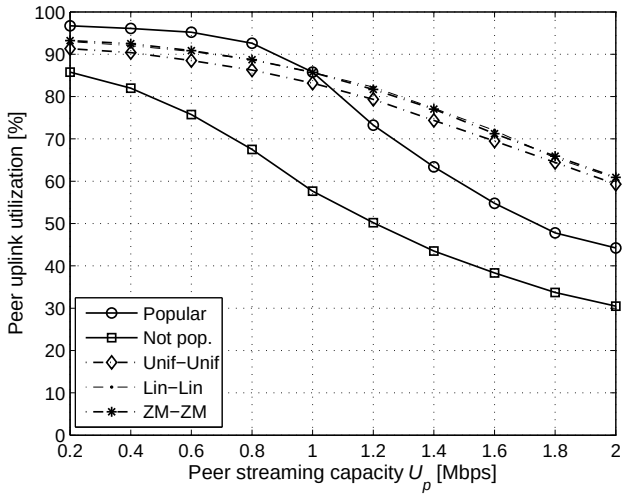


Figure 4.12.: Dependence of the uplink utilization on the peers' streaming capacity for various distribution schemes.

The introduction of the P2P concept in the system also increases its scalability because, as the number of clients increases, there are more storage and streaming resources in the system. However, the increased number of clients also implies more requests which in the model for peer-assisted streaming also requires more resources from the servers. In order to see how scalable the system is, the figures that follow show two cases of dependence of the traffic

cost on the number of peers in the local communities obtained for two different streaming capacities.

In Figure 4.13, the STBs have streaming capacity of $U_p = 600$ kbps. The distribution of the popular contents has no improvement in the transport cost with the growth of the system's size, i.e., the traffic cost relative to the pure-server system will not reduce. Although there are more available resources, the traffic cost will increase proportionally with the size of the local community. Unlike this distribution, the distributions that include the popular contents achieve more cost reduction as the number of the peers in the community grows. Compared to the distribution of the only not popular contents, the mixed distribution schemes give only moderate contribution to the cost reduction.

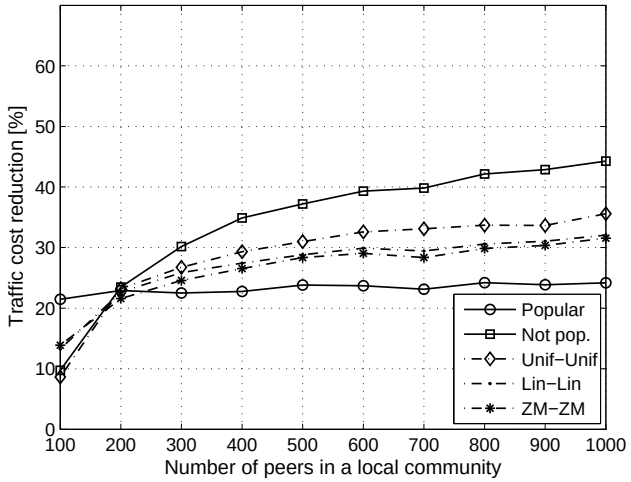


Figure 4.13.: Dependence of the traffic cost reduction on the size of the local community for streaming capacity $U_p = 600$ kbps and various distribution schemes.

The benefits of the peer-assisted streaming in the streaming traffic cost are even more evident when the peers have more streaming capacity, as shown in Figure 4.14, where each peer can stream up to $U_p = 1400$ kbps. Compared to the previous case, the mixed distribution schemes give considerably better results than the distribution of only the not popular contents. The distribution of only the popular contents has minor influence on the traffic cost reduction for small size communities and is almost independent on the number of peers that participate in the streaming process.

The last figures show that, although there are more requests in the system, there are more savings in the cost of streaming because of the increased streaming resources on the peers that alleviate the servers in the core of the networks.

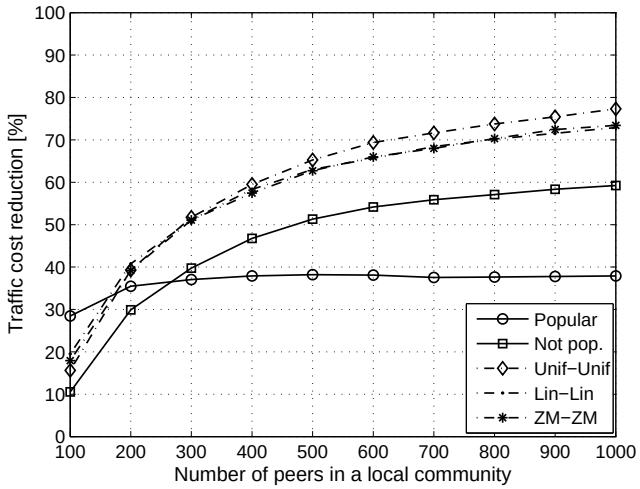


Figure 4.14.: Dependence of traffic cost reduction on the size of the local community for streaming capacity $U_p = 1400$ kbps and for various distribution schemes.

Another proof of this effect is the dependence of the portion of the core traffic relative to the overall server traffic on the community size shown in Figure 4.15 and Figure 4.16. The reason for invariable cost reduction when only the popular contents are distributed, in both of the cases for different streaming capacities is that the portion of the overall traffic served by the servers in the core of the network keeps the same value, and thus, the growth of the number of peers in the community will increase proportionally with the traffic demanded from the core servers. Distributing the not popular contents contributes to reducing the portion of the core traffic when the size of the community grows. This is especially remarkable in the case when the streaming capacity is $U_p = 1400$ kbps when the portion of traffic streamed from the core of the network reaches almost value 0 for large local communities. The mixed distribution schemes cause the same behavior of the system, however, when the peers have small streaming capacity (Figure 4.15), the reduction of the portion of the core traffic with the growth of the community size is almost unnoticeable, compared to the case with higher streaming capacity (Figure 4.16) when the peers achieve to reduce the core traffic to less than 10% of the overall traffic. The figures also show that the mixed Uniform-Uniform distribution outperforms the other proposed distribution schemes.

This observation is important for planning the network by the operators since, when the number of users grows, apart from the traffic cost, there is a price that has to be paid for installation of new servers and new interconnection links. From this point of view, the more feasible solution is distributing the not popular contents or the mixed distributions because the traffic demanded from the core of the network will require new resources in the core with much smaller scale of the growth compared to the scale of resource requirements when only the popular contents are distributed on the peers. Hence, the cost that has to be paid for

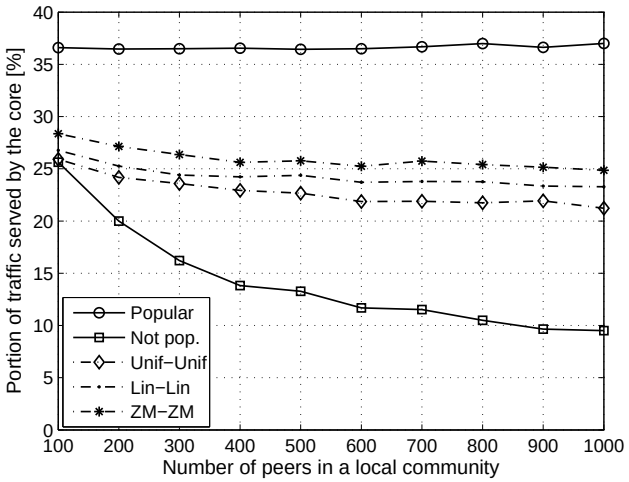


Figure 4.15.: Dependence of the portion of traffic served by the core servers on the size of local community for streaming capacity $U_p = 600$ kbps and various distribution schemes.

increasing the community size will be the cost for installation of new ESs and minor upgrades of the core servers and the inter-connection links. In the case of distribution of only popular contents, the growth of the community size would require much more costly upgrades of the core of the network.

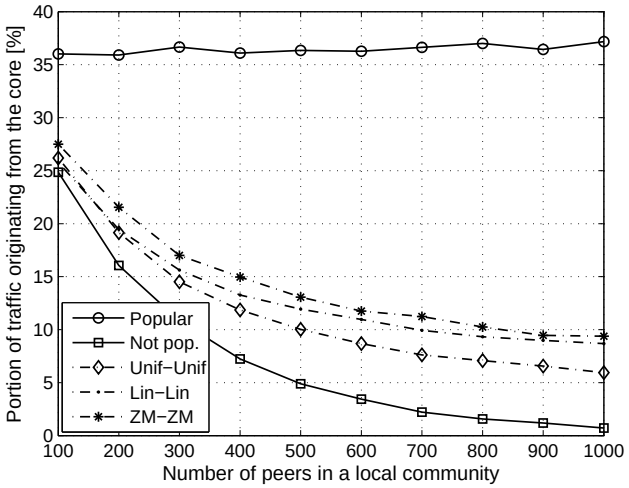


Figure 4.16.: Dependence of the portion of traffic served by the core servers on the size of local community for streaming capacity $U_p = 1400$ kbps and various distribution schemes.

4.5. Conclusions

This chapter proposed a model for peer-assisted VoD streaming in managed networks achieved by using the storage and streaming capacity of the peers. In order to increase the availability of the contents on the peers and to enable streaming of the contents, the system divides the contents into parallel strips which can be independently served both by the peers and the servers. The main focus of the chapter was showing that the distribution of the contents on the peers has a significant importance in the system's performance, provided that the contents are optimally distributed on the servers. Therefore, it proposed mixed popularity-based distributions by dividing the contents into two popularity groups and analyzed the system for each one of them.

The analysis showed that for small streaming capacities of the peers, the peer-assisted streaming improves the quality of service by reducing the rate of denials and the average time that the clients have to wait for service. The mixed distributions showed best performance in the reduction of these parameters although they did not achieve the level of reduction of the traffic cost of the distribution of the not popular contents and the level of utilization of the uplink capacity of the distribution of the popular contents.

The contribution of the peer-assisted streaming to the improvement of the performances is significantly noticeable for higher streaming capacities of the peers. The mixed distribution schemes achieve high locality of the traffic which also results in high traffic cost reduction and better link utilization. The increased locality of the traffic achieves that both the popular and not popular contents are close to the clients, and therefore, the core servers in the network are less loaded. The analysis also showed that the mixed distributions improve the scalability of the network, and hence reduce the installation cost in the network when the size of the local community increases. From all the considered distributions, the mixed Uniform-Uniform distribution of the contents showed the most satisfactory results.

The analysis in this chapter could potentially help the providers to better utilize the resources of the managed network for the VoD service and to estimate the performances for different content distributions and system parameters. However, although accurate, all these results are simulation based and the operator would have to run different simulations for even small changes of the parameters. A solution for this issue is proposed in the chapter that follows, which develops a mathematical model of the peer-assisted system for VoD streaming.

Chapter 5

Stochastic Modeling of Peer-assisted VoD Streaming with Cooperative Peers

The key question asked when implementing the P2P concept in the system for VoD streaming is how the peers will help to reduce the traffic originating from the servers. The answer of this question requires an extensive analysis that depends on many system parameters. Although the previous chapter offered a part of such an analysis, which was simulation based, there is still a large variety of simulation scenarios to be performed and system parameters to be analyzed in order to give a complete picture of the system.

In order to facilitate the analysis, this chapter proposes a precise mathematical tool that models the peer-assisted VoD streaming system including a large variety of parameters. The entire process of serving the clients is presented as a stochastic process which is brought to a closed analytical form using the foundations of the queuing theory (Kleinrock, 1975) and the generating functions (Flajolet & Sedgewick, 2009). The model takes as input the size of the local community, the storage and streaming capacities of the peers, the size of the videos, the intensity of requests, the size of the video library, and what is most important, the distribution of the contents in the peer. The output of the mathematical model is the utilization of the streaming capacity of the peers, which is further used for calculation of the peer-assisted traffic in the system and the traffic served by the servers.

The goal of this chapter is to identify the amount of traffic that is served by the peers and the utilization of the streaming capacity of the peers for various values of the system parameters using the mathematical model. Following these goals, the model generalizes the servers as one server that has the resources to serve all the clients, and therefore, the distribution of the

contents in the servers is of no importance for the analysis. Like in the previous chapter, the key assumption is that the peers are cooperative even when they are not watching a video, i.e., they are always willing to share their resources for streaming strips to the other members of the community.

After the comparison of the results obtained from the model to the results obtained with the simulation, the chapter verifies the validity of the mathematical model and gives a thorough analysis on how the system parameters influence on the contribution of the peer-assisted streaming.

5.1. Mathematical model description

This section presents the steps that lead to the mathematical model for the peer-assisted streaming model defined in the previous section. After defining the set of variables that are the building blocks of the model, the mathematical model is built on the bases of the queuing theory. For that purpose, the system is presented as a network of queues which is initially modified by reducing the number of states aimed for facilitating the computations. To compensate for the simplification, a specific method is used which helps to keep the generality of the basic network of queues despite the reduced number of states of the modified network of queues. At last, a system of linear equations is obtained that is used for calculating the utilization of the streaming capacity of the peers and the portion of the overall traffic streamed by the peers.

5.1.1. The building blocks

The basic building elements of the system are the servers, the peers and the contents they exchange. Therefore, the system is defined as $\mathcal{X} = \{\mathcal{S}, \mathcal{P}, \mathcal{V}\}$, where $\mathcal{S}$ is the set of streaming servers, $\mathcal{P}$ is the set of peers and $\mathcal{V}$ is the video content library. The set of servers in the hierarchy $\mathcal{S} = \{ES\}$ will contain only the ES which will be the representative of all servers that can serve one local community. The reason for using only one server is that the goal of this work is to see how certain parameters of the peers will affect the traffic that directed to the servers as a whole. It is assumed that the representative of the servers has enough resources to store the contents of the video library $\mathcal{V}$ and to serve all the requests that cannot be served by the peers.

The video content library $\mathcal{V} = \{V_1, V_2...V_c\}$ contains c videos with playback rate r. For modeling the users' behavior not to watch the entire video, it is considered that the average duration of the videos d is actually the average time they spend watching the videos. The

items in the library are sorted according to their popularity, beginning with the most popular video. Each content is divided into m strips with equal size. The time necessary to stream one strip is equal to the average watched video duration d, however, the streaming occupies m times less uplink capacity. The rate r/m necessary to stream one strip is defined as a channel. The probability that a video content with rank v will be requested by a peer is marked as $P(v)$ and it is modeled according to the ZM distribution, previously defined in (3.13).

The set $\mathcal{P} = \{P_1, P_2...P_n\}$ represents the set of active peers composing one local community. The number of active peers is denoted as n. This is the number of the peers that are connected to the network and have their STB switched on. The main assumption of the model is that the peers are reliable, i.e., they never turn the STB off, and therefore, the operator can count on their storage and streaming resources even if they are not receiving a video. Each peer has an STB that has capacity to store s strips of video contents. The downlink capacity of the STB is higher than the playback rate of the content items, while the uplink capacity will have variable values throughout the analysis and will be expressed as the number of simultaneous channels k that the peer is capable to stream. In the analysis, it is assumed that all the peers have the same storage and streaming capacity.

Another important parameter of the system is the distribution of the contents on the peers. The single probability that content with rank v will be stored in a peer is denoted as $P_x(v)$. The values of this probability depend on the strategy of the operator to distribute the contents. Although the model will be applicable to arbitrary distributions, for the purpose of the future analysis, a mixed Uniform-Uniform distribution will be used.

Same as in the system studied in Chapter 4, the process of generating requests for videos by the clients is a Poisson process with arrival rate λ, i.e., the inter-arrival time $w = 1/\lambda$ of two consequent requests has an exponential distribution. The serving of the requests by the server or by the peers is also a Poisson process with service rate $\mu = 1/d$. An overview of all the parameters used in this work is listed in Table 5.1.

Table 5.1.: Overview of the system parameters used in the mathematical model for cooperative peer-assisted VoD streaming.

Parameter	Description
n	Number of active peers in a community
n_{rcv}	Number of peers that are receiving a video
n_{idle}	Number of idle peers
m	Number of strips per video
k	Number of streaming channels per peer

Continues...

Parameter	Description
s	Number of strips that can be stored on a peer
l	Percentage of storage capacity reserved for popular contents
c	Size of video content library
μ	Average rate of service of videos by one peer or server
d	Average duration of a video session
λ	Average arrival rate of video requests generated by one client
$\lambda_{p,i,j}$	Average system rate of video requests that arrive at the peers when the system is in state (i, j)
$\mu_{s,i,j}$	Average system rate of service of videos on the server when the system is in state (i, j)
$\mathbf{k}(t) = (i, j)$	State vector of the system at time t, where i is total number of occupied channels on the peers and j is the number of occupied channels on the servers
z	Number of available channels on the peers
$P(v)$	Probability that a request for item with rank v will be generated in the system
$P_{i,j}(t)$	Probability that the system is in state (i, j) at time t
$p_{i,j}$	Stationary probability that the system is in state (i, j)
$P_{p2p}(i)$	Probability that a content item will be served by a peer when the peers have i busy channels
$P^{+}_{p,i,j}(\Delta t)$	Probability that there will be one arrival on the peer within period Δt when the system is in state (i, j)
$P^{-}_{s,i,j}(\Delta t)$	Probability that there will be one departure on the server within period Δt when the system is in state (i, j)
$P_x(v)$	Probability that item with rank v is chosen to be stored on the peers
$P_s(v)$	Probability that a strip of item with rank v is stored in a single peer
$P_s(v, i)$	Probability that a strip of video with rank v is stored in at least one available peer when there are i busy channels
$P_s(v, t, i)$	Probability that a strip of video with rank v is stored in at least one of t available peers when there are i busy channels on the peers
$S(t, z)$	Number of combinations of distributing z available channels on exactly t peers
$w(t, z)$	Probability that z available channels are distributed on exactly t peers
η	Utilization of the streaming capacity of the peers
θ	Portion of traffic served by the peers

5.1.2. The system's behavior as a birth-death process

The basic analysis used for getting the stochastic model of the peer-assisted VoD streaming system is the analysis of birth-death processes (Kleinrock, 1975) which can be used under the condition that the time between two requests and the service time of streaming have exponential distribution, as assumed in the previous section.

In the described model, each peer makes a request for a content item with rate λ. The contents requested by different peers, as well as the time when they are requested, are independent, so the arrival rate of the requests arriving at the ES is a sum of the arrival rates of all the peers. Since the content items are divided into strips, instead of one entire video, the peers are actually requesting m strips. Therefore, a request for one content item is equal to m simultaneous requests, i.e., one peer can be considered as m peers that simultaneously request a different strip of the same content with the same arrival rate. ES gathers all the requests from the peers and redirects them according to the distribution of the contents and the availability of the STBs either to the STBs or to the streaming server. From the ES's perspective, it is not important where the requests originate from or whether there is an available strip on the peers or not. What is important is to find a peer or server that can stream the requested strip. Therefore, if all the requests for strips that arrive in a given time period are gathered and rearranged in groups of m different strips, for the ES there will be no difference whether the rearranged group of strips comes from one peer or from different peers because, looking as a whole, in the given period of time, the total number of requested strips for each video is the same. Hence, instead of observing a peer as a source of m simultaneous requests for the same video, it can be observed as a source of requests for strips of different contents with arrival rate $m\lambda$. Since the requests for strips of one video are dependent on each other, i.e., the request for one strip implies requests for the rest of the strips of the same video, the probability that a request for a strip of video v will arrive at the ES is the same as the probability of request of the entire video. However, the requests for strips of one video are independent on the requests for strips of the other videos, and therefore, the process of requests for strips on one peer can be decomposed as a sum of m independent processes, each with arrival rate λ.

The model for peer-assisted streaming is presented as a closed network of queues with finite population because the number of subscribers for a VoD service is constant and all the clients stay in the system (5.1a). It consists of a waiting queue and video receiving queues. The peers are initially in the waiting queue, where they stay until they make a request. These peers are also called idle peers because, although they are not receiving a stream, they leave

their resources for participating in the streaming to the other peers. The average number of clients that are waiting to make a request for a video is n_{idle}, and hence, the size of the waiting queue in stationary state is mn_{idle}, provided that there are mn peers in the system that request only one strip of a video. When a peer makes a request for a strip, it can be served either by the other peers or by the server. When it starts to receive the requested streams, the peer enters the video receiving queue. The peers that receive a stream are also called receiving peers and their number in stationary state is n_{rcv}. Since all these peers are receiving m strips, the total number of busy video service facilities is mn_{rcv}. If there is at least one peer that has a copy of the requested strip and an available channel, it will be served by the peers. The probability of such an event is P_{p2p}, which is a function of the contents' availability and the current state of the system. As the system has n peers that can simultaneously stream k parallel strips, in the system shown in 5.1a, there are n queues with k service facilities, i.e., $M/M/k$ queues which can potentially serve a request for a strip. If the requested strip is not available on the peers or it is available, but none of the peers that contains the strip has available channels, the request is served by the server with probability $1 - P_{p2p}$. From the assumption that the representative of the servers has capacity to serve all those strips that the peers cannot serve, the server is modeled as a queuing system with an infinite number of parallel service facilities. In reality, the theoretical maximum number of necessary service facilities is mn, which would serve the requests only in the case when all the peers make simultaneous requests and none of the peers can take part in the peer-assisted streaming. Therefore, in the figure, the server is presented with a $M/M/mn$ queue. The service rate of each of the video service facilities is μ. After the end of the streaming, the peers leave the video receiving queues and re-enter the waiting queue. The main task of the mathematical model is to find the average size in stationary state of all these queues, so that the portion of the traffic served by the peers is determined. Although each separate queue of the network can be easily mathematically modeled, the mathematical model of the entire network of queues is complex to be determined because of the large number of queues and the complex representation of the probability function P_{p2p}.

The first step for obtaining the mathematical model of the closed network of queues is defining the state of the system at time t. Let this state is defined with the vector $\mathbf{k}(t) = (k_1(t), k_2(t), ..., k_n(t), k_s(t))$, where $k_i(t) \in \{0, 1, ..., k\}$ for $1 \leq i \leq n$ is the number of busy channels on peer P_i, and $k_s(t) \in \{0, 1, ..., mn\}$ is the number of busy channels on the servers $\mathcal{S}$ at time t. The values of the state vector $\mathbf{k}(t)$ at any time must satisfy the following condition:

$$\sum_{i=1}^{n} k_i(t) + k_s(t) \leq mn \tag{5.1}$$

If the process of serving the requests is considered as a Markov population process, the

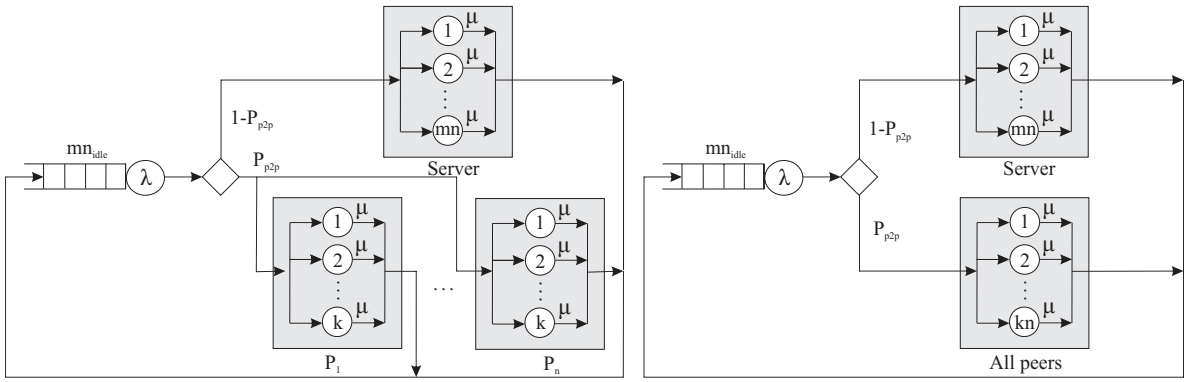


Figure 5.1.: Representation of the model as a network of queues.

transition from state $\mathbf{k}(t)$ to state $\mathbf{k}(t + \Delta t)$ within an infinitesimal period of time Δt will allow one and only one of the following cases: a birth, which is one request for a strip, a death, which is equivalent to the end of streaming of one strip, or nothing at all, which means that there is nor birth neither death during the infinitesimal interval of time Δt.

The calculation of all the possible combinations of different states of the system with this definition, given the set of values for the states of the peers and the server, the condition (5.1) and the number of active clients n, which is higher than a few hundreds, would result with an enormous set of states that is difficult to be handled. For simplification of the model, the number of states is reduced by considering all n peers as one representative with streaming capacity of kn channels (Figure 5.1b). One should note that with this modification it seems that the constraint that a single STB can have at most k channels is lost, however, later in the text, this constraint will be kept in the system by introducing a specific method for calculating the probability of redirecting a strip to be served by the peers P_{p2p}. Thus, although the network of queues is simplified, the generality of the network shown in Figure 5.1a is not lost. The reduction is made possible due to the fact that all the peers in the system have the same storage and streaming capacity, and the contents are independently distributed on the peers. With the reduction of the peers to one representative peer with their cumulative streaming capacity, the state vector reduces to $\mathbf{k}(t) = (k_p(t), k_s(t))$, where $k_p(t) \in \{0, 1, ..., kn\}$ is the number of busy channels on the representative peer and $k_s(t) \in \{0, 1, ..., mn\}$ is the number of busy channels on the servers at time t. The values of the state of the system must satisfy the following condition:

$$k_p(t) + k_s(t) \leq mn \tag{5.2}$$

The main objective of the mathematical model is to determine the probability of each possible

state $\mathbf{k}(t)$ in stationary state of the system, which will be later used for calculating the expected value of the number of busy channels of the peers, and hence, the utilization of the uplink capacity of the peers.

Let $P_{i,j}(t) = P(\mathbf{k}(t) = (i,j))$ be the probability that the system is in state $\mathbf{k}(t) = (i,j)$, i.e., that there are i busy channels on the peers and j busy channels on the server at a given moment of time t. The values of these probabilities when $t \to \infty$ can be determined by applying the condition of equilibrium in stationary state to the probability flow equations of the system, i.e., the change of probability of a certain state during an infinitesimal time period Δt. These equations can be obtained from the very definition of the Markov population process, which states that in an infinitesimal small time there can be only one birth, one death or none of the two. If this definition is applied in the network of queues with two service facilities (Figure 5.1b), one could easily get to the conclusion that for a given state of the system $\mathbf{k}(t+\Delta t) = (i,j)$ at time $t+\Delta t$ only one of the following events could have happened during the interval Δt:

1. the system had been in state $\mathbf{k}(t) = (i-1,j)$ and a request for a strip arrived at the peer,
2. the system had been in state $\mathbf{k}(t) = (i+1,j)$ and a streaming of a strip has been completed on the peer,
3. the system had been in state $\mathbf{k}(t) = (i,j-1)$ and a request for a strip arrived at the server,
4. the system had been in state $\mathbf{k}(t) = (i,j+1)$ and a streaming of a strip has been completed on the server,
5. the system had been in state $\mathbf{k}(t) = (i,j)$ and neither request arrived nor a stream has been completed on any of the service facilities.

Taking into consideration all these possible events, the probability $P_{i,j}(t+\Delta t)$ that the system will be in state $\mathbf{k}(t+\Delta t) = (i,j)$ can be expressed as:

$$\begin{aligned} P_{i,j}(t+\Delta t) &= P_{i-1,j}(t)P^{+}_{p,i-1,j}(\Delta t) + P_{i+1,j}(t)P^{-}_{p,i+1,j}(\Delta t) + \\ &+ P_{i,j-1}(t)P^{+}_{s,i,j-1}(\Delta t) + P_{i,j+1}(t)P^{-}_{s,i,j+1}(\Delta t) + \\ &+ P_{i,j}(t)\left(1 - P^{+}_{p,i,j}(\Delta t)\right)\left(1 - P^{+}_{s,i,j}(\Delta t)\right) \cdot \\ &\cdot \left(1 - P^{-}_{p,i,j}(\Delta t)\right)\left(1 - P^{-}_{s,i,j}(\Delta t)\right) \end{aligned} \tag{5.3}$$

In this expression, the probabilities of any of the above defined events during the interval Δt are marked in a way that the superscript $+/-$ denotes whether an arrival or an end of service

has happened, the first subscript denotes the service facility where the event has happened, with notation p referring to a peer and s to a server, and the remaining two subscripts denote the previous state of the system, before the event happened. The first two lines of the expression refer to the conditions (1) to (4), accordingly, while the last two lines refer to the condition (5) representing the joint probability that nor arrival of request, neither end of streaming will happen in the system when it is in state $\mathbf{k}(t) = (i, j)$. From the properties of the Markov chain, the arrivals and the departures are independent, and therefore, the joint probability can be calculated as the product of the single probability of each of the four possible events.

From the definition of a Poisson process, the probability that there will be one arrival within Δt, given the arrival rate λ is:

$$P(k = 1, \Delta t) = \lambda \Delta t e^{-\lambda \Delta t} = \lambda \Delta t (1 - \lambda \Delta t + \dots) \tag{5.4}$$

For very small values of Δt:

$$P(k = 1, \Delta t) = \lambda \Delta t + o(\Delta t) \tag{5.5}$$

where $o(\Delta t)$ is the infinitesimally small probability of multiple arrivals.

Applying this equation to the probability of arrival on the peers, $P^{+}_{p,i,j}(\Delta t)$ becomes $\lambda_{p,i,j}\Delta t + o(\Delta t)$. The same equation can be used for the departure event, with the difference that instead of the arrival rate λ, the departure rate μ is used. Thus, $P^{-}_{p,i,j}(\Delta t)$ becomes $\mu_{p,i,j}\Delta t + o(\Delta t)$. The event probabilities for the server are defined in the same way as the event probabilities for the peers. If these equations for representing the event probabilities are substituted in (5.3), the whole expression is divided by Δt, and then it is rearranged, the following expression is obtained:

$$\begin{aligned} \frac{P_{i,j}(t + \Delta t) - P_{i,j}(t)}{\Delta t} &= \lambda_{p,i-1,j} P_{i-1,j}(t) + \mu_{p,i+1,j} P_{i+1,j}(t) + \\ &+ \lambda_{s,i,j-1} P_{i,j-1}(t) + \mu_{s,i,j+1} P_{i,j+1}(t) - \\ &- (\lambda_{p,i,j} + \lambda_{s,i,j} + \mu_{p,i,j} + \mu_{s,i,j}) P_{i,j}(t) + \frac{o(\Delta t)}{\Delta t} \end{aligned} \tag{5.6}$$

For $\Delta t \to 0$, the left side of (5.6) becomes the first derivation of the dynamics of the probability $P_{i,j}(t)$ and $o(\Delta t)/\Delta t$ becomes 0. In this case, the set of equations (5.6) will be a system of differential equations which, if solved, will give the dynamics of the probability of each state $P_{i,j}(t)$ in the time. The equations of the dynamics of the probabilities will be used

for determining their values in stationary state of the system. This state is reached after sufficiently long time t when there are no changes of the probabilities that the system will be in a certain state. This leads to the condition for equilibrium which is:

$$\frac{dP_{i,j}(t)}{dt} = 0 \tag{5.7}$$

Let the stationary value of the probability $P_{i,j}(t)$ that the system will be in state $\mathbf{k}(t) = (i, j)$ when equilibrium is achieved be defined as:

$$p_{i,j} = \lim_{t \to \infty} P_{i,j}(t)$$

If these values are substituted in (5.6) and the equilibrium condition (5.7) is applied, then the system of probability flow equations becomes:

$$\begin{aligned} 0 &= \lambda_{p,i-1,j} p_{i-1,j} + \mu_{p,i+1,j} p_{i+1,j} + \\ &+ \lambda_{s,i,j-1} p_{i,j-1} + \mu_{s,i,j+1} p_{i,j+1} - \\ &- (\lambda_{p,i,j} + \lambda_{s,i,j} + \mu_{p,i,j} + \mu_{s,i,j}) p_{i,j} \end{aligned} \tag{5.8}$$

Taking into account the values that can be taken by the indexes i and j (the number of busy channels of the peers and the server) and the condition (5.2), (5.8) reduces to a system of linear equations.

In order to get the solutions of the system of linear equations, its coefficients have to be determined. These coefficients are the arrival and departure rates of the representative peer and the server in every possible state. The departure rates are simpler to be determined because in the case of parallel service facilities, the departure rate of the system as a whole is the departure rate of a single service facility, multiplied by the number of busy facilities. When the system is in state $\mathbf{k} = (i, j)$, the departure rate of the peer is independent on the number of strips that the server is serving, and the other way around, the departure rate of the server does not depend on the occupied streaming channels on the peer. Therefore, they will be determined as:

$$\mu_{p,i,j} = i\mu \tag{5.9}$$

$$\mu_{s,i,j} = j\mu \tag{5.10}$$

where μ is the departure rate of a single service facility which presents one streaming channel.

Because of the closed nature of the system, the arrival rates of the requests for streaming will depend on the number of the remaining clients that are not being served. This dependence will be sufficient for determining the arrival rates of requests on the ES, but not for determining the particular arrival rates on the peers and the server separately. The key reason is that not all the requested strips are available on the peers, and not always the peers have available channels to stream the strips they are hosting. Therefore, the arrival rate of the requests that are directed to be served by the peers will be:

$$\lambda_{p,i,j} = P_{p2p}(i)(mn - i - j)\lambda \tag{5.11}$$

where $P_{p2p}(i)$ is the joint probability that the requested strip is stored somewhere in the peers when they have i busy channels, and at least one of that peers has available channels to stream the strip when the system is in state $\mathbf{k} = (i, j)$. It is obvious that this probability depends on the state of the representative peer since although a strip might be stored in the peer, it cannot be served if there are no channels available.

The representative of the servers has all the strips in the system and always has sufficient number of channels to stream all the requested strips. Therefore, it will serve any request that cannot be accepted by the peer. Its arrival rate will be determined as:

$$\lambda_{s,i,j} = (1 - P_{p2p}(i))(mn - i - j)\lambda \tag{5.12}$$

It is interesting to note that when $i = kn$, $P_{p2p}(i) = 0$ because no matter whether a strip is stored in the peers or not, it will be redirected to the server because of lack of streaming channels on the peers.

Eventually, if (5.9)- (5.12) are substituted into (5.8), the following expression is obtained:

$$\begin{aligned} 0 &= P_{p2p}(i-1)(mn - i - j + 1)\lambda p_{i-1,j} + (i+1)\mu p_{i+1,j} \\ &+ (1 - P_{p2p}(i))\,(mn - i - j + 1)\lambda p_{i,j-1} + (j+1)\mu p_{i,j+1} - \\ &- ((mn - i - j)\lambda + (i + j)\mu)\, p_{i,j} \end{aligned} \tag{5.13}$$

To give a better picture of the states of the system in equilibrium, Figure 5.2 shows the state diagram obtained from the system of linear equations (5.13).

In Figure 5.2, each circle represents the probability of one possible state of the system, and the arrows represent the flow of probability that the system will go from one state to another.

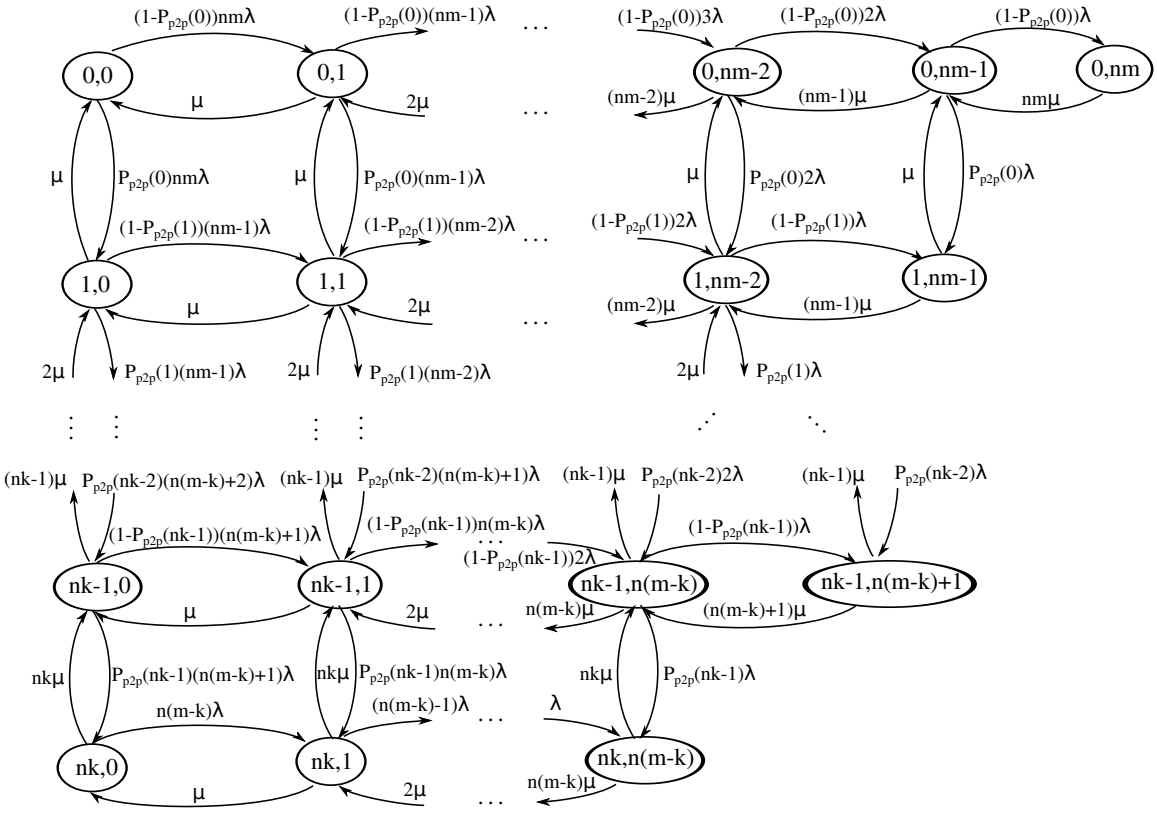


Figure 5.2.: Diagram of flow of state probabilities.

For the sake of clarity, the self-loops of the probabilities represented by the coefficient of the last member of (5.13) are intentionally omitted. The width of the diagram is always $mn+1$, while its height depends on the number of streaming channels k of the peers. These dimensions cause that the size of the state diagram is $(kn+1)((m-k/2)n+1)$ (Appendix B). Since the size of the system rapidly grows with the increment of the size of the community n and the streaming capacity of the peers k, in Appendix A, a sampling method for reduction of the system's size is proposed.

5.1.3. Re-establishing the generality

The only missing part for solving the system of linear equations (5.13) is the probability $P_{p2p}(i)$ that a strip will be served by a peer as a function of the distribution of the content items on the peers. In the model, the contents are stored according to a certain distribution which determines the frequency of appearance of the videos according to their popularity. By normalizing the frequency of appearance of the videos so that the sum of all values equals 1, the probability mass function is obtained. This function defines the probability $P_x(v)$ that a video with rank v is chosen to be stored in a peer. The strips of a single content with rank v are distributed in a way that first the content is selected with a probability $P_x(v)$, and

then, one of its m strips is randomly chosen and distributed to any peer that has sufficient space. The probability of choosing any of the m strips is uniformly distributed since each strip has equal importance, and hence, same probability to be chosen for storage. Therefore, the probability that one strip is selected to be stored in one available location of a peer will be $P_x(v)/m$. Since each peer has capacity to store s strips (there are s locations), and a single strip cannot be stored in more than one location, the probability that a strip of the video item with rank v is stored in exactly one specific location of a peer is determined as the product of probability $P_x(v)/m$ that it is chosen to be stored in one location and the probability $(1 - P_x(v)/m)^{s-1}$ that it is not stored in any of the remaining $s - 1$ locations. As the strip is equally probable to be stored in any of the s locations, the total probability $P_s(v)$ that a strip of video with rank v is stored in a single peer with capacity s is:

$$P_s(v) = s\frac{P_x(v)}{m}\left(1 - \frac{P_x(v)}{m}\right)^{s-1} \tag{5.14}$$

The probability defined in (5.14) will be used further on for calculating the probability that a strip will be stored in the peers that have available channels, which depends on the state of the peers. When the system is in state $\mathbf{k} = (i, j)$, there are i busy channels on the peers and $kn - i$ available channels for serving the strip. It is particularly important to determine the number of peers that can form a sum of available channels that equals $kn - i$, i.e., how many peers can have that number of available channels. For that purpose, the range R of number of peers that can form the given number of available channels is defined. Given the constraint that the peers have capacity of k channels, the minimum number of peers that can have $kn - i$ available channels is $\lceil (kn - i)/k \rceil$ in a way that almost all the peers will have all the k channels available. The upper limit of this range is $\min(n, kn - i)$ because $kn - i$ channels can be distributed on a maximum of n peers in the case when $kn - i > n$ or $kn - i$ peers, otherwise. Taking into consideration these limits, the range R can be presented as:

$$R = \left[\left\lceil \frac{kn - i}{k} \right\rceil, \min(n, kn - i)\right] \tag{5.15}$$

For each value t of this range, there is a different probability of finding available peer that contains a required strip of a video. If $kn - i$ available channels are distributed on exactly t peers, the probability $P_s(v, i, t)$ that a requested strip of video v is stored in at least one of these t peers when the peers have i busy channels is obtained by subtracting from one the probability that the strip cannot be found on any of these available peers:

$$P_s(v, i, t) = 1 - (1 - P_s(v))^t, \quad t \in R \tag{5.16}$$

It must be noted that although i is not explicitly present in (5.16), it is implicitly present through the dependence of the values of the range R on the number of busy channels on the peer i. For every value that t can take from the range R, a different probability of finding the strip on any of that given number of peers is obtained. Although any of these situations is possible, they occur with different probability, and therefore, the probability of finding at least one available peer that contains a strip of item with rank v when there are i busy channels on the peers will be the weighted sum:

$$P_s(v,i) = \sum_{t \in R} w(t, kn-i) P_s(v,i,t) \tag{5.17}$$

where $w(t, kn-i)$ is the probability that of all the possibilities of distributing $kn-i$ available channels, they will be distributed on exactly t peers. For the sake of clarity, the number of available channels on the peers will be defined as $z = kn-i$, and the probability of every value of the range R will be noted as $w(t,z)$. With this approach, it is justified the simplification made earlier that consisted in substituting all the peers with one peer that has their streaming capacity, without losing the constraint that each peer can stream up to k channels.

Each value of the range R is not equally probable, so each one of them have to be determined. For this purpose, all the possible combinations of distributing i available channels in these peers have to be found, under the constraint that each peer can have available channels in the range $[1,k]$. The value 0 of this range is intentionally omitted because if a peer has 0 channels, it could not be considered as available peer.

Let the k-tuple $(q_1, q_2, ..., q_k)$ be defined as weights such that:

$$q_1 \cdot 1 + q_2 \cdot 2 + ... + q_k \cdot k = z \tag{5.18}$$

and let Q be the set of all k-tuples that fulfill the condition (5.18). Each k-tuples represents one combination of distributing z available channels on t peers. Then, the number of all possible combinations of channels distributed on t peers such that their sum equals z is:

$$S(t,z) = \sum_{(q_1,q_2,...,q_k) \in Q} \frac{t!}{q_1! q_2! ... q_k!} \tag{5.19}$$

The problem of determining $S(t,z)$ by finding the whole set Q is that it is time-consuming process, especially for higher values of t. A neater and faster solution for calculating $S(t,z)$ can be obtained by using the generating functions (Flajolet & Sedgewick, 2009) and the properties of the product of more ordinary generating functions. Starting from the definition of an ordinary generating function:

$$f(x) = \sum_{m=0}^{\infty} a_m x^m \tag{5.20}$$

an ordinary generating function specific to the considered system for peer-assisted streaming of VoD can be defined as:

$$g(x) = \sum_{m=1}^{k} x^m = x + x^2 + ... + x^k \tag{5.21}$$

In terms of applicability of the generating functions in the model, the above function means that one peer can have only one of the values defined in the interval $[1, k]$ as a value for available channels. Therefore, (5.21) is obtained by setting $a_0 = 0$ and value 1 for the rest of the coefficients in (5.20). The power of the product of the generating functions comes from the fact that if an arbitrary number of functions is multiplied, each of which defines a range of values for available channels on a peer, the coefficients of each member of the obtained generating function give the number of combinations that can be made with the defined range of values with sum equal to the power of the member they are multiplying. In the specific case of the considered model, the product will be obtained by multiplication of the ordinary generating function defined in (5.21) as many times as there are active peers that are to be considered for distribution of the z available channels:

$$G(t, k) = g(x)^t = (x + x^2 + ... + x^k)^t \tag{5.22}$$

According to the product property of the generating functions, $S(t, z)$ will be obtained as the coefficient that multiplies the member x^z in the polynomial $G(t, k)$:

$$S(t, z) = \text{Coefficient}\,(G(t, k), z) \tag{5.23}$$

Thus, the weight $w(t, z)$ will be obtained as a fraction of the sum of all the combinations of distributions of z available channels on any of the peers of the range R:

$$w(t, z) = \frac{S(t, z)}{\sum_{x \in R} S(x, z)} \tag{5.24}$$

Although the use of the generating functions considerably accelerates the calculation of the weight $w(t, z)$, it proves that for higher values of the community size n and the peers' streaming capacity k, the calculation of $w(t, z)$ is still a time-consuming process. This issue is addressed in Appendix A where, based on the values obtained for smaller values of n and k

with the method proposed in this section, a numerical extrapolation method is proposed for calculating the weight $w(t, z)$ for any values of n and k.

Up to this point, it is obtained only the probability $P_s(v, i)$ that a strip of the content item with rank v will be directed to be served by the peers when they have i busy channels. In order to get the general probability that any strip of the video content library $\mathcal{V}$ will be directed to the peers, the probability that a given video with index v will be requested by a client is considered. This probability is the popularity of the item $P(v)$, previously defined in (3.13). The popularity of one strip is the same as the popularity of the entire video because with each request of a video there are m simultaneous requests for its m composing strips. Recalling that the size of the video library is c, the probability that a strip will be redirected to be served by the peers when the system is in state $\mathbf{k} = (i, j)$ can be calculated as:

$$P_{p2p}(i) = \sum_{v=1}^{c} P(v) P_s(v, i) \tag{5.25}$$

Finally, the coefficients of the system of linear equations (5.13) can be determined after substituting (5.14) into (5.16), (5.16) and (5.24) into (5.17), and (5.17) into (5.25).

5.1.4. Final results

After calculation of the probability $P_{p2p}(i)$, all the coefficients of the system of linear equations (5.13) are known, and the stationary values of the probabilities of the states can be easily determined by solving the system. To get to the solution, the system of linear equations is presented in a matrix form:

$$A\mathbf{p} = \mathbf{0} \tag{5.26}$$

where $\mathbf{p} = [p_{0,0}\ p_{0,1}\ \ldots p_{kn,(m-k)n}]^T$ is the vector of the unknown stationary probabilities of the states of the system with size $((m-k/2)n+1)(kn+1)$, and $A = [\mathbf{a}_{0,0}^T\ \mathbf{a}_{0,1}^T\ \ldots\ \mathbf{a}_{kn,(m-k)n}^T]^T$ is the coefficient matrix, where $\mathbf{a}_{i,j}$ is a row vector with the same size as $\mathbf{p}$ such that contains the weights of the arcs on the state diagram in Figure 5.2 that go out of the state $p_{i,j}$ on the positions corresponding to the positions of the states in the vector $\mathbf{p}$ that they are directed to.

The simplest solution of the given system $\mathbf{p} = \mathbf{0}$ is not of interest because the probabilities have to fulfill the condition:

$$\sum_{i+j \leq mn} p_{i,j} = 1 \tag{5.27}$$

From the values of the matrix A, it can be proven that $\text{rank}(A) = \text{size}(A) - 1$, which implies linear dependence of the equations and, i.e., an infinite number of solutions. In order to get a unique solution, the system is modified by using the condition (5.27). The modification consists in replacing the first row of the coefficient matrix A with a row with the same size and values 1 on every position. The result of this operation is a new coefficient matrix $B = [\mathbf{1}\ \mathbf{a}_{0,1}^T\ \ldots\ \mathbf{a}_{kn,(m-k)n}^T]^T$. This change also implies the replacement of the vector on the right side of (5.26) by a new vector $\mathbf{b} = [1\ 0\ \ldots\ 0]^T$ which has 0 values on every position, except on the first position which has value 1. After the modification, $\text{rank}(A) = \text{size}(A)$, and therefore, the unique solution of the system can be calculated as:

$$\mathbf{p} = B^{-1}\mathbf{b} \tag{5.28}$$

From the values of the vector $\mathbf{p}$, the probability that there are exactly i busy channels in the system will be further calculated as:

$$P_i = \sum_{j=0}^{mn-i} p_{i,j}$$

Using this probability, the expected number of busy channels in the system will be:

$$\overline{K} = E[K] = \sum_{i=1}^{kn} iP_i \tag{5.29}$$

Eventually, the utilization of the streaming capacity of the peers (peer uplink utilization) η will be calculated as the fraction of the expected number of busy channels from the overall number of streaming channels in the system:

$$\eta = \frac{E[K]}{kn} \tag{5.30}$$

Another measure that is important for the analysis is the percentage of overall traffic in the system that is streamed by the peers θ. This measure can be obtained from the value of the overall number of busy channels in the system, both on the peers and the servers mn_{rcv}. In the model, any request for a strip is served by any of the service facilities, i.e., there is always available service facility for serving the request. If the strip is available on a peer with available channels, it is served by the peers, and if this condition is not satisfied, the strip is streamed by the servers. Therefore, looking as a whole, the system can be modeled as a finite customer queue with infinite parallel serving facilities $M/M/\infty$. In that case, the average number of busy channels in steady state (Kleinrock, 1975) is:

$$\overline{K}_{all} = mn_{rcv} = \frac{mn\rho}{1+\rho}$$

where $\rho = \lambda/\mu$.

Since only $\overline{K}$ of those channels are occupied for streaming the strips by the peers, the portion of traffic served by the peers (peer-assisted traffic) will be:

$$\theta = \frac{\overline{K}}{\overline{K}_{all}} = \eta \frac{k}{m}\left(1 + \frac{1}{\rho}\right) \tag{5.31}$$

5.2. Model verification and analysis

In order to validate the correctness of the proposed mathematical model, the results obtained from the model for given input parameters are compared to the results obtained by simulations for the same parameters. For the simulation purposes, the same simulation environment developed in the previous chapter was used.

The system contains one server which is a representative of all the servers with sufficient resources to serve any request. This server has a role of an ES and an index server for a community of $n = 200$ peers. In the simulations, only one community of users is considered because each local community is served by a designated server and each community has the same distribution of contents. Although the results for one community are presented, the simulations are made for a system with 10 local communities which simultaneously make requests to its own designated ES, and then the mean values of the measurements are presented. All the peers in the community are active during the whole period of the simulations. Since the system is defined for privately managed network, the operator can reserve resources on the peers both for streaming and storing the strips. There is a library of $c = 1500$ SD videos with average watched duration of $d = 90$ min. The playback rate of the videos is $r = 2$ Mbps. The streaming capacity has a variable value and is expressed as a number of channels k for streaming a single strip of a video. Each strip is streamed with a rate of 200 kbps which imposes a division of the contents into $m = 10$ strips. The storage capacity is also a variable parameter in the system and is expressed as the number of strips s that can be stored in an STB. The peers in the system are homogeneous, i.e., all the peers have the same storage and streaming capacity.

The arrivals of requests are modeled as a Poisson process with inter-arrival time of $w = 20$ min which is the average time a client waits to request a video after previously watching a video. Although the average service duration and the average inter-arrival time have specific values in the simulations, what is important in the performance of the system is the ratio

$\rho = d/w = \lambda/\mu$.

The popularity of the contents obeys the laws of the ZM distribution, defined in 3.13, with skew factor $\alpha = 0.8$ and shifting constant $q = 10$.

Since most of the requests are for a small number of the first most popular videos, the contents are divided into two groups of popularity according to the 80-20 rule (Yu *et al.*, 2006). According to this rule, 80% of the requests are for the first 20% of the most popular contents, and therefore, the first 20% of the contents are placed into the group of popular contents and the rest of the videos in the not popular contents. In order to make a distinction between these groups of contents, a parameter l is introduced, defined as the portion of the storage space reserved for placement of strips of popular videos. The videos are previously distributed on the peers in the off-peak hours when the streaming activity of the clients is very low. Although the mathematical model allows any type of distribution of the contents, initially, only uniform distribution for both the popular and not popular videos will be considered. The difference between these two types of distributions is that it is dedicated $l = 30\%$ of the peer storage capacity for storing strips of popular contents and the rest of the storage for the not popular contents. The portion of the storage dedicated to the popular videos is chosen to this specific value since the analysis will show that it yields best performance of the system. Since the mathematical model is defined for a stationary state of the system, the simulated time is long enough so that a stable value of the average channel utilization is obtained. The value that will be finally plotted is the mean value of the average utilization within the time interval when a stable value is reached.

The simulations are based on the network of queues described in Figure 5.1a, because in the simulation model, every peer is a separate service facility with k parallel servers, and the ES is a service facility that has enough resources to serve the requests that cannot be served by the peers. The results from the mathematical model, however, are based on Figure 5.1b because of the modification made in order to reduce the number of states of the network of queues and reduce the complexity of the model. Despite this modification, the generality of the model is kept, respecting the streaming capacity constraints of the clients by the way the probability P_{p2p} is calculated. In the simulations, the cumulative streaming traffic of the peers and the streaming traffic originated from the servers is measured. These results are then compared to the results obtained by the mathematical model.

Figure 5.3 shows the utilization of the peer streaming capacity in the system η, defined in (5.30), for two different values of the storage capacity of the peers s. In the first case, the storage capacity of each peer is equal to the capacity necessary for storing $s = 50$ strips, and in the second case, this value is $s = 100$ strips. It can be seen that as the streaming capacity of the peers grows, its average utilization falls. The reason for the reducing efficiency

is that although the peers have more capacity to assist in the streaming process, they cannot respond to the requests since they do not have sufficient number of different strips to satisfy the demand.

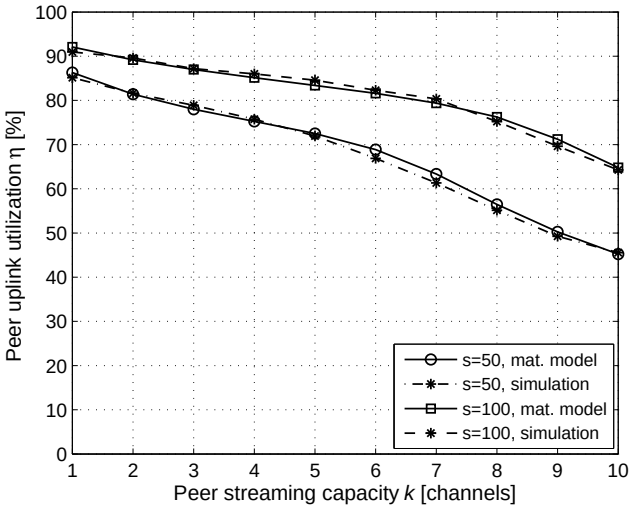


Figure 5.3.: Comparison between the simulation and the mathematical model results, and influence of the peers streaming capacity k for two storage capacities ($s = 50$ and $s = 100$ strips) on the uplink utilization η.

This can be particularly noticed in Figure 5.4, which shows the portion of the total traffic in the system served by the peers θ, defined in (5.31), with the same conditions as in the previous case. It can be seen that, as the streaming capacity increases, the amount of traffic served by the peers increases almost linearly until it saturates to a maximum value obtained for the maximum streaming capacity. This result shows that depending on the storage capacity, increasing the streaming capacity will not improve the system's performance from a certain value. This value increases with the storage capacity. In Figure 5.4, it can also be seen that for a storage capacity of $s = 50$ strips, there is almost no improvement in the overall peer-assisted traffic for values of streaming capacities higher than $k = 6$ channels (1.2 Mbps). In the second case, however, the streaming capacity has considerable importance in the amount of peer-assisted traffic in almost entire range of values. Another interesting remark is that, although the storage capacity is doubled, the overall peer-assisted traffic does not double. Some of the reasons for such a behavior are that there is a different proportion of distribution of popular and not popular contents and the fact that although a peer might have a copy of a strip, it might not be able to serve it because it is busy with streaming strips of other, more popular items.

The maximum value that the peer-assisted traffic can reach can be determined for an infinite number of streaming channels so that $P_{p2p}(i)$ will depend only on the fact whether a strip is

stored in the peers, excluding the condition that the peer that contains a certain strip might be busy.

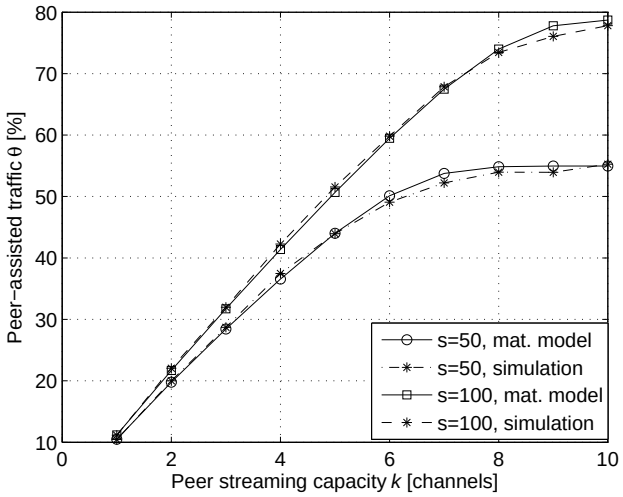


Figure 5.4.: Comparison between the simulation and the mathematical model results and influence of the peers streaming capacity k for two storage capacities $s = 50$ and $s = 100$ strips on the peer-assisted traffic θ.

Another important result from Figure 5.3 and Figure 5.4 is the mathematical model verification. It can be seen that the curves obtained from the mathematical model and the simulations overlap within the entire range of values. This overlapping proves that the mathematical model precisely models the system for peer-assisted VoD streaming with cooperative peers.

In Figure 5.5, the dependence of the uplink utilization η on the peers' storage capacity s for various streaming capacities of the peers is presented. As expected, the utilization improves as the space for storing the contents grows. However, it can be seen that at the beginning, the utilization considerably improves with the increase of the storage capacity, but later, although the storage capacity increases several times, the uplink utilization improves very slowly, asymptotically getting closer to a maximum value of saturation. This effect is especially noticeable for a lower uplink capacity. From Figure 5.5 it can be concluded that, although the storage capacity is increased to infinity, a full utilization of the links will never be reached. One could come to a similar conclusion by analyzing the results from Figure 5.6, where the portion of the overall peer-assisted traffic is shown. For small values of the storage capacity the quantity of traffic served by the peers is considerably getting higher as the storage space grows, and increases slowly with increasing the storage capacity, especially for small values of the uplink capacity. The theoretical maximum value that can be reached can be calculated by substituting $P_{p2p} = 1$ in (5.26) since for infinite storage capacity, there will be always a peer that contains a requested strip, even if there is a single

peer with an available streaming channel. The results from Figure 5.6 help to determine the optimal storage capacity of the peers, given the streaming capacity, so that a peer-assisted streaming utilization close to the maximum value is achieved.

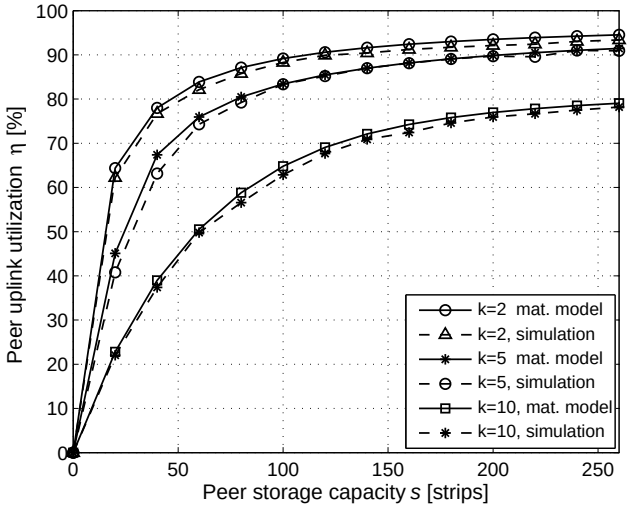


Figure 5.5.: Dependence of the peer uplink utilization η on the peer storage capacity s for different streaming capacities k and portion of storage dedicated to popular contents $l = 30\%$.

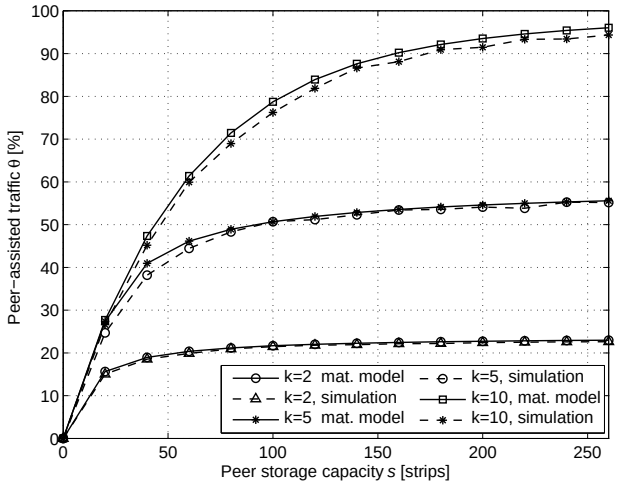


Figure 5.6.: Dependence of the peer-assisted traffic θ on the peer storage capacity s for different streaming capacities k and portion of storage dedicated to popular contents $l = 30\%$.

The distribution of the contents on the peers has also an important role in the system's performance. For a simpler analysis, the content distribution is obtained as joint uniform distributions of the popular and not popular contents in a way that different parts of the

STB storage capacity are dedicated to the popular and not popular contents. Thus, many different distributions are obtained by simply changing the percentage of dedicated space for popular contents l from 0 to 100 %.

The dependence of the overall peer-assisted traffic θ on the content distribution for a storage capacity of $s = 100$ strips for various streaming capacities of the peers is presented in Figure 5.7. The figure shows that for small uplink capacity of the peers, the distribution has almost no influence on the part of the traffic that is redirected to be served by the peers. In the case of higher uplink capacity, however, the distribution is of great importance for a better utilization of the uplink capacity of the peers. As more space for the popular contents is dedicated, the peer-assisted traffic increases and after reaching the maximum value, it decreases. This behavior shows that storing only the popular contents does not use optimally the streaming resources of the peers. On the contrary, reserving more space for the not popular contents will keep most of the channels busy for serving both the popular and not popular videos. From the figure it can be concluded that for high streaming capacities of the peers, choosing $l \approx 30\%$ gives best performance of the system, which justifies choosing this value in the initial simulations.

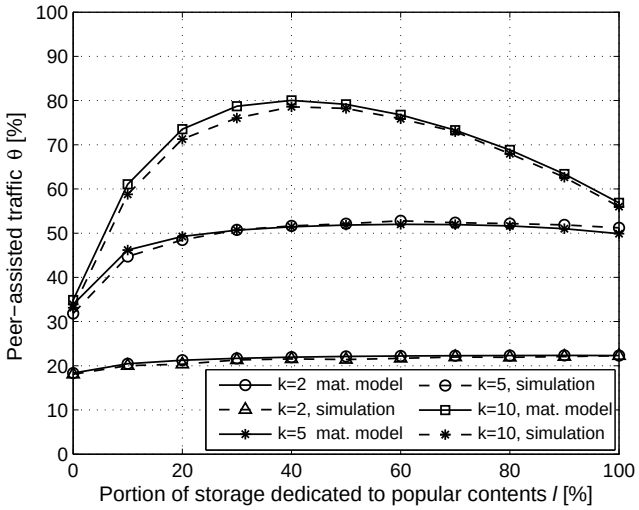


Figure 5.7.: Dependence of the peer-assisted traffic θ on the storage portion dedicated to popular contents l for different values of streaming capacity k with storage capacity $s = 100$ strips.

Since the distribution is an important factor in the utilization of the streaming resources of the peers in the cases when they have more dedicated channels for streaming, in Figure 5.8, it is presented how the distribution affects the performance of the system for various values of the storage capacity when the streaming capacity of the peers is $k = 5$ channels. The figure helps to figure out what portion of the storage to be dedicated to the popular contents,

given the streaming and storage capacity of the peers, so that the resources of the peers are optimally used, i.e., when the STBs have small storage capacity it is likely that bigger part of that space is dedicated to the popular contents so that the peer-assisted traffic is maximized. As the storage capacity grows, the maximum value of the peer-assisted traffic is reached for considerably smaller portions dedicated to the popular contents.

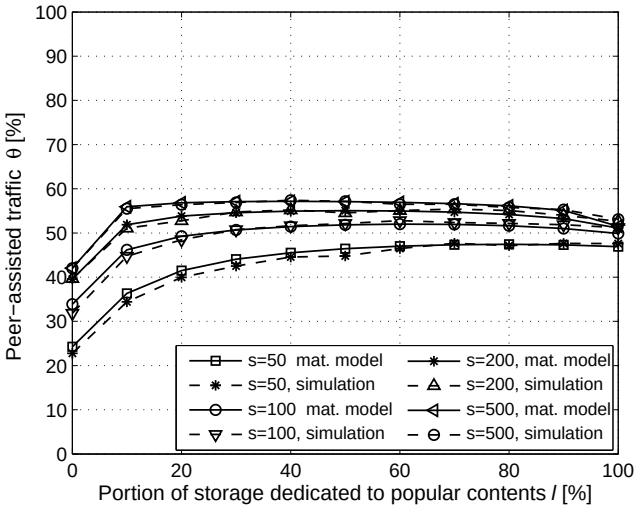


Figure 5.8.: Dependence of the peer-assisted traffic θ on the storage portion dedicated to popular contents l for different values of storage capacity s with streaming capacity $k = 5$ channels.

An important remark that can be taken from the figures presented in the analysis so far is that the simulation results still keep close to the results obtained from the mathematical model. Therefore, in the analysis that follows, only the results obtained from the mathematical model will be presented, and for clarity, the curves obtained from the simulations will be omitted.

In Figure 5.9 and Figure 5.10, the influence of the content distribution on the peer-assisted traffic θ is shown. In the analysis, there are considered three different distributions of the contents: uniform, linear and ZM distribution (with the same parameters as the distribution of the contents popularity). These distributions are applied to the entire content library, i.e., there is no division between popular and not popular contents. The results obtained for these distributions for different storage and streaming capacities of the peers are presented in Figure 5.9. The results show that for a small streaming capacity of the peers ($k = 2$ channels), the distribution has almost no influence on the peer-assisted traffic, no matter the storage capacity. With the increase of the streaming capacity to $k = 5$ channels, a small difference can be noticed in the peer-assisted traffic, but only for smaller storage capacities. In this case, the ZM distribution proves to give better results compared to the linear and the uniform distribution. However, when the streaming capacity is set to $k = 10$ channels,

it can be seen that the distributions do have more noticeable influence on the peer-assisted traffic which also depends on the storage capacity. For small storage capacities, the uniform distribution gives worst performances and then reaches the best performance for high storage capacity. The ZM distribution, on the other hand, has considerably better performance than any other distribution for small storage capacity, but, as the storage capacity increases, it performs worse compared to the other two distributions. The linear distribution has a medium performance compared to the ZM and uniform distribution for the initial and the final values of the considered range, however, it outperforms them for values in the middle of the range of storage capacities.

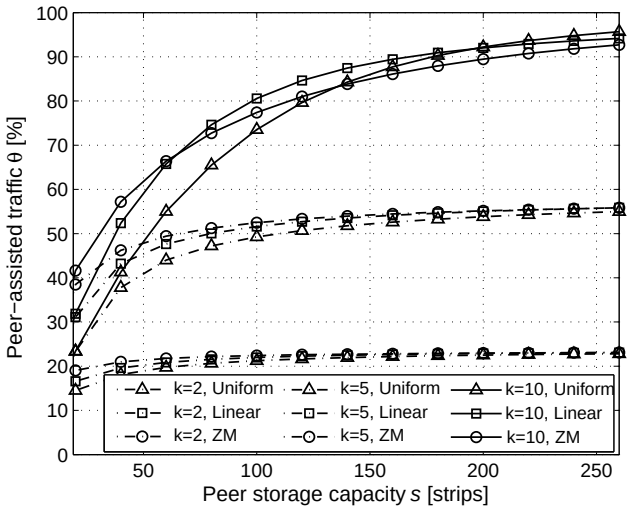


Figure 5.9.: Influence of the content distribution on the dependence of the peer-assisted traffic θ on the peers' storage capacity s for streaming capacity k=*2*, 5 and 10 channels and portion of storage dedicated to popular contents $l = 30\%$.

As the results in Figure 5.9 show that the distribution of the contents is of a higher importance when the storage capacity is smaller, the analysis is extended to finding out how more, different distributions based on the previous three will influence the peer-assisted traffic θ. For that purpose it is again considered the case when part of the storage is dedicated to the popular contents and other part to the not popular contents. The distribution is determined as a joint distribution of two simple distributions implemented on the reserved storage space, both for the popular and not popular contents. It is chosen a storage capacity of $s = 50$ strips, and then, the dedicated storage space for popular contents l is changed from 0 to 100%. There are also considered separate cases of streaming capacities of $k = 5$ and 10 channels. The case when $k = 2$ is intentionally omitted since the content distribution has little influence on the performance for small streaming capacities. Figure 5.10 shows the dependence of the peer-assisted traffic on the different content distribution schemes. From

the figure it can be seen that for small streaming capacities, the type of content distribution has insignificant importance, while for higher streaming capacity of the peers, the joint linear and uniform distributions contribute to slightly higher peer-assisted traffic in the range of the values of l when the system reaches best performance.

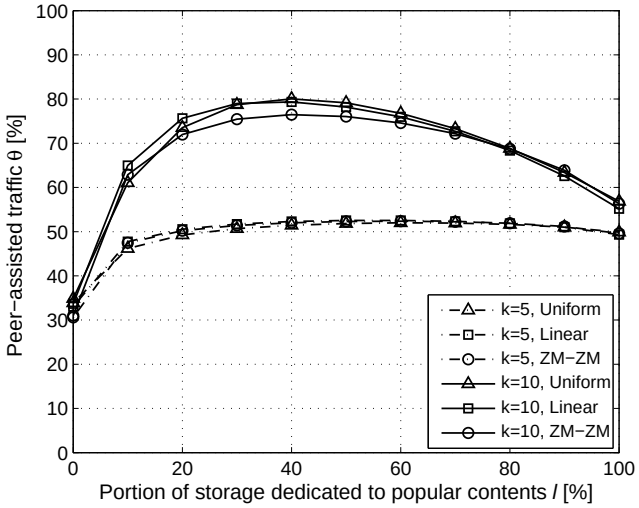


Figure 5.10.: Influence of the content distribution on the dependence of the peer-assisted traffic θ on the portion of storage dedicated to popular contents l for streaming capacity k and storage capacity $s = 50$ strips.

Figure 5.11 plots the dependence of the portion of the traffic served by the peers on the number of peers that form the local community with storage capacity of the peers of $s = 100$ strips. It can be concluded that the more peers the community has, the more traffic they serve. This effect is emphasized more when the uplink capacity of the peers is higher. From the figure, it can also be seen that adding new peers in the community will not contribute to considerable improvement in the utilization when the community gets large size. This can help the operators to estimate the size of the community as the number of the customers grows. On the one hand, the larger size implies more requests served by the peers, but on the other hand, higher number of peers would overload the ES. Thus, the model would help to determine which is the point when it is better to install a new ES and divide a growing community into two smaller communities.

Figure 5.12 shows the influence of the library size c on the peer-assisted traffic in the system θ for different values of the storage capacity s and uplink capacity of $k = 5$ channels. From the figure, it can be concluded that the amount of traffic streamed by the peers depends linearly on the size of the library. The growth of the size of the library contributes to lower peer-assisted traffic. This effect is especially noticeable for small storage capacities of the peers and diminishes as the storage capacity grows. This behavior of the system can be

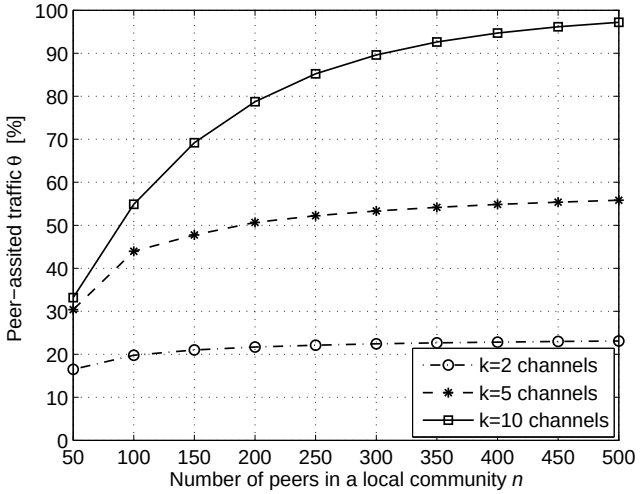


Figure 5.11.: Dependence of the peer-assisted traffic θ on the size of the local community n streaming capacity k=*2*, 5 and 10 channels, portion of storage dedicated to popular contents $l = 30\%$ and storage capacity $s = 100$ strips.

explained with the fact that for smaller sizes of the library more copies of each content item can be stored because the storage space is constant in size. Thus, the availability of the videos increases, and so does the peer-assisted traffic.

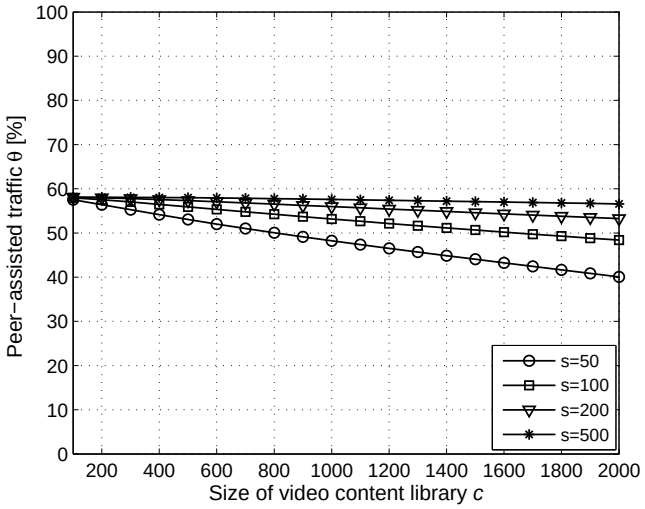


Figure 5.12.: Dependence of the peer-assisted traffic θ on the size of the content library c for portion of storage dedicated to popular contents $l = 30\%$ and, peers' storage capacity s=*50*, 100, 200 and 500 strips and streaming capacity of $k = 5$ channels.

5.3. Conclusions

This chapter presented a new stochastic model for peer-assisted streaming of VoD contents in privately managed networks with cooperative peers. The model enables estimation of the performance of the system for various parameters, and therefore, it is useful for the operators to better utilize the peers and to alleviate the servers employed for providing he highly traffic-demanding VoD service. The model considers a large variety of configurable system parameters which can be tested with the aim to estimate how they affect the performance of the system. Another advantage of the model is that it permits the estimation of the performance of arbitrary content distributions which could serve for verification of various algorithms for optimal distribution of contents on the peers.

The validity of the model has been verified by comparing it with the results obtained by simulations and has been used to make analyses on the influence of the system parameters on the utilization of the peers' uplink capacity, as well as, the portion of the overall traffic that is served by the peers. The aim of these analyses was to show that the model can be used by the network operators to estimate the limits and the optimal values of a large variety of parameters according to their specific network configuration.

One of the key contributions of this work is that the model can substitute the long lasting simulations or the tedious measurements of the real systems in order to estimate the changes in performance of the system due to modifications of the system environment. The model can be also easily extended for planning the network environment for offering VoD service for High Definition (HD) videos. Although the network technologies permit higher upload rates, most of the users still do not have the sufficient capacity to stream entire videos, especially when the operator offers HD quality content. Therefore, the model could serve as a flexible tool for planning privately managed environments for VoD streaming.

One of the disadvantages of the proposed model is that it treats the perfect scenario where the clients are constantly active, which is not the case in the real systems where the clients decide to turn off their STBs. In such cases, despite the control that the provider has over the STBs, the resources of the peers cannot be used for peer-assisted streaming since they are unreachable until the clients decide to turn the STB on and request a video. The efforts to make a more sophisticated model that will take into consideration the failures of the peers to make a model of more realistic system for peer-assisted VoD streaming are presented in the next chapter.

Chapter 6

Stochastic Modeling of Peer-assisted VoD Streaming with Non-cooperative Peers

Although the previous chapter gave a precise mathematical model of the peer-assisted VoD streaming system, it still misses important features so that the model can be a representative of a real system. The operator of the managed network that offers the VoD service can reserve part of the storage and streaming resources of the STBs for peer-assisted streaming, however, some of the resources cannot be guaranteed by the provider with certainty because there are some scenarios when it cannot have the complete control over the clients' STB. Such a scenario is the case when some clients decide not to cooperate in the peer-assisted streaming while they are not watching a video. They turn off their STBs after watching a video and keep them turned off until their next watching session. During this period of time, their streaming and storage resources are unavailable for the community they belong to.

One of the questions that address the failures is how their intensity and the time the peers spend in state of failure will affect the overall performance, especially the load on the servers. It is quite straight-forward that the failures of the peers will cause change of the size of the local community which is one of the most important parameters for the portion of the overall traffic served by the peers. However, what adds complexity to the further analysis is whether this will increase or decrease the load on the streaming servers. The negative effect of the failures is the increased load on the servers because, in the peer-assisted system, the peers also have the role of servers that stream strips of the videos to the other clients in the same local community. The failure of one peer means also interruption of all the strips it is streaming to other peers. In order to provide uninterrupted video service, upon detection of a failure,

the remaining content of the strips that were streamed by the failed peer are assigned to be streamed by the server. Thus, the server has to compensate for all the interrupted streams which results in increased traffic. The positive effect of the failure of the peers, however, comes from the fact that less peers implies less requests for videos and thus less traffic on the servers. Hence, one of the goals of this chapter is to determine the amount of extra traffic on the servers generated for the compensation of the strips streamed by the failed peers and the amount of reduced traffic due to the reduction in the number of requests. The answers to these questions will be given with the aid of the extended mathematical model which includes the possibility that the clients fail after they receive the entire content and the possibility that the failed peers recover after a certain time.

6.1. Mathematical model description

The model for peer-assisted VoD streaming with non-cooperative peers is extension of the model developed in the previous chapter. In order to include the failures of the peers in the streaming process, some changes have been made in the system to provide uninterrupted streaming. The system is monitoring the state of the STBs in the period after their streaming session ends, and if it detects that a peers has failed, it runs a procedure for compensating for all the streams that were streamed by the failed peer. In order to avoid long delays for the new recovery streams to begin, instead of searching for other candidate peers, which might be non-cooperative too, the system puts the reliable streaming servers in charge of finishing the interrupted streams. For every failure or recovery of a peer, the ES in charge of that peer updates an availability table which is used in the request process for redirecting the peers to other peers in the local community. The model assumes that the peers will not fail in the middle of the reception of a stream due to power failures or other unexpected interruptions, since power outages in a local community would also affect all the peers. It is also assumed that the number of peers in the local community is constant, i.e., the number of peers that subscribe to the VoD service or break the contract is the same. The number of failed peers can vary within the community, but, although a peer might fail, it is still part of the local community.

6.1.1. System parameters

As the mathematical model of the peer-assisted system for VoD streaming with non-cooperative peers is an extension of the basic model discussed in the previous chapter, only a brief review of the parameters will be given with focus on the new parameters that will be introduced in

the system for modeling the failures of the peers. A more detailed overview of the parameters is given in Table 6.1.

Table 6.1.: Overview of the system parameters used in the mathematical model for cooperative peer-assisted VoD streaming.

Parameter	Description
n	Number of peers in a community
n_{idle}	Number of idle peers
n_{rcv}	Number of receiving peers
n_{off}	Number of failed peers
m	Number of strips per video
k	Number of streaming channels per peer
s	Number of strips that can be stored on a peer
c	Size of video content library
μ	Average rate of service of videos by one peer or server
d	Average duration of a video session
λ	Average arrival rate of video requests generated by one client
w	Average time a peer waits to make a request
p_{off}	Failure probability of a receiving peer
μ_{off}	Recovery service rate of a failed peer
T_{off}	Recovery time of a failed peer
τ_{off}	Average recovery time of a peer
$\mathbf{k}(t) = (i, j, l)$	State vector of the system at time t, with i occupied channels on the peers,j occupied channels on the servers and l failed channels
$P_{i,j,l}(t)$	Probability that the system is in state (i, j, l) at time t
$P^{+}_{p,i,j,l}(\Delta t)$	Probability that there will be one arrival on the active peers within period Δt when the system is in state (i, j, l)
$P^{-}_{s,i,j,l}(\Delta t)$	Probability that there will be one departure on the server within period Δt when the system is in state (i, j, l)
$\lambda_{p,i,j,l}$	Average system arrival rate of requests that arrive at the peers when the system is in state (i, j, l)
$\mu_{s,i,j,l}$	Average system video service rate on the server when the system is in state (i, j, l)
$p_{i,j,l}$	Stationary probability that the system is in state (i, j, l)
$P_{p2p}(i, l)$	Probability that a request will be served by a peer when there are i busy channels on the peers and l failed channels

Continues...

Parameter	Description
$P(v)$	Probability that a request for item with rank v will be generated in the system
$P_x(v)$	Probability that item with rank v is chosen to be stored on the peers
$P_s(v)$	Probability that a strip of item with rank v is stored in a single peer
$P_s(v,i,l)$	Probability that a strip of video with rank v is stored in at least one available peer when the system has i busy channels on the peers and l failed channels
$P_s(v,t,i,l)$	Probability that a strip of video with rank v is stored in at least one of t available peers when there are i busy channels on the peers and l failed channels
z	Number of available channels on the peers
$w(t,z)$	Probability that z available channels are distributed on exactly t peers
η	Utilization of the streaming capacity of the active peers
θ	Portion of the streaming traffic served by the active peers
Φ	Portion of the streaming traffic served by the server relative to the maximum system's throughput

The system is defined as $\mathcal{X} = \{\mathcal{S}, \mathcal{P}, \mathcal{V}\}$, where $\mathcal{S}$ is the set of the streaming servers, $\mathcal{P}$ is the set of peers and $\mathcal{V}$ is the video content library. The set of servers $\mathcal{S}$ is represented by one server which serves one local community and has enough resources to stream any strips that cannot be served by the peers in the community.

The video content library $\mathcal{V}$ contains c videos with playback rate r. For modeling the users' behavior not to watch the entire video, it is considered that the average duration of the videos d is actually the average time they spend watching the videos. The popularity of each video item v from the library is expressed through the probability that it will be requested by a peer $P(v)$. The probability that a content will be chosen to be stored in a peer depends on the distribution scheme implemented by the service provider and is noted as $P_x(v)$. Because of the limited streaming capacity of the clients, the contents are divided into m strips with equal size. The time necessary to stream one strip is equal to the average watched video duration d, however, the streaming occupies m times less uplink capacity. The capacity r/m necessary to stream one strip is defined as a channel.

The set $\mathcal{P}$ represents the set of peers composing one local community. The number of peers subscribed to the VoD service is denoted as n, no matter whether their STBs are on or off.

Each peer has the capacity to store s strips of video contents and to stream k simultaneous strips. The peers are generating requests for videos according to the Poisson process with arrival rate λ. The process of streaming the strips also obeys the Poisson process with service rate $\mu = 1/d$, where d is the average duration a client spends watching a video. When the client finishes the streaming session, it decides to turn off the STB with probability p_{off}. This probability is called failure probability. After the failure, the average time the client spends until it turns the STB on is T_{off}, also called recovery time. This time has exponential distribution. Consequently, the process of the recovery of the peers is modeled as a Poisson process with service rate $\mu_{off} = 1/T_{off}$, also referred to as recovery rate. Another important parameter for the analysis that follows is the time that in average every peer in the systems spends in state of failure defined as $\tau_{off} = p_{off} T_{off}$.

6.1.2. Representation of the system as a network of queues

The general view of the system is an essential step for obtaining a better picture of the entire process of service with non-cooperative peers. It describes the system viewed as a whole, without entering into the insides of the system, i.e., ignoring the fact that the peers can assist in the streaming. In this view, the system is considered as a set of n peers that generate requests for videos, get served and occasionally fail. Since all of these processes are Poisson processes and the number of peers in a local community is constant, including the failed ones, the system can be considered as a closed network of queues with finite population, where each queue is one of the processes in which the peer can enter: waiting for request, getting video and failure. The relation of these queues is shown in Figure 6.1.

The waiting queue is serving the idle peers and its size in stationary state is n_{idle}. Idle peers are all those peers which are waiting to request a video. Although they are not receiving a video, they participate in the streaming of contents to other peers. The peers in the getting video queue are called receiving peers because they requested a video and are receiving strips of the video. The size of this queue in stationary state is n_{rcv}. The receiving peers are also participating in the streaming of contents to other peers. The idle and the receiving peers are altogether called active peers because they both participate in the process of streaming. Their cumulative number is noted as $n_{act} = n_{idle} + n_{rcv}$. The failure queue is "serving" the failed peers and has average size in stationary state n_{off}. The failed peers do not participate in the streaming process, i.e., they neither send nor receive streams.

For each request, the ES tries to find peers that have the strips of the required content and have available channel for streaming the strips. In the case when all the strips cannot be streamed by the peers, the rest of the strips are streamed by the server. No matter the availability of the contents and the occupancy of the peers, each request is immediately

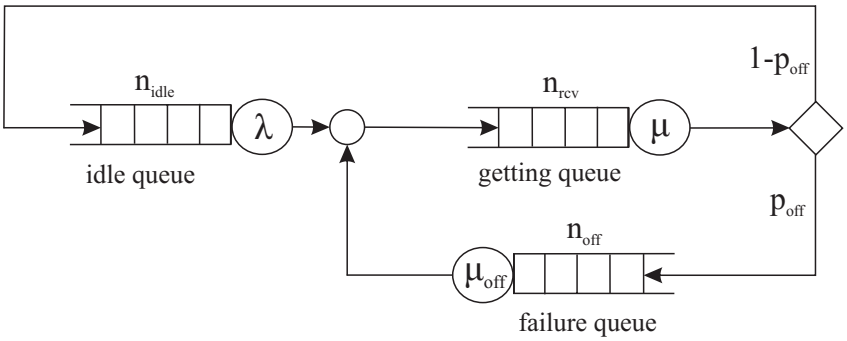


Figure 6.1.: General representation of the system as a closed network of queues with finite population.

served, i.e., the peer leaves the waiting queue and enters the getting video queue. It stays in this queue until it watches the video, which is in average $d = 1/\mu$, and then it decides with probability p_{off} whether to turn the STB off. If so, it enters the failure queue, where it stays in average $T_{off} = 1/\mu_{off}$. After this special service, the peer re-joins the community when it turns the STB on to make a new request and enters the getting video queue. This process is also called recovery process. In the opposite case, the peer decides to keep the STB on with probability $1 - p_{off}$ and let its resources available for serving other peers in the community. It stays in the waiting queue in average $w = 1/\lambda$ until it requests a new content and enters the getting video queue.

A more detailed representation of the system can be given if each peer that makes a request for m different strips of one video is treated as m peers that make independent request for a strip of a different video. In this case, the system is considered to have mn peers that make independent requests of only one strip of video. Applying this approach to the network of queues from Figure 6.1 and representing the server and the peers as separate service facilities, the detailed representation of the system gets the form shown in Figure 6.2. In the figure, the getting video queue is represented by two different queues: a server, represented by an $M/M/mn$ queue and a representative peer which has the streaming and storage capacity of all the peers in the local community, modeled as an $M/M/kn$ queue. The probability that a strip will be found on a peer with at least one available channel is P_{p2p} and is a complex function that depends on the current state of the system and the distribution of the contents on the peers. In the detailed representation of the system, the sizes of the waiting and the failure queues are mn_{idle} and mn_{off} since each peer is treated as m peers that request one strip. The failures of the peers give more complexity to the system since once a peer fails, the system has to handle all the streams that were streamed by the failed peer and pass them to the server. Thus, at the moment of a peer failure, some channels on the server are released because the streams received from the peer that fail are over, but some channels will be instantly occupied for providing uninterrupted service to the peers that were receiving

strips from the failed peer.

In order to obtain the portion of the traffic that is served by the peers, the average number of occupied channels on the peers or on the server has to be determined. Although the stationary value of the idle, receiving and failed peers can be obtained from Figure 6.1, the percentage of the streaming clients served by the peer or the server separately cannot be easily determined because of the complexity of calculation the probability P_{p2p} and the additional traffic burden on the servers caused by the failures.

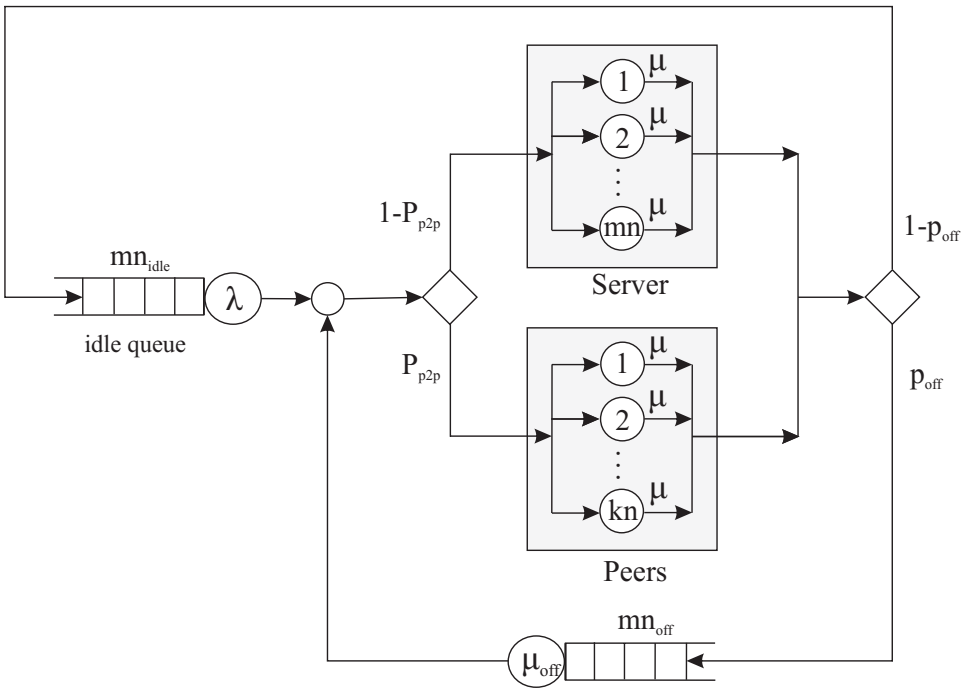


Figure 6.2.: Detailed representation of the system for non-cooperative VoD streaming as a closed network of queues with finite population.

6.1.3. The system's behavior as a birth-death process

The first step for obtaining the average number of busy channels of the active peers and the average number of busy channels on the server for serving the receiving peers and compensating for the failed peers is defining the set of states that can have the system in the time. Since each peer in the network of queues must be in any of the four queues (Figure 6.2), the state of the system at time t can be uniquely defined with the triple $\mathbf{k}(t) = (k_p(t), k_s(t), k_o(t))$, where $k_p(t) \in \{0, 1, ..., kn\}$ is the number of busy channels on the peers, $k_s(t) \in \{0, 1, ..., mn\}$ is the number of busy channels on the server, and $k_{off}(t) \in \{0, 1, ..., kn\}$ is the number of failed channels. The values of the state triple must satisfy the condition:

$$k_p(t) + k_s(t) + k_{off}(t) \leq mn \tag{6.1}$$

The average number of busy channels on any facility can be calculated if the probability of every possible state is found. Let one of the states of the system be $\mathbf{k}(t) = (i, j, l)$, describing the situation when there are i busy channels on the peers, j busy channels on the server and l failed channels at a given moment of time t, and let its probability be defined as $P_{i,j,l}(t) = P(\mathbf{k}(t) = (i, j, l))$. Based on the exponential nature of the service times of all the facilities, the whole system represents a Markov population process. Therefore, in an infinitesimal small period of time Δt, there can be only one request, one end of streaming, one failure, one recovery or none of the four events. More precisely, if the system is in a state $\mathbf{k}(t+\Delta t) = (i, j, l)$ at time $t+\Delta t$, also referred to as a central state, only one of the following events could have happened during the interval Δt:

1. the system had been in state $\mathbf{k}(t) = (i, j-1, l)$ and a request for a strip arrived at the server from an idle peer,
2. the system had been in state $\mathbf{k}(t) = (i, j-1, l+1)$ and a request for a strip arrived at the server from a failed peer,
3. the system had been in state $\mathbf{k}(t) = (i-1, j, l)$ and a request for a strip arrived at an active peer from an idle peer,
4. the system had been in state $\mathbf{k}(t) = (i-1, j, l+1)$ and a request for a strip arrived at an active peer from a failed peer,
5. the system had been in state $\mathbf{k}(t) = (i, j+1, l)$ and a streaming of a strip has been completed on the server without failure of the peer,
6. the system had been in state $\mathbf{k}(t) = (i, j+1, l-1)$ and a streaming of a strip has been completed on the server with failure of the peer,
7. the system had been in state $\mathbf{k}(t) = (i+1, j, l)$ and a streaming of a strip has been completed on a receiving peer without failure of the peer,
8. the system had been in state $\mathbf{k}(t) = (i+2, j-1, l-1)$ and a streaming of a strip has been completed on a receiving peer with failure of the peer,
9. the system had been in state $\mathbf{k}(t) = (i, j, l)$ and neither request arrived nor a stream has been completed on any of the service facilities.

The explanations of these conditions are quite straight-forward except the one for condition (8). According to this condition, the central state (i, j, l) can be reached when a receiving peer finishes the reception of a stream from the peers and immediately fails, provided that the system was previously in state $(i+2, j-1, l-1)$. The reason for decreasing the number

of channels on the peers for reaching the central state by two, instead by one, is that if one stream that originates from a peer finishes and the peer fails, it means that the peer that fails is a receiving peer, which also streams one stream to the other peers in the community. Therefore, besides the finished stream, there is also one interrupted stream due to the failure of the peer. This comes from the fact that if a receiving peer fails, the number of finished strips that come to it from other receiving peers is the same as the number of interrupted streams.

This can be proven from a detailed observation of the peer-assisted streaming process shown in Figure 6.3. The figure shows the three different categories of peers: the idle peers which only stream strips, the receiving peers which stream and receive strips, and the failed peers which do not participate in any of these activities. The number of outgoing streams k_{out} that originate from the active peers (idle and receiving) is the same, because as long as the peers are active, it has no importance to the ES whether they are idle or receiving in the moment when it assigns them to serve a request for a strip. The total number of input streams that receives every receiving peer is the number of strips that composes the video m.

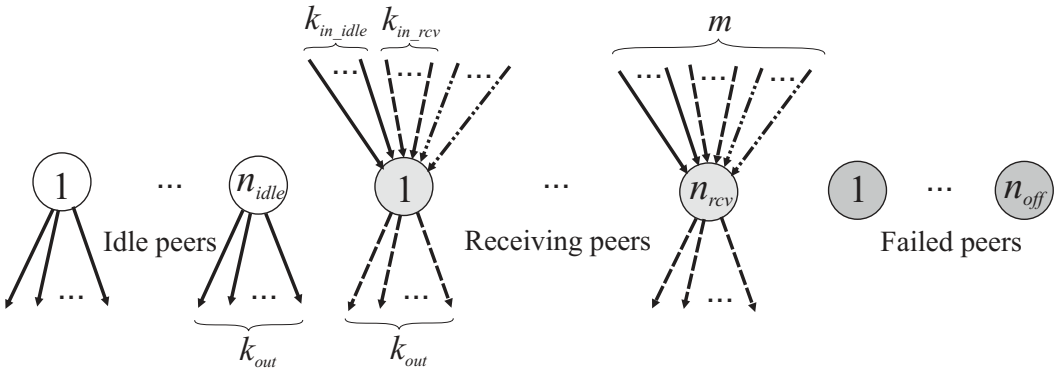


Figure 6.3.: Illustration of non-cooperative peer-assisted VoD streaming.

From all these input streams, k_{in_idle} originate from the idle peers, k_{in_rcv} originate from the receiving peers and the rest of the streams originate from the server. The total number of outgoing strips from the receiving peers is $n_{rcv}k_{out}$ and the total number of incoming streams that each receiving peer receives from the receiving peers is $k_{in\_rcv} = (n_{rcv} - 1)k_{out}/(n_{rcv} - 1) = k_{out}$. This means that when k_{in_idle} streams finish and the receiving peer fails, there will be also $k_{out} = k_{in\_rcv}$ interrupted streams. The failure of these strips will increase the number of failed strips in the system by k_{out} but it will also increase the number of strips served by the servers by k_{out} because the server has to compensate for the interrupted strips. As a conclusion, the end of k_{in_rcv} strips with failure of the peer implies reduction of the overall number of strips streamed by the peers by $2k_{in\_rcv}$ and increment of the number of failed streams and the number of streams served by the server by k_{in_rcv}, i.e., the central state (i, j, l) can be reached only from the state $(i + 2k_{in\_rcv}, j - k_{in\_rcv}, l - k_{in\_rcv})$. Substituting

$k_{in\_rcv} = 1$ will give the originating state $(i+2, j-1, k-1)$ in the condition (8).

From the conditions (1)-(9), the probability $P_{i,j}(t+\Delta t)$ that the system will be in state $\mathbf{k}(t+\Delta t) = (i,j,l)$ can be expressed as:

$$\begin{aligned} P_{i,j}(t+\Delta t) &= P_{i,j-1,l}(t)P^{+}_{s,i,j-1,l}(\Delta t) + P_{i,j-1,l+1}(t)P^{+}_{s,i,j-1,l+1}(\Delta t) + \\ &+ P_{i-1,j,l}(t)P^{+}_{p,i-1,j,l}(\Delta t) + P_{i-1,j,l+1}(t)P^{+}_{p,i-1,j,l+1}(\Delta t) + \\ &+ P_{i,j+1,l}(t)P^{-}_{s,i,j+1,l}(\Delta t) + P_{i,j+1,l-1}(t)P^{-}_{s,i,j+1,l-1}(\Delta t) + \\ &+ P_{i+1,j,l}(t)P^{-}_{p,i+1,j,l}(\Delta t) + P_{i+2,j-1,l-1}(t)P^{-}_{p,i+2,j-1,l-1}(\Delta t) + \\ &+ P_{i,j,l}(t)\cdot\left(1-P^{+}_{s,i,j,l}(\Delta t)\right)\left(1-P^{+}_{p,i,j,l}(\Delta t)\right)\cdot \\ &\quad \cdot\left(1-P^{-}_{s,i,j,l}(\Delta t)\right)\left(1-P^{-}_{p,i,j,l}(\Delta t)\right) \end{aligned} \tag{6.2}$$

where the probabilities of any of the above defined events during the interval Δt are marked in a way that the superscript $+/-$ denotes whether an arrival or an end of service has happened, the first subscript denotes the service facility where the event has happened, with notation p referring to a peer and s to a server, and the remaining three subscripts denote the previous state of the system, before the event happened.

The first four lines of the expression refer to the conditions (1) to (8), accordingly, while the end of the expression refers to the condition (9) representing the joint probability that nor arrival of request, neither end of streaming will happen in the system when it is in state $\mathbf{k}(t) = (i,j,l)$.

Using the assumption (5.5) for the probability that one event of a Poisson process will happen within an infinitesimal period of time Δt for the event probabilities in (6.2), making some rearrangements and dividing the whole expression by Δt, reduces (6.2) to:

$$\begin{aligned} \frac{P_{i,j,l}(t+\Delta t) - P_{i,j,l}(t)}{\Delta t} &= \lambda_{s,i,j-1,l}P_{i,j-1,l}(t) + \lambda_{s,i,j-1,l+1}P_{i,j-1,l+1}(t) + \\ &+ \lambda_{p,i-1,j,l}P_{i-1,j,l}(t) + \lambda_{p,i-1,j,l+1}P_{i-1,j,l+1}(t) + \\ &+ \mu_{s,i,j+1,l}P_{i,j+1,l}(t) + \mu_{s,i,j+1,l-1}P_{i,j+1,l-1}(t) + \\ &+ \mu_{p,i+1,j,l}P_{i+1,j,l}(t) + \mu_{p,i+2,j-1,l-1}P_{i+2,j-1,l-1}(t) - \\ &- (\lambda_{s,i,j,l} + \lambda_{p,i,j,l} + \mu_{p,i,j,l} + \mu_{s,i,j,l})P_{i,j,l}(t) + \frac{o(\Delta t)}{\Delta t} \end{aligned} \tag{6.3}$$

The equation for each state in the system represents the dynamics of the probability flow of the state. For $\Delta t \to 0$:

$$\lim_{\Delta t \to 0} \frac{P_{i,j,l}(t+\Delta t) - P_{i,j,l}(t)}{\Delta t} = \frac{dP_{i,j,l}(t)}{dt} = 0 \tag{6.4}$$

Another simplification that can be introduced is

$$\lim_{\Delta t \to 0} o(\Delta t)/\Delta \mathrm{t}{=}0 \tag{6.5}$$

because $o(\Delta \mathrm{t})$ is the infinitesimal probability that more than one event will happen within the time interval Δt.

Defining the value of the probability $P_{i,j,l}(t)$ in stationary state as:

$$p_{i,j,l} = \lim_{t \to \infty} P_{i,j,l}(t) \tag{6.6}$$

and substituting the last equations in (6.3), the system becomes:

$$\begin{aligned} 0 \;=\; & \lambda_{s,i,j-1,l} p_{i,j-1,l} + \lambda_{s,i,j-1,l+1} p_{i,j-1,l+1} + \\ + \; & \lambda_{p,i-1,j,l} p_{i-1,j,l} + \lambda_{p,i-1,j,l+1} p_{i-1,j,l+1} + \\ + \; & \mu_{s,i,j+1,l} p_{i,j+1,l} + \mu_{s,i,j+1,l-1} p_{i,j+1,l-1} + \\ + \; & \mu_{p,i+1,j,l} p_{i+1,j,l} + \mu_{p,i+2,j-1,l-1} p_{i+2,j-1,l-1} - \\ - \; & (\lambda_{s,i,j,l} + \lambda_{p,i,j,l} + \mu_{p,i,j,l} + \mu_{s,i,j,l}) p_{i,j,l} \end{aligned} \tag{6.7}$$

6.1.4. Determining the coefficients of the system of equations for the system's state probabilities

The system (6.7) can be solved once the departure and arrival rates that lead to the central state $p_{i,j,l}$ are determined.

The coefficient $\lambda_{s,i,j-1,l}$ refers to the arrival rate of requests directed to the server from the idle peers when the system is in state $(i, j-1, l)$. Its value depends on the number of peers in the waiting queue, obtained by subtracting the number of peers in the getting video and failure queues from the overall number of peers in the system, and the arrival rate of each request λ. From all the candidate peers in the waiting queue, only those that cannot be served by the peers will be directed to the server. The portion of these requests is $1 - P_{p2p}(i, l)$, where $P_{p2p}(i, l)$ is the probability that a strip can be served by a peer when there are i busy channels on the peers and l failed channels. The probability of finding a peer that will stream a strip is a function of i and l because they are necessary to determine the number of potential

active peers which could possibly serve the request in case they have available channels and a copy of the requested strip. Eventually, the final value of the intensity of requests directed to the server when the system is in state $(i, j-1, l)$ is:

$$\lambda_{s,i,j-1,l} = (1 - P_{p2p}(i,l))(mn - i - j + 1 - l)\lambda \tag{6.8}$$

The coefficient $\lambda_{s,i,j-1,l+1}$ determines the arrival rate of requests directed to the server from the failed peers when the system is in state $(i, j-1, l+1)$. Any of the $l+1$ peers in the failure queue can recover with rate μ_{off} and independently request a strip, however, only a portion $1 - P_{p2p}(i, l+1)$ of them will be directed to be served by the server. Therefore, the intensity of the probability flow from state $(i, j-1, l+1)$ will be:

$$\lambda_{s,i,j-1,l+1} = (1 - P_{p2p}(i,l+1))(l+1)\mu_{off} \tag{6.9}$$

The coefficient $\lambda_{p,i-1,j,l}$ determines the arrival rate of requests directed to the active peers from the idle peers when the system is in state $(i-1, j, l)$. It is similar to (6.8) because the peers that make the requests are those in the waiting queue, but it differs in the portion of requests that will be directed to the peers which is determined by the probability $P_{p2p}(i-1, l)$ that the peer is able to serve the required strip when the system is in state $(i-1, j, l)$:

$$\lambda_{p,i-1,j,l} = P_{p2p}(i-1,l)(mn - i + 1 - j - l)\lambda \tag{6.10}$$

The coefficient $\lambda_{p,i-1,j,l+1}$ determines the arrival rate of requests directed to the active peers from the failed peers when the system is in state $(i-1, j, l+1)$. Its value is obtained by multiplying the rate of recovery of the peers μ_{off} with the size of the failure queue, which is $l+1$, and the probability that the peers can serve the requested strip:

$$\lambda_{p,i-1,j,l+1} = P_{p2p}(i-1,l+1)(l+1)\mu_{off} \tag{6.11}$$

The coefficient $\mu_{s,i,j+1,l}$ determines the departure rate of finished streams on the receiving peers from the server without failure of the peer when the system is in state $(i, j+1, l)$. In this state, there are $j+1$ streams that could possibly end but only a portion of $1 - p_{off}$ of them will decide not to fail. Therefore, its value is expressed as:

$$\mu_{s,i,j+1,l} = (1 - p_{off})(j+1)\mu \tag{6.12}$$

The coefficient $\mu_{s,i,j+1,l-1}$ determines the departure rate of finished streams on the receiving peers from the server with failure of the receiving peer when the system is in state $(i, j+$

$1, l-1$). Its value is similar to the value obtained in the previous expression, with the difference that the portion of candidates that would finish the stream with failure is equal to the probability of failure p_{off}:

$$\mu_{s,i,j+1,l-1} = p_{off}(j+1)\mu \tag{6.13}$$

The coefficient $\mu_{p,i+1,j,l}$ determines the departure rate of finished streams on the receiving peers streamed from the active peers without failure of the peer that receives the stream, when the system is in state $(i+1, j, l)$. The explanation of the expression for this coefficient is the same as (6.12), with the difference that the streams are originating from the active peers:

$$\mu_{p,i+1,j,l} = (1-p_{off})(i+1)\mu \tag{6.14}$$

The coefficient $\mu_{p,i+2,j-1,l-1}$ determines the departure rate of finished streams on the receiving peers streamed from the active peers with failure of the peer that receives the stream when the system is in state $(i+2, j-1, l-1)$. Its value is obtained by multiplying the video service rate μ with the size of the getting video queue $i+2$ and the probability that each of the peers will decide to fail p_{off}:

$$\mu_{p,i+2,j-1,l-1} = p_{off}(i+2)\mu \tag{6.15}$$

In order to obtain the values of the remaining coefficients, one should examine more precisely the last member of the expression (6.2) which denotes the probability that neither arrival nor departure happens when the system is in state (i, j, l). The first multiplier $1 - P^{+}_{s,i,j,l}(\Delta t)$ denotes the probability that there will be no arrival for a request on the server. There are two situations when arrival can happen on the server: when an idle peer makes a request, and when a failed peer makes a request immediately after the recovery. Therefore, the overall rate of arrivals will be the sum of the rates of each of these events. The final value can be obtained by adding or subtracting such values to the indexes of the functions defined in (6.8) and (6.9), so that they get the values i, j and l. The result of this operation is:

$$\begin{aligned} \lambda_{s,i,j,l} &= \lambda_{s,i,j-1+1,l} + \lambda_{s,i,j-1+1,l+1-1} = \\ &= (1 - P_{p2p}(i,l))(mn - i - j - l)\lambda + (1 - P_{p2p}(i,l))l\mu_{off} \end{aligned} \tag{6.16}$$

In a similar fashion, from the second multiplier $1 - P^{+}_{p,i,j,l}(\Delta t)$, the arrival rate of requests on

the peers is calculated from (6.10) and (6.11):

$$\begin{aligned}\lambda_{p,i,j,l} &= \lambda_{p,i-1+1,j,l} + \lambda_{p,i-1+1,j,l+1-1} = \\ &= P_{p2p}(i,l)(mn-i-j-l)\lambda + P_{p2p}(i,l)l\mu_{off} \end{aligned} \tag{6.17}$$

The calculation of the coefficient $\mu_{s,i,j,l}$ comes from the third multiplier $1 - P^{-}_{s,i,j,l}(\Delta t)$ of (6.2),which determines the probability that there will be no end of a stream on a server within the interval Δt when the system is in state (i, j, l). The end of a stream on a server implies two possibilities: that the receiving peer will not fail, and that the receiving peer will fail. The sum of the rates of these two events is the total departure rate of finished streams on the server. Like in the previous cases, it will be obtained from (6.12) and (6.13) by adjusting their indexes to values i, j and l:

$$\begin{aligned}\mu_{s,i,j,l} &= \mu_{s,i,j+1-1,l} + \mu_{s,i,j+1-1,l-1+1} = \\ &= p_{off}j\mu + (1-p_{off})j\mu = j\mu \end{aligned} \tag{6.18}$$

In a similar way, from the fourth multiplier $1 - P^{-}_{p,i,j,l}(\Delta t)$ of (6.2), (6.14) and (6.15), the coefficient $\mu_{p,i,j,l}$ will be calculated as:

$$\begin{aligned}\mu_{p,i,j,l} &= \mu_{p,i+1-1,j,l} + \mu_{p,i+2-2,j-1+1,l-1+1} = \\ &= (1-p_{off})i\mu + p_{off}i\mu = i\mu \end{aligned} \tag{6.19}$$

In the expressions for the coefficients, the only unknown variable is the probability $P_{p2p}(i, l)$ that a request for a strip will be redirected to be served by the peers when the system is in any of the set of states (i, j, l) such that $j \in [0, mn - i - l]$. The method for determining this value justifies the representation of all the peers in Figure 6.2 with one representative peer that has the cumulative streaming and storage resources of all the peers. The probability $P_{p2p}(i, l)$ depends only on the number of busy channels on the peers i and the number of failed channels l because, as far as the service of the peer is concerned, it is important to determine how many available channels are there in the system that could serve the request. For the given set of states such that $j \in [0, mn - i - l]$, the number of available channels is $kn - i - l$. These available channels can be distributed on various ranges of peers. The minimum number of peers that can have that many available channels is $\lceil (kn - i - l)/k \rceil$,

which is the case when almost each of these peers has k channels available. The maximum number of peers that can have that number of available channels is $\min(n, kn-i-l)$. In the last case, the minimum of the two values is used because if the number of available channels is lower than the number of peers, then the result would be distributing one available channel on each of the peers. In the opposite case, the maximum number of peers that can have that number of available channels will be n, where each of these peers will have at least one available channel. Having the minimum and maximum values, the range of peers that can have $kn-i-l$ available channels can be defined as:

$$R = \left[\left\lceil \frac{kn-i-l}{k} \right\rceil, \min(n, kn-i-l)\right] \tag{6.20}$$

By using the probability $P_s(v)$ that a video item v is stored in one peer, defined in (5.14), this range of values can be used for finding the probability that the content is stored in at least one peer. For every value t of the range R, there will be obtained a different probability $P_s(v,i,j,t)$ of finding at least one peer among t that has a copy of the requested strip of the content v, according to equation (5.16). However, although each distribution of the available channels among the values of R is possible, they appear with different probability. The probability that $z = kn-i-l$ available channels will be distributed on t nodes is $w(t,z)$ and is calculated by the expression (5.24) obtained by using the product of the generating function of the system defined in (5.22). From the probability $P_s(v,i,l,t)$ of finding at least one peer among t that has a copy of the requested strip of the content v and the probability of each value of the range R, the probability $P_s(v,i,l)$ that there will be found an available peer that has a copy of the requested strip of the item v when the system is in a state with i busy channels and l failed channels will be calculated as:

$$P_s(v,i,l) = \sum_{t \in R} w(t,z) P_s(v,i,l,t) \tag{6.21}$$

Using the last expression and the probability $P(v)$ of appearance of item v, the required probability of redirecting a request for a strip of any video item will be:

$$P_{p2p}(i,l) = \sum_{v=1}^{c} P(v) P_s(v,i,l) \tag{6.22}$$

Substituting the coefficients (6.8)-(6.19) in (6.7) will give the final equation for the probability of each state in the system:

$$
\begin{aligned}
0 &= (1 - P_{p2p}(i,l))(mn - i - j + 1 - l)\lambda p_{i,j-1,l} + \\
&+ (1 - P_{p2p}(i,l+1))(l+1)\mu_{off} p_{i,j-1,l+1} + \\
&+ P_{p2p}(i-1,l)(mn - i + 1 - j - l)\lambda p_{i-1,j,l} + \\
&+ P_{p2p}(i-1,l+1)(l+1)\mu_{off} p_{i-1,j,l+1} + \\
&+ (1 - p_{off})(j+1)\mu p_{i,j+1,l} + p_{off}(j+1)\mu p_{i,j+1,l-1} + \\
&+ (1 - p_{off})(i+1)\mu p_{i+1,j,l} + p_{off}(i+2)\mu p_{i+2,j-1,l-1} - \\
&- ((mn - i - j - l)\lambda + l\mu_{off} + (i+j)\mu)p_{i,j,l}
\end{aligned}
\tag{6.23}
$$

If this equation is written for every state of the system, then a system of linear equations will be obtained, which if solved, will give the percentage of the traffic that is originating from the peers and the server, separately. In order to make a clearer picture of the system of linear equations, Figure 6.4 shows a partial diagram of flow of probabilities. In the diagram, the probability of every state of the system is represented by an elliptic node and the flow of probability from one state to another is represented by an arrow which designates the direction of the flow. The intensity of each flow is the coefficient that multiplies the probability state in equation (6.23), but for the sake of clarity, it is intentionally omitted in the figure. The self-loop representing the probability flow from the central state to itself is also omitted. Because of the large size and complexity of the diagram, only the probability of the central state (i, j, l) and the probabilities of the states from which the central state can be reached are emphasized in the figure. The figure contains $kn + 1$ plains, and each plain contains the probabilities of the states with the same number of busy channels on the peers. In the partial diagram of a single plane, the probabilities of the states with the same number of failed channels are placed in the same row. The probabilities of the states with the same number of busy channels on the server are placed in a same diagonal. In the partial diagram that refers to the states that have i busy channels on the peers, the first row (the furthermost row) has $mn - i + 1$ states, and each following row has one state less than the previous. The last (closest to the viewer) row of each partial diagram has only one state.

The size of the system depends on the number of peers in the system n, the number of strips of each video m, and the number of channels of the peers k. From the figure, it can be seen that the size can be obtained by summing the size of the partial diagrams of the separate plains. Considering the fact that the lowest plain has $mn + 1$ rows, and each plain above has one row less than the row below, reaching the highest plain with $mn - kn + 1$ rows, and using the fact that each row of the partial diagrams has one state less than the previous, the size of the system is obtained as $(nk+1)\left(n^2(6m^2 + 2k^2 - 9mk) + n(18m - 8k) + 12\right)/12$

(Appendix B). Since the size of the system reaches high values even for small values of n, m and k, in Appendix B a sampling method for reduction of the size of the diagram is proposed.

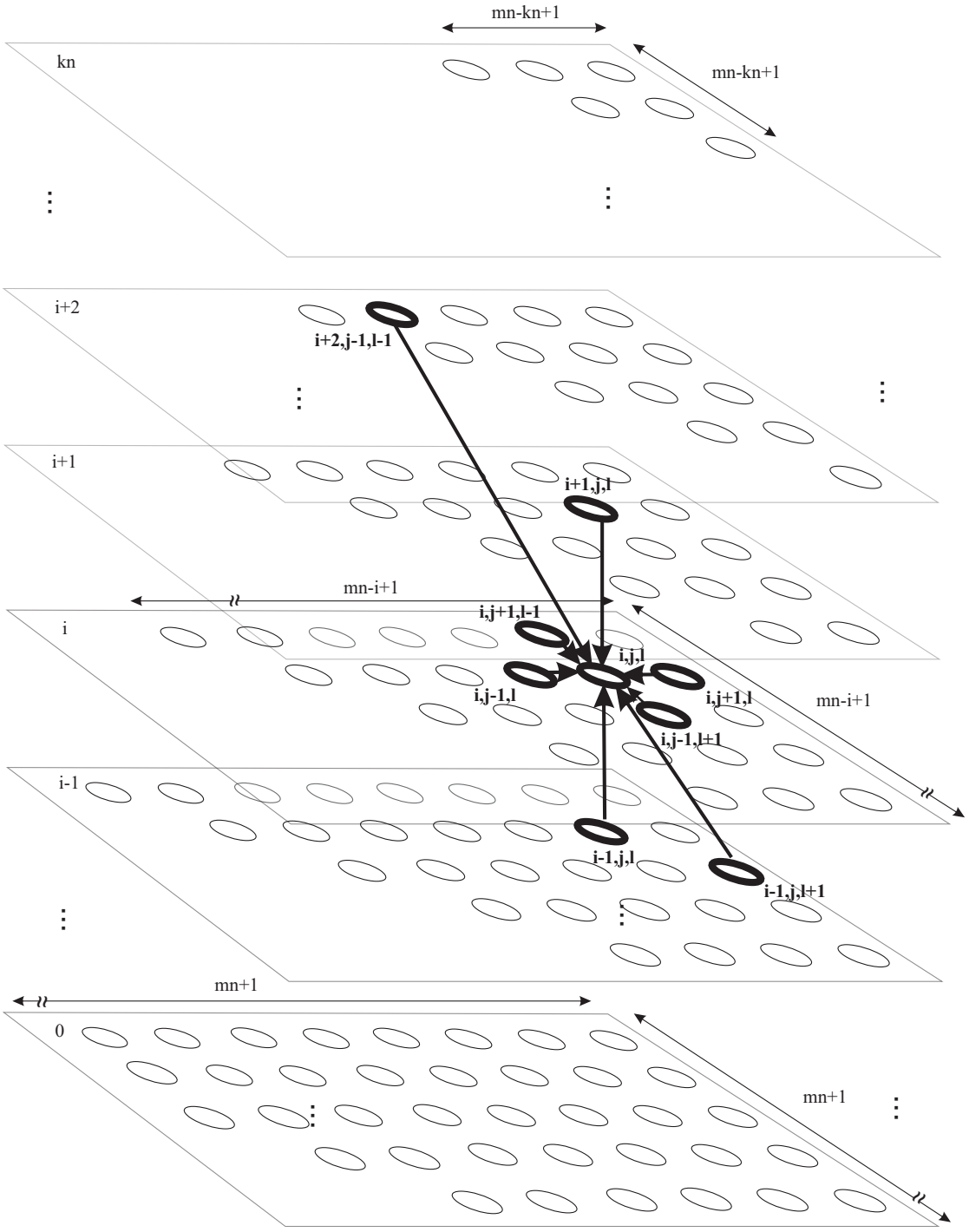


Figure 6.4.: Partial diagram of flow of probabilities of the system.

Because of the huge size of the system and the fact that most of the state probabilities can be reached by up to 9 state probabilities, the accurate determination of the coefficients of

all the probability flows directed to each of the state probabilities seems rather difficult task to accomplish. However, this task can be easily accomplished if the chosen probability of the central state in Figure 6.4 is more carefully analyzed. It can be seen that this state receives flows from the plain where it is situated, but also from the plan below, above and two plains above. An easy way to identify all the probabilities of states that have flow to a given probability of state is to do the movements, if possible, shown in the first column of Table 6.2, starting from the probability of state of interest. The triple of indexes of any of the reachable probabilities of state is obtained by applying the action in the second column of Table 6.2 to the indexes of the probability of state taken as a departure point, corresponding to the movement.

Table 6.2.: Overview of the movements in the state diagram and their corresponding actions for obtaining the indexes of the states that lead to the central state.

Movement	Action
one step right	add $+1$ to j
one step left	add -1 to j
one step up	add $+1$ to j and -1 to l
one step down	add -1 to j and $+1$ to l
one step above	add $+1$ to i
one step below	add -1 to l
one step below, one step right and one step down	add -1 to i and $+1$ to l
two steps above, two steps left and one step up	add $+2$ to i, -1 to j and -1 to l

6.1.5. Final results

The solution of the system of linear equations (6.23) can be obtained by presenting the system in a matrix form and performing simple multiplication operations. Therefore, the unknown stationary probabilities of the states are presented with a column vector $\mathbf{p} = [p_{0,0,0}\ p_{0,0,1}\ \cdots\ p_{i,j,l}\ \cdots\ p_{kn,(m-k)n,0}]^T$ and the coefficients that multiply these variables are presented with a coefficient matrix $A = [\mathbf{a}_{0,0,0}^T\ \mathbf{a}_{0,0,1}^T\ \cdots\ \mathbf{a}_{i,j.l}^T\ \cdots\ \mathbf{a}_{kn,(m-k)n,0}^T]^T$, where $\mathbf{a}_{i,j,l}$ is a row vector with the same size as $\mathbf{p}$ such that contains the weights of the arcs on the state diagram in (Figure 6.4) that go out of the probability of the state $p_{i,j,l}$ on the positions corresponding to the positions of the states in the vector $\mathbf{p}$ that they are directed to. The matrix form of the system of linear equations is $A\mathbf{p} = \mathbf{0}$ and it has an infinite number of solutions because of the singularity of the matrix A. To avoid this situation, the coefficient matrix is modified by using the condition that the sum of the probability of all the states

equals 1. The result is the modified coefficient matrix $B = [\mathbf{1}^T \ \mathbf{a}_{0,0,1}^T \ \ldots \ \mathbf{a}_{kn,(m-k)n,0}^T]^T$ obtained when the first row of the matrix A is substituted by the row vector $\mathbf{1}$ with the same size as the row it substitutes. This change also implies substitution of the column vector $\mathbf{0}$, by a column vector $\mathbf{b} = [1 \ 0 \ \ldots \ 0]^T$ which has 0 values on every position, except on the first position which has value 1. After the substitution, the solution is obtained by a simple multiplication of the inverse of the modified coefficient matrix B and the column vector $\mathbf{b}$:

$$\mathbf{p} = B^{-1}\mathbf{b} \tag{6.24}$$

The average number of busy channels on the peers can be obtained as the expected number of busy channels in the system. For that purpose, the probability that the system has exactly i busy channels on the peers in stationary state has to be calculated. This value is obtained by summing the probabilities of all the states that lie in the plain in the diagram of probabilities shown in Figure 6.4 corresponding to i busy channels:

$$P_i = \sum_{j=0}^{mn-i} \sum_{l=0}^{mn-i-j} p_{i,j,l} \tag{6.25}$$

If this value is used in the expression for the expected value of a random variable, it will be obtained that the mathematical expectation of the number of busy channels $\overline{K}$ in the system is (same as (5.29)):

$$\overline{K} = E[K] = \sum_{i=1}^{kn} iP_i \tag{6.26}$$

The average utilization of the streaming capacity can be calculated if the average number of busy channels is divided by the total number of active clients that take part in the streaming process. Therefore, the number n_{idle} of idle peers and the number n_{rcv} of receiving peers have to be calculated. For that purpose it can be used the condition for equilibrium for the network of queues in Figure 6.1 in stationary state. Since the considered network of queues is a closed system with finite population, in stationary state, it will achieve equilibrium, i.e., there will be no change of the number of customers (channels) in each service facility. The invariant number of customers in the service facilities means that there is also equilibrium in the flows of customers between adjacent service facilities, i.e., the number of customers entering a service facility in a given time interval must equal the number of customers that leave the facility in the same interval. Taking into consideration this fact, the following equations are derived:

$$\lambda n_{idle} = (1 - p_{off})\mu n_{rcv} \tag{6.27}$$

$$\mu_{off} n_{off} = p_{off}\mu n_{rcv} \tag{6.28}$$

$$\mu n_{rcv} = \lambda n_{idle} + \mu_{off} n_{off} \tag{6.29}$$

From the closed nature of the network of queues where the customers only pass from one queue to another, it can be written:

$$n_{idle} + n_{rcv} + n_{off} = n \tag{6.30}$$

Using any two of the equations (6.27)-(6.29) and (6.30) will give the expression for the number of active peers in the system:

$$n_{act} = n_{idle} + n_{rcv} = \frac{1 - p_{off} + \rho}{(1 + \tau_{off}\mu)\,\rho + 1 - p_{off}} n \tag{6.31}$$

where $\rho = \lambda/\mu$ is the service rate of the system and $\tau_{off} = p_{off}/\mu_{off} = p_{off}T_{off}$ is the average off-time of the peers.

Eventually, the average utilization of the streaming capacity of the active peers η will be calculated as percentage of the overall number of channels on the active peers in the system that are actually busy streaming strips to the other peers:

$$\eta = \frac{\overline{K}}{n_{act}k} \tag{6.32}$$

The portion of traffic served by the peers θ can be calculated as a ratio between the traffic served by the active clients, which expressed as number of channels is the average number of busy channels in the system $\overline{K}$, and the overall traffic in the system. The overall traffic in the system, expressed as number of strips, is the traffic received by the receiving peers, and is obtained by multiplying n_{rcv} by the number of strips per video m. Hence:

$$\theta = \frac{\overline{K}}{n_{rcv}m} = \eta\frac{k}{m}\frac{1 - p_{off} + \rho}{\rho} \tag{6.33}$$

Another parameter that will be considered in the analyses is the quantity of streaming traffic served by the server Φ, expressed as a percentage of the maximum system's throughput, i.e., the traffic that would be generated if all the peers were receiving a stream without failures:

$$\Phi = (1 - \theta)\frac{n_{rcv}}{n}$$

6.2. Model verification and analysis

The verification of the model is performed by comparison of the calculated mathematical results and the results obtained from simulations that simulate the real-time behavior of the proposed system for peer-assisted VoD streaming with non-cooperative peers. The simulation environment is the same as in the previous chapters (Section 5.2), with the difference that the model is upgraded with the new features so that it can support failures of the peers and passing the strips streamed by the failed peers to the servers. The only new parameters that are introduced in the system are the failure probability p_{off} and the average recovery time T_{off}. Therefore, this section will focus on how these parameters influence on the system's performance for different values of the rest of the parameters defined for the system.

Same as in the previous chapter, the system has local communities with size of $n = 200$ peers. The streaming capacity of the peers will most commonly take values $k = 2$ and 5 channels and their storage capacity in most of the cases will be $s = 100$ strips. Each local community is served by an ES that has enough resources to stream every request in the system. The video library consists of $c = 1500$ SD quality videos with play-back rate $r = 2$ Mbps and average duration of $d = 90$ min. The contents are divided into $m = 10$ strips. The videos are previously distributed on the ES and the peers. The ES contains the entire library and the peers contain videos with Uniform-Uniform distribution such that $l = 30\%$ of their storage space is dedicated to the popular contents. The clients are requesting videos according to the ZM distribution with skew factor $\alpha = 0.8$ and shifting constant $q = 10$. The inter-arrival time of the requests is $w = 20$ min.

The main issue of the proposed mathematical model is that for the given values of n, m and k, the size of the system of linear equations is so high that it requires enormous computer power and time to be computed. Therefore, a sampling method for reduction of its size is proposed in Appendix B, used to conduct the computations with an ordinary personal computer within decent time.

In the first simulation scenario, it is analyzed how the probability of failure p_{off} impacts on the utilization of the streaming capacity η when the recovery time is $T_{off} = 150$ min for streaming capacity of $k = 2$ and 5 channels.

The first conclusion from the comparison of the results obtained from the simulations and the mathematical model in Figure 6.5 is that the insignificant differences between the curves validate the correctness of the mathematical model. The figure also shows the dependence of the uplink utilization on the failure probability p_{off} for values of the streaming capacity of $k = 2$ and 5 strips. As the probability of failure p_{off} increases, the utilization of the streaming capacity falls almost linearly in both cases of streaming capacity. This behavior is

quite expected because the increasing failure probability increases the number of failed nodes in stationary state, which implies less peers that participate in the streaming. Therefore, although the peers have available streaming resources, they cannot be optimally used because the probability that a requested strip will be found somewhere on the peers reduces. Like in the simulations in the case with no failures (Figure 5.3), when the peers have lower streaming capacity, the capacity itself gets better utilized. It can be also noted that the results for $p_{off} = 0$ correspond with the results of the system with now failures.

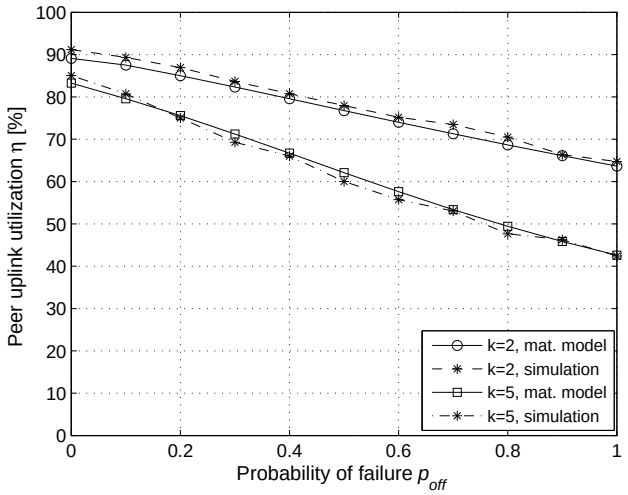


Figure 6.5.: Dependence of the peer uplink utilization η on the probability of failure of the peers p_{off} for streaming capacity $k = 2$ and 5 channels and recovery time $T_{off} =$ 150 min.

The effects of the worsened uplink utilization can be also seen in the percentage of the traffic streamed from the peers θ shown in Figure 6.6. In this figure, the peer-assisted traffic falls from 50% to 20% of the overall traffic within the range of the failure probability p_{off} when the streaming capacity is $k = 5$. For streaming capacity of $k = 2$ strips, the uplink utilization falls more evenly, which is a result of the small participation of the peer-assisted traffic in the overall streaming traffic. If these results are compared to the results obtained for the influence of the number of nodes on the peers' uplink utilization η presented in Figure 5.11, it can be seen that the reduction of the number of active peers n_{act} reduces the uplink utilization in a higher scale. The reason for this steeper curve is that, apart from the reduction of the number of peers that can participate in the streaming, when there are failing peers in the system every end of a streaming on a peer that fails implies interruption of an additional strip that is streamed to the other peers by the failed peer. Figure 6.6 also verifies the accuracy of the calculated results with the mathematical model compared to the results obtained from the simulations.

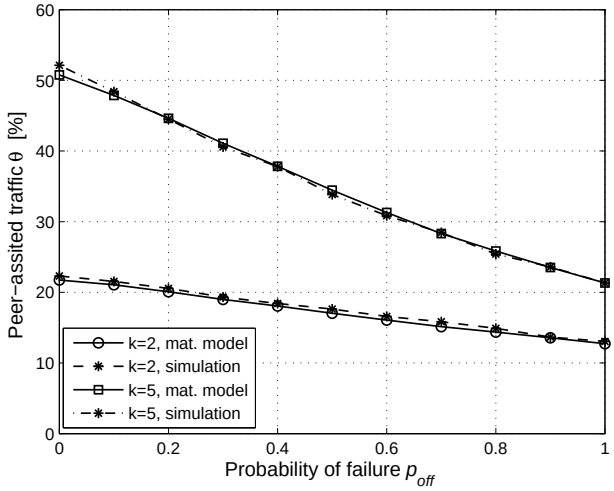


Figure 6.6.: Dependence of the peer-assisted traffic θ on the failure probability p_{off} for streaming capacity $k = 2$ and 5 channels and recovery time $T_{off} = 150$ min.

To show that the failures of the peers introduce more reduction of the peer-assisted traffic than the reduction that would cause only the reduced number of active peers, Figure 6.7 shows a comparison of the overall server traffic in the case when the system has constant size n and there are failures of the peers and when there are no failures, but the size of the system n_{act} changes so that it has the same number of peers as there would be active peers when there are failures. For every value of the failure probability p_{off}, the size of the system $n = n_{act}$ is calculated according to (6.31). The purpose of this simulation scenario is to show the amount of additional traffic that is requested from the server for compensating the failure of the nodes for uninterrupted streaming. Since there is different amount of overall traffic in the two considered cases, the reference point for comparison is chosen to be the streaming capacity of a system with $n = 200$ peers. Therefore, the amount of traffic requested from the server Φ is presented as a percentage of the maximum traffic that could be generated in the system, i.e., the traffic that would be generated when all the n clients would simultaneously receive a stream. The figure shows that in the case when there are no failures, as expected, the reduction of the size of the system causes reduction of the requested traffic from the server Φ. However, when there are failures in the system with the same number of active clients as when there are no failures, there is smaller reduction of the server traffic. The reduction of the traffic comes from the reduced size of the local community due to the failures of the peers, but the more even slope of the curve comes from the additional traffic generated for compensating the strips that are interrupted with the failure of the peers. The additional traffic is actually the difference between the curve obtained for a system with failures and without failures. The value of the failure probability p_{off} has a great impact on the amount of this additional traffic. As it increases, there are more peers that fail and thus there are more streams that

have to be compensated. The figure also shows that the additional traffic increases with the streaming capacity of the peers. There is more additional traffic when there are $k = 5$ channels for streaming than when there are $k = 2$ channels, which is explained by the fact that the higher capacity implies higher number of outgoing streams from the peers, and thus, the failure of one peer requires more compensating streams from the server. The comparison of the simulated values are intentionally omitted in the figure since the percentage of traffic served by the server Φ is calculated from the peer-assisted traffic θ, which was proven to be accurately computed in the Figure 6.6.

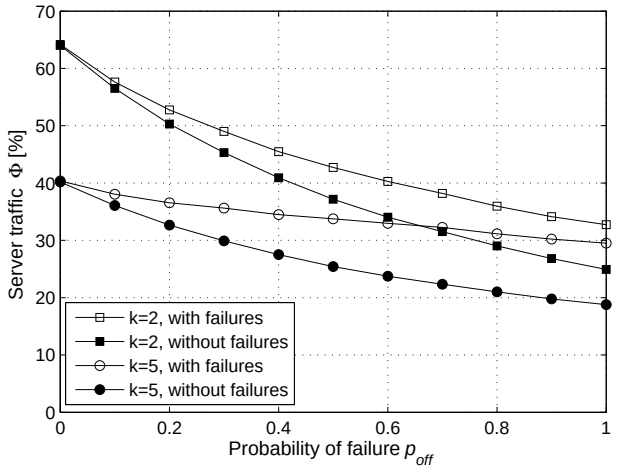


Figure 6.7.: Comparison of the requested traffic from the server Φ of a system with failures and a system without failures with the same number of active peers as the system with failures for recovery time $T_{off} = 150$ min and streaming capacity $k = 2$ and 5 channels.

In the next simulation scenario, the influence of the recovery time of the failed peers T_{off} on the system's performance is shown. For that purpose, the failure probability is set to value $p_{off} = 0.3$ and the recovery time T_{off} is varied in the range from 10 to 510 min. The results presented in Figure 6.8, show that the uplink utilization has a linear dependence on the recovery time T_{off}. The longer it takes a failed peer to recover, the less active peers are there in the system that can serve the receiving peers. Comparing the results obtained for the two different streaming capacities ($k = 2$ and 5 channels), it can be concluded that, although insignificantly, the influence of the recovery time is more emphasized for higher streaming capacities.

The influence of the recovery time T_{off} on the percentage of the traffic served by the peers θ is shown in Figure 6.9. According to the figure, a system with peers with $k = 5$ channels will be more affected by the longer recovery time of the failed peers than a system with $k = 2$ channels. Comparing the influence of the probability of failure p_{off} and the recovery

time T_{off}, it can be concluded that the former has more important role in the performance of the system. The failure probability p_{off} is more important because the traffic that has to be compensated is generated only in the moment of failure, which depends on p_{off}, while T_{off} only determines how long there will be reduced number of active clients in the system.

As in the previous analysis, these figures also show that the results from the mathematical model mostly overlap with the results obtained with simulation. For the sake of clarity, in the future analysis, this comparison will be intentionally omitted.

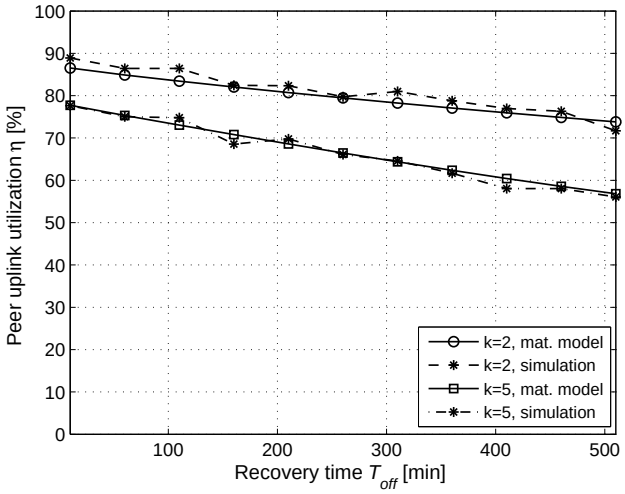


Figure 6.8.: Dependence of the peer uplink utilization η on the recovery time of the peers T_{off} for streaming capacity $k = 2$ and 5 channels and failure probability $p_{off} = 0.3$.

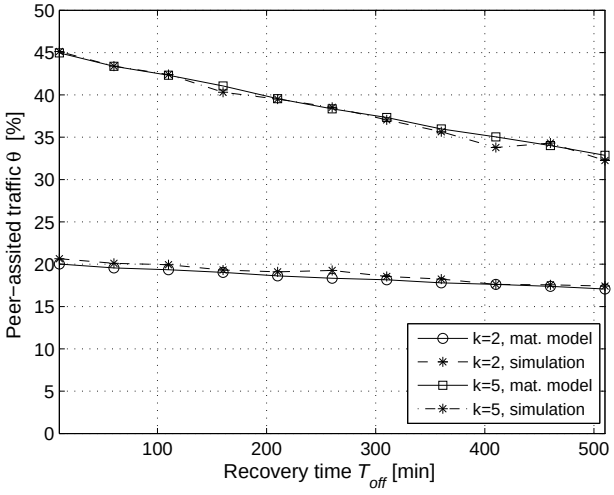


Figure 6.9.: Dependence of the peer-assisted traffic θ on the recovery time T_{off} for streaming capacity $k = 2$ and 5 channels and failure probability $p_{off} = 0.3$.

The additional traffic requested from the server for compensating the streams of the failed peers is shown in Figure 6.10 by comparing the curves of the server traffic in a system with failures with a constant size of the network and the server traffic in a system without failures and size of the network equal to the number of active peers of the system with failures. The maximum value of additional traffic is achieved for the smallest value of the recovery time $T_{off} = 10$ min, and then, it slightly reduces, which can be seen from the curves which are almost parallel with each other and have a tendency to join for very high values of the recovery time T_{off}. This figure justifies the earlier explanation that the additional traffic mainly depends on the failure probability p_{off} which determines the number of compensating streams from the failed peers, while the recovery time T_{off} only determines the general reduction of the traffic because of the reduced number of peers.

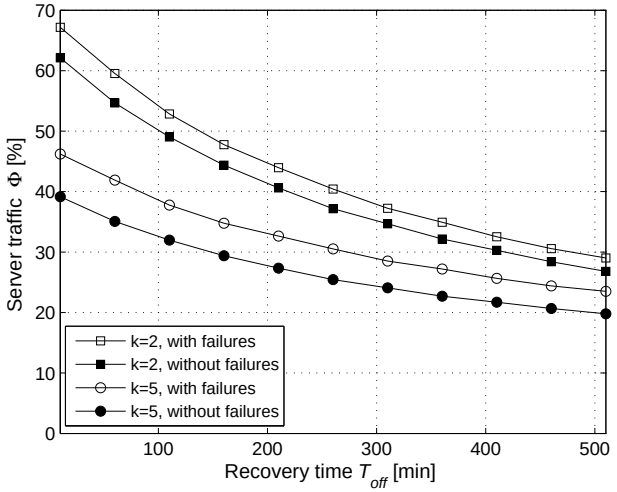


Figure 6.10.: Comparison of the server traffic Φ of a system with failures and a system without failures with the same number of active peers as the system with failures for failure probability $p_{off} = 0.3$ and streaming capacity $k = 2$ and 5 channels.

Figure 6.11 shows the dependence of the server traffic on the failure probability p_{off} and the service rate $\rho = \lambda/\mu$. In order to obtain different values of the service rate ρ, we set the average duration of the streaming sessions to value $d = 100$ min and the inter-arrival time to values $w = 10$, 20, 50 and 100 min. Thus, the service rate obtains values $\rho = \lambda/\mu = d/w = 10$, 5, 2 and 1, respectively. Each value of the service rate ρ is a separate simulation and is presented with a separate curve. The streaming capacity of the peers is $k = 5$ channels, and the recovery time is $T_{off} = 150$ min. The server traffic is presented as a portion of the maximum throughput of the system. When there are no failures in the system ($p_{off} = 0$), the server is most loaded for the highest service rate and slowly reduces as the number of failures increases. The server load decreases because the recovery time T_{off} is longer than the duration of the streaming session, which means that in equilibrium, there will be more

failed peers than receiving peers. The server traffic for compensating the interrupted streams increases but not considerably compared to the other cases because the reduced number of receiving peers implies only a moderate number of strips that have to be compensated. This can be seen from the curve obtained for the case when there are no failures, but there is the same number of peers as the number of active peers. On the contrary, for low service rates ($\rho = 1$), the server traffic increases as the failure probability p_{off} increases. The reason for this is that, initially, there is a large number of idle peers. As the probability of failure increases, the number of receiving peers that fail increases. Consequently, the traffic that has to be compensated by the server increases. Although the system has a tendency to reduce the server traffic with the reduction of the size of the community in the cases with no failures of the peers, for low service rates, the traffic is considerably increased for compensating the streams from the failed peers. This means that in the cases with low service rate, instead of alleviating the servers, the increased number of failing peers adds more load on the servers.

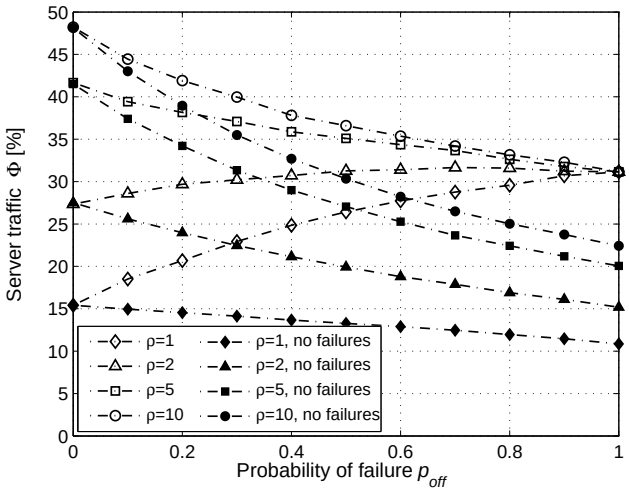


Figure 6.11.: Dependence of the server traffic Φ on the probability of failure p_{off} and the system's service rate ρ for $k = 5$ channels and recovery time $T_{off} = 150$ min.

Another important observation from Figure 6.11 is that, as the probability for failure p_{off} increases, all the curves converge to one point which is reached for $p_{off} = 1$. No matter what the service rate ρ is, for the case when all the peers fail with certainty, the server has to serve the same amount of traffic. This phenomenon can be justified with the fact that when $p_{off} = 1$, the number of idle peers is $n_{idle} = 0$, i.e., a peer is either streaming or it is failed. The number of receiving and failed peers is independent on the inter-arrival rate, which can also be seen if $p_{off} = 1$ is substituted in (6.31). From the two different behaviors of the system, i.e., the tendency of the server traffic to decrease for some values of the service rate and to increase for others, it is natural to expect that for some value of the service rate ρ_x

the server traffic will be constant, independent on the failure probability. The value of this service rate ρ_x is an important parameter for the system since for any rate higher than ρ_x the server traffic will decrease with the increment of the probability, and for the lower service rates the traffic will increase. Since for this specific case $\Phi(p_{off} = 0) = \Phi(p_{off} = 1)$, the value of ρ_x can be found from the curve of the dependence of the server traffic Φ on the service rate ρ obtained for $p_{off} = 0$ by locating the point which has value equal to $\Phi(p_{off} = 1)$. The value of the service rate in that point is the required critical value of the service rate ρ_x.

For a given service rate, the position of the joint point will depend on the value of the recovery time T_{off}. As the value of T_{off} decreases, the point moves upwards. This can be shown in Figure 6.12, which gives the dependence of the server traffic in the joining point on the recovery time T_{off} for value of the duration of the video sessions $d = 50$, 100 and 150 min. If Figure 6.12 is taken into consideration, the dependence in Figure 6.11 for different values of T_{off} can be visualized by joining the initial points for $p_{off} = 0$ with a point obtained for a specific value of T_{off}.

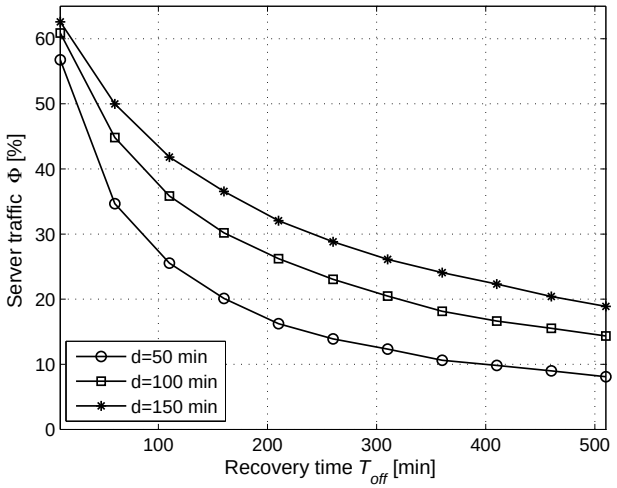


Figure 6.12.: Dependence of the server traffic Φ in the joining point on the recovery time T_{off} for duration of the session $d = 50$, 100 and 150 min.

Figure 6.13 gives the dependence of the server traffic Φ on the peers' storage capacity s for streaming capacity $k = 5$ channels and recovery time $T_{off} = 150$ min. The server traffic has highest values for small storage capacities because, although the peers have available channels for streaming, they are not fully used as a result of the small number of strips they host. Therefore, all the requests for the strips that are not stored in the peers have to be streamed by the server. As the storage capacity increases, the availability of the contents increases, and the server traffic reduces. The figure shows that for storage capacities higher than $s = 100$ strips, the curves that describe the dependence of the server traffic on the failure probability

have the same shape and are parallel to each other, which implies that the system reacts in nearly the same manner to the failures in all the cases of storage capacity, but with a different scale.

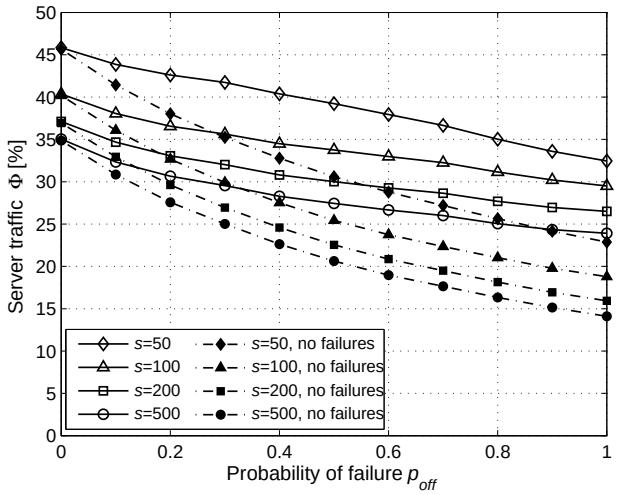


Figure 6.13.: Dependence of server traffic Φ on the probability of failure p_{off} and the peers' storage capacity c for $k = 5$ channels and recovery time $T_{off} = 150$ min.

The comparison of the server traffic Φ with failures with the traffic generated in a network with the same size of active peers, but with no failures, shows that the additional traffic requested from the servers for compensating the interrupted streams is almost the same in all the cases for different storage capacities s and same probability of failure p_{off}. This can be seen from the equal distance between the curves obtained for a case with failures and those obtained for a case without failures for all considered values of the storage capacity s.

Figure 6.14 shows the same dependence as in Figure 6.13 with the difference that the recovery time is $T_{off} = 50$ min. The shorter recovery time T_{off} requires more additional traffic from the server, which causes the overall traffic requested from the server Φ to increase with the increment of the failure probability p_{off}, although there is a smaller number of receiving peers. In the same way as in the previous case, the additional traffic requested from the servers has almost the same value for all the cases of streaming capacities for a given value of the failure probability p_{off}.

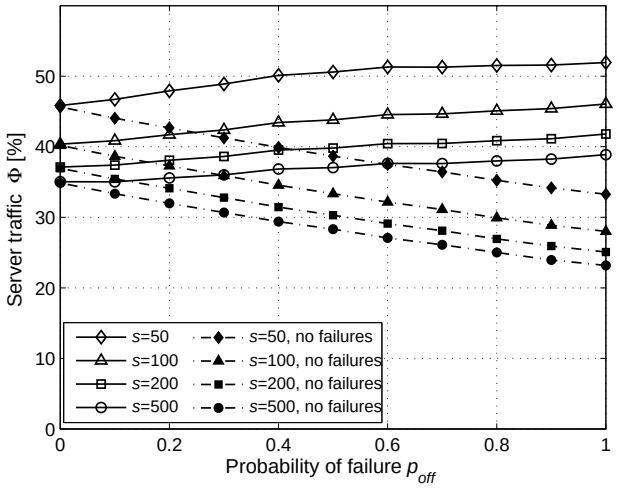


Figure 6.14.: Dependence of server traffic Φ on the probability of failure p_{off} and the peers' storage capacity c for $k = 5$ channels and recovery time $T_{off} = 50$ min.

The next simulation scenario shows the dependence of the server traffic Φ on the distribution of the contents in the peers obtained by changing the percentage of the storage capacity dedicated to the popular contents l for storage capacity $s = 100$ strips and recovery time $T_{off} = 150$ min. The distribution of both the popular and not popular contents is uniform, but with a different portion of the storage space dedicated for each of the two groups of contents. Figure 6.15 shows that the server is most loaded when the peers store only the not popular contents, i.e., when the portion of the storage space dedicated to the popular contents is $l = 0\%$. Dedicating only $l = 10\%$ of the storage to the popular contents will significantly reduce the traffic originating from the server. Furthermore, the figure shows that dedicating more space to the popular contents will not remarkably contribute to the reduction of the server traffic because the curve obtained for $l = 20\%$ slightly differs from the one obtained for complete dedication of the storage to the popular contents ($l = 100\%$). As far as the additional traffic for compensating the interrupted strips is concerned, it can be concluded that the key factor is the failure probability p_{off}, while the distribution has only minor effect.

The similar dependence of the server traffic Φ on the portion of the storage space dedicated to the popular contents l can be observed in Figure 6.16, where the recovery time is $T_{off} = 50$ min. The difference is that instead of the tendency to decrease, the server traffic increases as the probability of failure p_{off} increases.

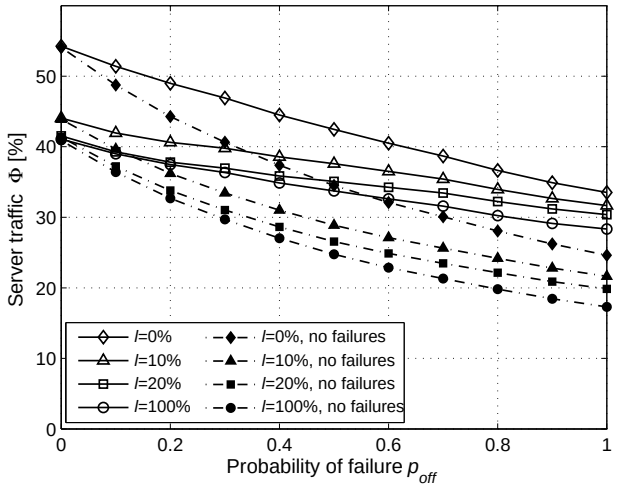


Figure 6.15.: Dependence of the server traffic Φ on the failure probability p_{off} and the portion of storage dedicated to popular contents l for streaming capacity $k = 5$ channels and recovery time T_{off} =150 min.

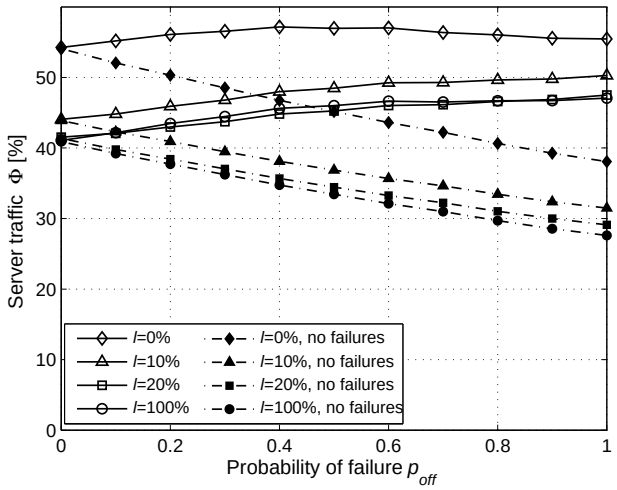


Figure 6.16.: Dependence of the server traffic Φ on the probability of failure p_{off} and the portion of storage dedicated to popular contents l for streaming capacity $k = 5$ channels and recovery time T_{off} =50 min.

The influence of the distribution scheme on the server traffic Φ in conditions of failures for streaming capacity $k = 5$ channels, storage capacity $s = 100$ strips and recovery time T_{off} =150 min is shown in Figure 6.17. It can be concluded that when the contents are distributed according to the Uniform-Uniform scheme there is only a minor increment in the server traffic compared to the other two schemes.

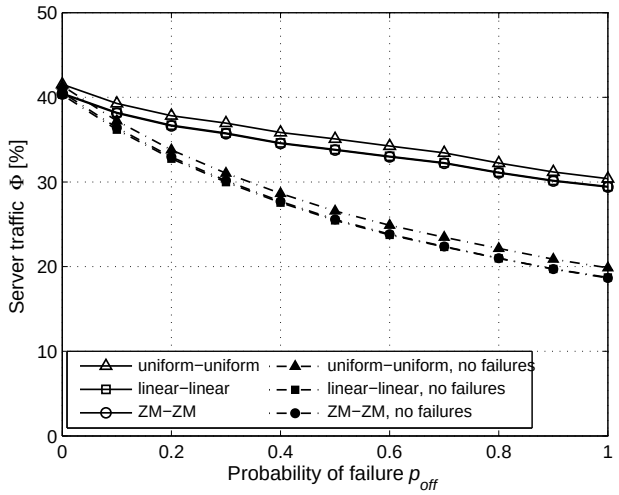


Figure 6.17.: Dependence of the server traffic Φ on the probability of failure p_{off} and the distribution scheme for streaming capacity $k = 5$ channels, storage capacity $s = 100$ strips and recovery time T_{off} =150 min.

6.3. Conclusions

This chapter proposed a stochastic mathematical model for peer-assisted VoD streaming in managed networks with failures of the peers. Although it is based on the system from the previous chapter, the failures of the peers introduce more states in the system which largely increases its complexity.

The simulations showed that the mathematical model is an accurate representative of the described system since the results obtained from the calculations are almost identical to the simulation results. Throughout the chapter, the mathematical model was used as a tool to conduct various analyses on how the failures influence on the system's performance. The results showed that the duration of the streaming sessions, the inter-arrival time and the recovery time of the failed have a crucial role in the overall performance. One of the findings with a significant importance is the fact that with the increment of the failure probability, the traffic streamed from the servers converges to one point for any value of the inter-arrival rate of the request. However, in the cases when the service rate of the system is higher than the threshold service rate, although there is additional traffic for compensating the interrupted streams, the overall traffic of the server decreases with the increasing failure probability of the peers. In the opposite case, however, the failures of the peers have tendency to increase the overall streaming traffic from the servers. This fact could be a serious issue for the VoD provider because the servers are loaded with more traffic than it is planned for conditions with cooperative peers. The storage capacity of the peers is also important for the amount of

served traffic in the cases of failures, but this parameter only determines the initial value of the dependence curve, which follows the same shape as in the other cases of storage capacity. The same conclusion can be taken for the portion of storage space dedicated to the popular contents since it has a minor influence on the shape of the curve of the dependence on the server traffic for small values, but for the rest of the values it only determines the elevation of the curve.

The mathematical model proved to be a powerful and precise tool for estimating the behavior of the system for various parameters depending on the level of cooperativeness of the clients. The model could be used for planning the networks and predicting their performances for a large variety of parameters, which could save precious time and resources of the providers of VoD service in the privately managed networks.

Chapter 7

Conclusions and Future Work

7.1. Conclusions

The work presented in this thesis is an effort to contribute to the solution of the issue of high traffic demands of the VoD service in the managed networks. It initially proposes, in Chapter 3, an architecture for distribution of VoD contents in the private networks with the objective of optimally distributing the traffic among the serves so that the VoD service is provided with a minimal cost for the operator. The solution is obtained by the implementation of a redistribution algorithm that positions the contents in the network according to the behavior of the clients, the popularity of the contents and the state of the streaming servers. It also implements a redirection strategy which tends to keep the servers well balanced. The simulation results of the proposed model proved that the algorithm is highly responsive to the traffic imbalance in the network which is a result of the change of the popularity of the contents or the change of the intensity of the requests of the clients. The algorithm achieves the goal to direct most of the traffic closer to the clients and to reduce the price that the operator has to pay for streaming the requested videos. It also improves the QoS because it eliminates the cases of denial of service when the servers are overloaded because of sudden changes of popularity or increase of the frequency of requests.

The proposed algorithm gives the system autonomy to react in an intelligent way because, whenever it is necessary, it brings decisions on how to distribute the contents and how to redirect the requests of the clients so that it is optimally used. After the execution of the algorithm it achieves an instantaneous change of the state of the servers by transferring the streams of the most popular contents to the new locations.

In Chapter 3, the work also proposes an estimation method used to estimate the average traffic demand for each content based on the overall number of requests for the contents obtained

from the SS server. The advantage of the method is that it saves computing power on the streaming servers necessary for taking account of the history of the requests and calculating the demand based on that data. The estimation method also reduces the management traffic for sending messages with the demand of the contents from the streaming server to the ACM server. Instead, this task is done by the ACM which calculates the demand based on the access data obtained by the SS server.

The negative aspect of the algorithm is that it brings extra traffic in the core of the network for delivery of the contents to the streaming servers. However, the additional traffic has no influence on the QoS received by the clients because the system provides a dedicated server that delivers replicas of the contents that have to change their original location on the streaming servers. Moreover, the algorithm takes into consideration the cost for delivery of the contents in the process of deciding whether it is feasible for a content to be moved to a new location, i.e., whether the movement of a content to a new location would bring more gain to the system due to its vicinity to the clients or it would only increase the cost for delivery from the CR server to a new sever closer to the clients. Despite all the savings that the algorithm tends to make, the additional traffic is an inevitable cost that has to be paid for optimal distribution of the traffic among the streaming servers and improved QoS. Additionally, the traffic for delivery of the contents in the core of the network is reduced due to the implementation of the multicast because it uses only one stream from the CR server to distribute the content to multiple servers. In order to prevent flooding of the network with delivery traffic, the system implements delayed distribution of some of the contents which consists in assigning resources for these contents but delivering them to the servers upon request from the clients.

Constrained by the limited resources of the network and following the example of the implementation of the P2P concept for sharing contents, Chapter 4 proposes a solution that includes the peers in the streaming process by taking advantage of their streaming and storage capacity. In practice, the users subscribed to the VoD service are a part of an IPTV service, and therefore they possess an STB which can be controlled by the operator. The control over the STB of the clients is used as an advantage for increasing the reliability of the peers by reserving part of their storage and streaming capacity, so that they can be used for streaming. The contents are divided into strips that can be simultaneously streamed in order to enable streaming from the peers although their uplink rate is smaller than the playback rate of the videos. Unlike the most common solutions that aim to store the popular contents in the peers, so that the traffic demanded from the server is minimized, the proposed model dedicates a significant part of the storage on the peers for hosting the not so popular contents. The aim of such a distribution is localizing most of the traffic in the periphery of the network, takes into consideration the optimal distribution of the contents on the servers

obtained from the execution of the redistribution algorithm. The contents are distributed according to mixed popularity-based distributions which divide the contents into popular and not popular contents. The mixed distributions are obtained by assigning a different type of distribution probability function to the popular and not popular contents. The mixed distributions dedicate only a small portion of the storage capacity to the popular contents and the rest of the storage is used for storing the not popular contents. The results showed that this helps to keep the servers at the edge of the network busy serving the popular contents, and, at the same time, it alleviates the servers that are deeper in the core of the network, which usually store the not popular contents.

The analysis made on the bases of the simulation results for various distribution schemes, showed that the mixed distributions significantly improve the QoS for small streaming capacity of the peers. For higher streaming capacities, the mixed distribution schemes are the optimal solution because they keep a balance of the advantages and disadvantages of the distributions of only popular or not popular contents. The popular contents increase the utilization of the links but keep the core servers busy, leave the servers at the edge unused and keep the cost of the streaming traffic high. On the contrary, the distribution of only the not popular contents keeps the edge servers busy, reduces the cost to a minimum but poorly utilizes the streaming resources of the peers.

Another important contribution of the proposed distribution schemes is that they increase the scalability of the system, which is very important for the networks that are constantly growing because of the expansion of the VoD service. The reduction of the traffic in the core servers frees streaming resources of the servers for serving higher number of clients. The analysis showed that the mixed distribution schemes enable growth of the network without high additional installation costs in the core of the network. The only necessary change is increasing the capacity of the edge servers. On the contrary, the distribution of the popular contents requires constant upgrade of the resources of the core servers and their interconnection links, which is significantly more costly than adding new servers at the edge of the network required by the mixed distributions.

Although the distribution of the contents is an important factor for the benefits of the peer-assisted streaming, there is still a large variety of system parameters that are of no less importance. The analysis of the influence of every single parameter on the system's performance requires many simulations which could be time-consuming and resource-demanding, especially because of the necessity to present the average value of multiple simulations with the same parameters. Motivated by this requirement, a significant part of this work is dedicated to developing a precise stochastic model of the system for peer-assisted VoD streaming with cooperative peers, unique in the literature, which is one of the most genuine contributions of the thesis. In the stochastic model in Chapter 5, the system is presented as a closed

network of queues with finite population. The queues in this network are the peers, which are presented as a service facility for streaming limited number of strips, and the server, which is representative of all the servers and has theoretically unlimited resources because it can serve any request that cannot be served by the peers. In the process of serving the peers, each request for a video is treated as independent requests for all the strips that compose the video. Therefore, instead of the peers requesting entire videos, the customers of the network of queues are "partial" peers that request only a strip of a video. The network of queues in stationary state is completely defied with a set of states that define the number of the costumers in every queue of the network. The probability of each state is obtained from a system of linear equations used to calculate the average number of busy channels of the peers that take part in the peer-assisted streaming. This value is used for calculating the portion of the overall traffic served by the peers, which is considered as one of the parameters for estimating the contribution of the peer-assisted streaming.

Since in the managed networks the size of a local community is in the range of few hundreds, presenting each peer as a separate queue gives a system with enormous number of states that are very complex to be handled. Therefore, the model introduces such a simplification of the model, so that it manages to significantly reduce the number of states without losing the generality of the model. The simplification consists in presenting all the peers as one representative peer with the cumulative capacity of all the peers without ignoring the constraint that one peer within that community cannot stream more strips than its capacity. This is achieved by using a novel method for calculating the probability that a strip will be served by a peer, given the community size, streaming and storage capacity of a single peer, the distribution scheme and popularity of the contents. The novelty of the method is the use of the powerful product property of the generating functions. Although this method facilitates and accelerates the calculation of the probability that a strip will be served by the peer, it is still very resource-demanding for computation, especially for higher values of the community size, streaming capacity and number of strips. Therefore, in Appendix A, the work presents a numerical extrapolation method based on the values obtained from the generating functions, which reduces the calculation to a table look-up operation with calculation time independent on any parameter. This method not only reduces the computation time for the stochastic models of the systems considered in this work, but also contributes to the global solution of the problem of finding the probability of distribution of z items among t objects, where each object has capacity to store k items.

Because of the specific nature of the stochastic model to include a wide set of states that describe the system, the calculation of the probabilities of each of the states is very time-consuming process. Therefore, a method for sampling the state diagram of the system for cooperative streaming is proposed in Appendix B. The method consists in joining several

neighbor states into one and modifying the coefficients of the probability flows. Consequently, the size of the system is reduced several times. The reduced model gives results that are infinitesimally different from the system with original size of the states.

A great contribution of the mathematical model of the system with cooperative peers is that it successfully describes a system that includes a complex process of handling the requests of the clients, treated as multiple requests of different strips which, depending on the availability of the peers and their current states, might be served by the peers or by the servers. The achieved precision of modeling the system for cooperative peer-assisted VoD streaming is proven by comparison of the results obtained from simulations, which present the behavior of a real system, and the results obtained from the mathematical model with identical parameters as in the simulations. The comparison of the results for different cases of a wide range of parameters of the system confirms the success of the mathematical model to accurately describe the behavior of the system in a closed analytical form in each of the cases.

The mathematical model for cooperative peer-assisted VoD streaming includes a large variety of configurable input parameters that describe the system like, the size of the local community of peers, the storage and streaming capacity of the peers, the duration of the videos, the inter-arrival rate of the requests, the size of the video library and the distribution of the contents on the peers. The output of the algorithm is the utilization of the peers and the portion of traffic streamed by the peers and the server, accordingly. The model can be of a great benefit for the VoD service providers because it helps to estimate the values of the output parameters for given input parameters. This benefit is very important for optimal planning of the networks without the necessity of simulations.

In this work, the mathematical model was used for an analysis that shows how some of the input parameters influence on the utilization of the streaming capacity and the portion of the peer-assisted traffic. From the analysis, it is concluded that the utilization of the links of the peers decreases as their streaming capacity increases, which also results with limited increment of the peer-assisted traffic. This finding is important for deciding to what level the increment of the streaming capacity can be beneficial because, although the operator invests in increasing the uplink capacity, there might not be further increment in the peer-assisted traffic. This behavior indicates that the storage capacity of the peers is another parameter of significant importance for the efficiency of the peer-assisted streaming as it defines the availability of the contents on the peers. The analysis showed that increasing the storage capacity will also not always bring the expected improvements in the system because the peer-assisted traffic is limited by the streaming capacity, despite the increased availability of the contents. Hence, these findings can help the operator to estimate the optimal value of the storage capacity for a given streaming capacity, i.e., the value of the storage capacity upon which further investments in increasing the storage capacity will not be profitable regarding

the obtained increment in the peer-assisted traffic.

As far as the distribution scheme is concerned, the findings indicate that the type of mixed distribution has an influence on the performances of the system in the ranges of small storage capacities. This influence is even more emphasized for higher streaming capacities. An important factor in the amount of peer-assisted traffic is the percentage of storage dedicated to the popular contents, especially for higher streaming capacities. The results indicate that dedicating only a small portion of the storage capacity to the popular content gives significant increment of the peer-assisted traffic, but, once this value reaches the maximum, further increments of the portion dedicated to the popular contents will either have no effect on the peer-assisted traffic or it will have negative effect.

The analyses also show that the size of the local community contributes to the improvement of the system's performance because of the fact that more peers also offer more storage and streaming resources. The size is especially important when the peers have higher streaming capacity.

The size of the video library can also have an effect on the portion of the traffic streamed by the peers, especially when the peers have small capacity for storing the video contents. The increased size of the library reduces the availability of the contents, and therefore, it worsens the performances of the system. An important conclusion from the analysis is that the size of the video library has almost no influence for higher storage capacities of the peers.

Guided by the importance of the mathematical model for the analysis of the managed networks for peer-assisted streaming of VoD contents, the work presented in this thesis goes one step further in making the proposed model more realistic. Although the operator has the control over the STBs while the clients are watching the videos, the clients have the last word in the decision of whether to cooperate in the streaming or not. When their session ends they can simply turn off their STB and exclude their resources from the system. Therefore, the failures of the peers are an additional feature that is used to extend the mathematical model for peer-assisted VoD streaming, in the model proposed in Chapter 6. This feature increases the complexity of the model because a failure of a receiving peer means interruption of all the streams that originate from the failed peer. In order to provide uninterrupted viewing experience, the system handles this situation by assigning the streaming servers to deliver the remaining portions of the interrupted streams. The new feature is also included in the stochastic model by extending the network of queues with the addition of another queue that receives all the peers that fail after the end of their video session. When the clients recover, they leave the queue and join the system. The new queue increases the number of states that describe the system, but it also increases the number of states that can be reached from one of the defined states. The process of handling of the outgoing streams of the failed peers also

increases the complexity of the probability flows in the probability state diagram.

The increased size of the state diagram of the system for peer-assisted streaming with non-cooperative peers proves to be a serious issue for calculation of the probabilities of the states. As a solution, in Appendix B, this work also proposes a sampling method, specific to the new state diagram, which reduces the size of the system and makes the calculations possible.

Although computationally more extensive, by means of comparison, the mathematical model for non-cooperative peer-assisted VoD streaming proves to be a precise representative of the extended model that includes the failures of the peers.

The stochastic model for peer-assisted VoD streaming with non-cooperative peers is used in this work to conduct analyses on how the failure probability, the recovery time of the peers and the rest of the system parameters influence the performances of the system. Unlike the system with cooperative peers, the traffic that is required from the servers not only depends on the number of peers that request contents, but it also depends on the number of peers that fail. Although it is expected that the failures of the peers will require additional traffic from the servers for compensating the interrupted streams, one of the important questions that answers the mathematical model is how will the additional traffic affect the overall traffic required from the servers, i.e., whether it will reduce because of the reduced number of active peers, or it will increase because of the increased number of streams that have to be compensated. The findings that come out from the analysis indicate that if the inter-arrival time of the requests is such that the service rate is higher than a threshold value, the system has tendency to request less traffic from the servers when the probability of failures increases. In contrast, for lower values of the service rate, the system has a tendency to increase the traffic demanded from the servers. The work also proposes a method for determining the value of the threshold rate based on the results from the analysis obtained by the mathematical model. The operators can benefit from this conclusion because according to the behavior of the clients, they can estimate whether the performance of the peer-assisted streaming can deteriorate because of the extensive amounts of traffic generated for compensating for the streams interrupted with the failures.

The results from the analyses show that the storage capacity, the portion of the storage dedicated to the popular contents and the type of mixed distribution scheme are important for defining the shape of the dependency curve, but the various values of these parameters only determine the elevation of the curve and have hardly any influence on its shape.

Although the analyses are conducted for streaming of SD quality videos, a great contribution of the mathematical model is that it can be universally used for analyses of any quality rate of the videos. This property comes from the fact that the mathematical model does not depend on the streaming capacity of the peers and the playback rate of the videos expressed

as bits per second, but it depends on these values expressed as a number of channels, i.e., strips. Thus, for analyses of the streaming of HD quality videos in a system with exactly the same parameters as the analyses conducted throughout the thesis, the only change that has to be made is to increase the number of strips that compose the video because the increased playback rate requires higher number of strips.

7.2. Future work

The findings obtained in this work open a wide field of further investigation which could contribute to further reduce the traffic in the core of the network or to extend the current models by introducing new features for modeling more complex situations.

In that direction, the algorithm could be extended in a way that it will take into consideration the peers in the distribution of the contents. The algorithm would base the calculation on the current requests of the peers and the state of the servers, but also on the available streaming and storage capacity that offer the peers. Thus, apart from issuing commands directed to the streaming servers to reserve storage for the new contents or to delete some of the contents, the ACM server would further include commands that would direct the edge servers to push strips of the contents in the peers. The extension of the model would also include a strategy implemented by the edge servers which would indicate the clients which strips of the videos that they are watching to store in their STBs in a way that the availability of all the strips of different videos is maximized.

A feature that would considerably improve the algorithm is the capability of learning and predicting the behavior of the clients in the future. It is known that the customers have a tendency to watch TV in certain parts of the day when the level of the streaming traffic reaches the peak value. Therefore, learning the watching habits of the clients would contribute to undertaking actions for redistribution of the contents before the necessity for such actions appears.

The performance of the system of peer-assisted streaming can be considerably improved if the concept of adaptive streaming is implemented in the streaming and delivery process. Depending on the congestion in the network and on the occupancy of the streaming resources of the peers and the servers, the streaming can be adjusted to different rates. This would add complexity to the process of handling the requests during the entire service time, as well as the delivery of the contents from the CR server to the streaming servers.

One of the most challenging directions of investigation is the further extension of the stochastic model of peer-assisted VoD streaming. Since the model for non-cooperative streaming itself is an extension of the model of cooperative streaming, it will be taken as a reference model for

further improvements. One of the improvements for the current model for non-cooperative streaming is developing new numerical methods that would furthermore decrease the number of states without significant deviations in the accuracy of the obtained results, which would enable the modeling of systems with bigger local communities and with higher number of channels. The new method would use the fact that many of the states have probability that is close to 0 and have no influence in the final results. The basic idea is to identify these states and to group them in a single state. The number of states that would be joined into one would depend on their probability.

Following the trends of development of the mobile devices, which nowadays are all-in one personal devices, and the trends of increasing the capacity of the wireless links, adding mobility in the model would open an interesting field of investigation that could be beneficial for the operators to further develop the VoD service. The model would include clients that constantly enter or leave the local communities that belong to a common base-station. Despite the advance of the technologies, the energy source of the mobile devices is still an issue, and therefore, the limited battery life would be yet another parameter that would have a central role in such a mobile system.

Unlike the current version of the model where all servers were generalized as one server with infinite resources, a more advanced approach in a future extension of the model is treating the servers separately, as a network of servers with specific streaming and storage resources. This extension would be used to calculate the portion of the server traffic that would be served by each of these servers depending on the distribution of the contents, both on the peers and on the servers. The calculation of the traffic originating from each server would make a ground for including the traffic cost in the list of output parameters of the mathematical model. Since the servers have limited streaming capacity, the denial of service and the process of re-requesting the same contents would also make the system more realistic.

Another challenging extension of the model is enabling inter-community streaming. This feature would increase the availability of the peers and the utilization of the uplink capacity which is beneficial for reducing the traffic requested from the servers. Nevertheless, it would generate high amounts of inter-community traffic in the core of the network and would also require additional computational power from the index servers because they would have to keep up-to date the availability of the contents in each of the local communities. Therefore, an extension of the mathematical model would help to calculate whether the cost of the inter-community traffic would be higher than the cost saved for streaming the contents from the servers, i.e., it would answer the question whether the inter-community peer-assisted streaming is profitable and under which conditions.

Defining incentive methods for the clients to cooperate in the peer-assisted process even after

the end of their video session, so that they gain certain savings in the price that they pay for the service every time when they leave the STB available for peer-assisted streaming could be a good idea for increasing the number of active peers in the network. Although this would reduce the profit of the provider at first glance, in long terms, it could be beneficial because the traffic originating from the servers will be decrease and so will the traffic cost.

The energy efficiency is another interesting and currently popular topic that could be included in the future work for the distribution of VoD contents. The storage of every byte of video on the servers, the reading of the videos from the memory for streaming, the processing of the packets by the network elements and their physical transfer through the network require certain amount of energy. Therefore, including the energy costs in the design of the redistribution algorithms will not only provide optimal solution from QoS point of view, but will also provide an energy efficient solution which will be beneficial both for the operator and the environment.

7.3. Epilogue

The work presented in this thesis is a result of the effort to solve the problem of distribution of VoD contents in the privately managed networks. The main issue with the distribution of these contents is that they are traffic-demanding and therefore, are a serious burden for the network and the streaming servers. This topic has been a motivation of an extensive research work in the past few years, which has resulted with different solutions recognized through scientific publications in international conferences and journals. The first effort for reducing the traffic in the core of the managed network is the architecture of hierarchical structure of streaming servers which uses an algorithm for optimal placement of the contents in the servers according to the users' behavior (Gramatikov *et al.*, 2011). Since the algorithm only redistributes the same traffic among the servers in a way that the most popular contents are streamed from the servers closest to the clients, the next step in this work is including the peers in the streaming process so that they alleviate the servers. This system for peer-assisted streaming of VoD contents focuses on the distribution schemes of the contents, putting an accent on the not popular contents (Gramatikov *et al.*, 2012a) and the cost reduction that is achieved by storing these contents in the peers (Gramatikov *et al.*, 2012b). In order to facilitate the analysis of the influence of various network parameters, including the distribution schemes of the contents on the peers, this work proposes mathematical models of both cooperative (Gramatikov *et al.*, 2013) and non-cooperative (Gramatikov & Jaureguizar, 2014) streaming. Besides, the work proposes numerical methods that considerably accelerate the computation of the results, making these models applicable and useful tools for estimating the performance of the system under given conditions.

The overall work in this thesis complies with the settled objectives to propose architectures for VoD streaming that will minimize the traffic requested from the servers and improve the QoS which are the main issues of the traffic-demanding VoD service. As this service has a growing popularity and a tendency to be a dominant source of traffic in the future managed networks, this work offers a great contribution to the global scientific efforts to offer the most of the service with the minimum cost of the network.

Appendix A

Numerical Methods for Accelerating the Mathematical Calculations

Determining the probability of serving a request for a strip by the peer $P_{p2p}(i)$ in (5.25) and $P_{p2p}(i,l)$ in (6.22) is the key step in the process of defining the mathematical models for peer-assisted streaming in Chapter 5 and Chapter 6. This step is done by defining a range R that depends on the number of available channels on the peers in the system, noted as z, for every state of the system. The range includes all the possible combinations of nodes that can form the number of available channels defined with the state of the system under the constraints that the streaming capacity is k and the number of nodes is n. The analysis that follows will be equal for both the considered models for peer-assisted streaming (cooperative and non-cooperative peers), with only difference in the definition of the number of available channels z. For the case of cooperative peer-assisted streaming, the number of available channels is obtained by subtracting the number of busy channels on the peers i from the overall number of channels on the peers kn, i.e., $z = kn - i$. In the case of non-cooperative peer-assisted streaming, the number of failed channels l is additionally subtracted, thus obtaining $z = kn - i - l$.

The probability $w(t,z)$ of each value t of the range R is determined by using the product property of the generating functions. Although this method accelerates the calculation of the probability $w(t,z)$, it still requires a significant computational power for a system with large number of clients in the local community n and large number of streaming channels on the peers k. In order to further accelerate the calculation of this probability, a numerical extrapolation method is proposed, which determines the probability $w(t,z)$ based on the

results obtained for smaller values of n and k.

The key step that leads to the extrapolation method is the comparison of the shape curves of the probability w obtained for various values of n and k. Figure A.1 shows the values of $w(t,z)$ obtained for various values of available channels z for $k=5$ channels and $n=200$ and 500 peers. For each value of z there is one bell-shaped curve which defines the probability $w(t,z)$ for every value $t \in R$. The figure shows the curves obtained for values of z with step 50 available channels. In the case when $n=200$ peers, the first bell-shaped curve shows that if the system is in a state with $z=50$ available channels, from all the possibilities of distributing these channels among $t \in R = \left[\left\lceil z/k \right\rceil, \min(n,z)\right] = [10,50]$ peers, it is most probable that they will be distributed on $\mu = 27$ peers. As the value of t gets further from the value 27, the probability $w(t,z)$ reduces and reaches value 0. It can also be seen that the bell-shaped curves change the width as z increases, but also they start to get irregular shape as the most probable number of nodes μ approaches the upper limit of the range R, which is equal to the size of the community, i.e., $t=200$ peers. For $z=nk=1000$ available channels, which is the maximum value of available channels in the system, the bell converges into the Kronecker delta function $\delta_{t,200}$, defined as:

$$\delta_{i,j} = \begin{cases} 1 & i=j \\ 0 & i \neq j \end{cases} \tag{A.1}$$

An important observation from the figure is that for $n=500$ peers, the bell-shaped curves overlap with the curves for $n=200$ peers for small values of z. However, as the curves obtained for $n=200$ peers start to get irregular shape ($z \geq 350$ available channels), the bell-shaped curves obtained for $n=500$ peers continue with the regular shape and the expanding tendency of their width. With the further movement of the most probable number of peers μ of the bells towards the maximum value of the range $t=500$, their shape becomes irregular, and similarly like in the previous case, it reaches the shape of the Kronecker delta function $\delta_{t,500}$ for the maximum value of $z=nk=2500$ available channels.

Figure A.1 shows the calculated values for steps of 50 available channels, however, for accurate modeling of the peer-assisted streaming of VoD contents, the step should have value 1. This means that for obtaining the complete set of curves for a given system with n peers with capacity k, there have to be calculated $nk+1$ different curves (one for each number of available channels in the system). The drawbacks of the calculation of the values of the probability $w(t,z)$ according to (5.24) is that it requires high computation power as the values of n and k increase since R obtains wider span of values and the calculation of the expression (5.22) becomes a time-consuming and resource-demanding process. Therefore, the main objective is to use the shape of the curves obtained by means of the generating functions

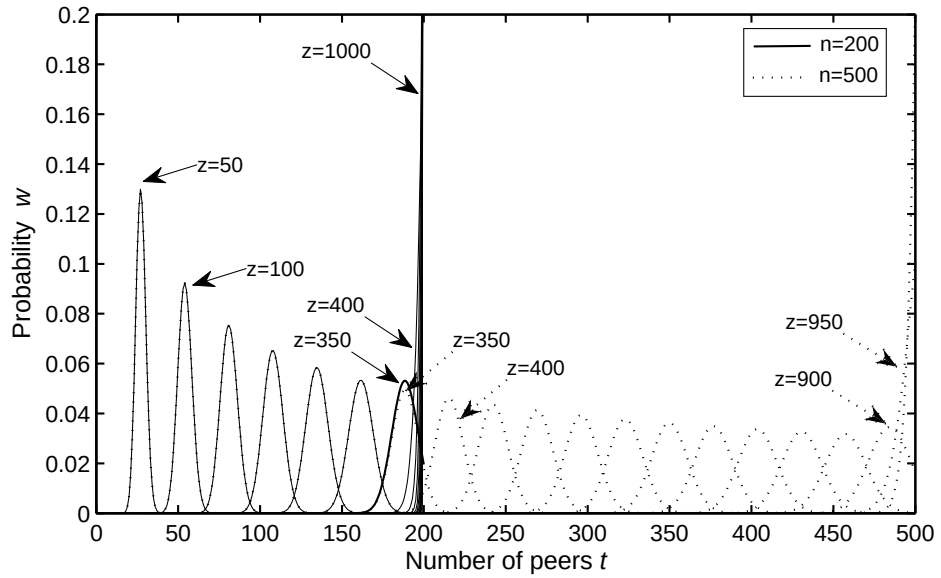


Figure A.1.: Dependence of the probability w on the values of the range R for streaming capacity $k = 5$ channels and community size $n = 200$ and 500 nodes.

(despite the long time necessary to obtain them) and relate it to a known function that can be easily calculated for any values of n, k and z. This relation can be obtained if the shape of the bell-shaped curves obtained in Figure A.1 is compared to the shape of the Gaussian (Normal) distribution function, which is also known to be bell-shaped.

Let us consider the curve obtained for $n = 200$ peers, $k = 5$ channels and $z = 200$ available channels. Figure A.2 shows the set of values of the probability $w(t, z)$ obtained by means of the generating functions and the shape of a Gaussian function with mean value μ and standard deviation σ obtained by applying an interpolation-based fitting function on the same set of values of $w(t, z)$. It can be seen that the values of the discrete probability function $w(t, z)$ are accurately represented with a continuous Gaussian function with μ=107.84 and standard deviation $\sigma = 6.0989$. This result shows that instead of the time-consuming computation of the coefficients of the generating functions, the values of the probability $w(t, z)$ can be obtained by sampling the Gaussian function in the integer points, provided that the mean value μ and the standard deviation σ are known. Therefore, the next important step is determining the dependence of the mean value μ and standard deviation σ on the size of the community n and the streaming capacity k.

Applying the fitting function on the rest of the bell-shaped curves defined by the set of values of $w(t, z)$ for the case when the community size is $n = 200$ peers, and the streaming capacity of the peers is $k = 5$ channels, gives the values of the mean value μ and standard deviation σ shown in Figure A.3 and Figure A.4.

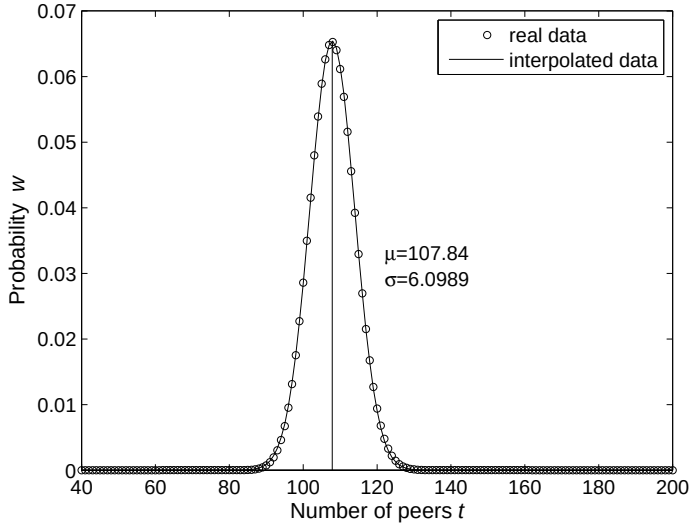


Figure A.2.: Comparison of real data and data of a Gaussian fitting function for the probability w for $n = 200$ peers, $k = 5$ channels and $z = 200$ available channels.

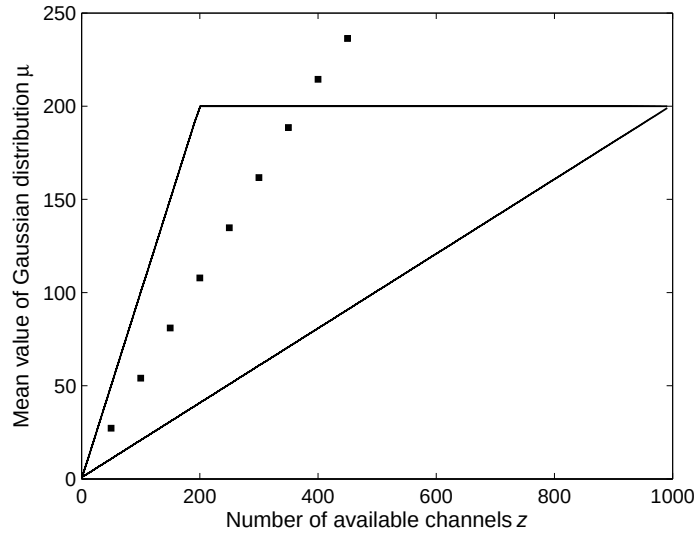


Figure A.3.: Dependence of the mean value of the Gaussian distribution on the number of available channels z for $n = 200$ peers and $k = 5$ channels.

From Figure A.3, it can be concluded that the mean value μ is linearly dependent on the number of available channels z. As z increases, the mean value μ moves in right direction, which can also be seen in Figure A.1. The figure does not present the values of μ for $z > 450$ available channels. The reason for this can be found in the shape of the curves for $w(t, z)$ for higher values of the number of available channels z. These figures converge from a bell-shaped Gauss function to a Kronecker delta function, and therefore, the fitting function for

Gaussian distribution cannot accurately determine the parameters μ and σ. Figure A.3 also presents the range of possible values of number of nodes t that can have z available channels as a closed set of values within a triangular surface. In the figure, it can be seen that the imaginary line obtained by joining the set of points for μ in direction from $z = 0$ to infinity, leaves the surface of possible values for higher values of z. The critical value of z when the imaginary line leaves the range surface is noted as z_o and is dependent on the size of the community n and the streaming capacity k. If the curves that have mean values that lie outside of the surface are localized in Figure A.1, it can be seen that these are the ones that have irregular shape.

Figure A.4 shows the dependence of the standard deviation σ on the same set of values for the number of available channels z as in Figure A.3. Unlike the mean value μ, the standard deviation σ has non-linear dependence. The increasing of the standard deviation for higher values of z explains the tendency of increasing the width of the bell-shaped curves in Figure A.1. It is important to note that for the last two values of z ($z = 400$ and $z = 450$ available channels) that generate mean values outside of the triangular surface, the values of the standard deviation σ do not follow the growing pattern of the rest of the values, which is also justified by the irregular shape of these curves in Figure A.1. Therefore, in the calculations in this work, for every value of $z > z_o$ the shape of the curve of the probability $w(t, z)$ is approximated to a Kronecker delta function.

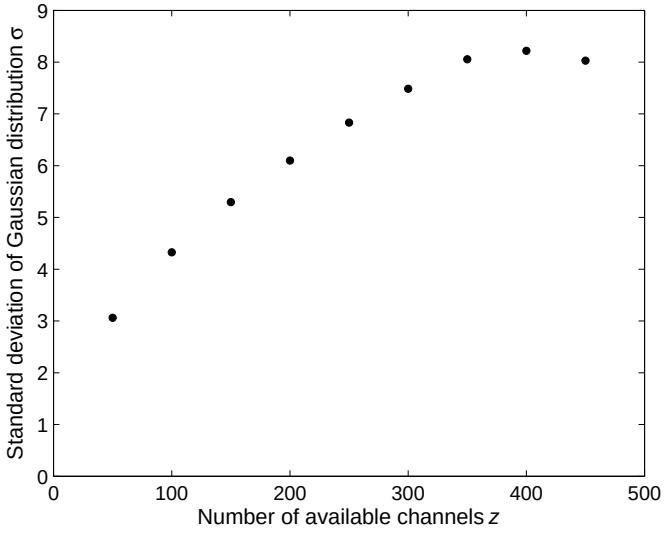


Figure A.4.: Dependence of the standard deviation of the Gaussian distribution on the number of available channels z for $n = 200$ peers and $k = 5$ channels.

The dependence of the Gaussian function parameters on the parameters n, k and z can be better understood if the same interpolation fitting function is applied to the regular bell-

shaped curves of $w(t,z)$, i.e., those curves obtained for $z > z_o$, for various values of these parameters. Figure A.5 shows the dependence of the mean value μ on the number of available channels z for communities with sizes $n = 200$, 300 and 500 peers, and streaming capacities $k = 2$ and 5 channels. From the figure, it can be concluded that the mean value, if exists, lies in the same position for a given value of z and k, no matter what is the value of n. This means that if the equation of the line defined for one streaming capacity is known, the values of the mean value of the Gaussian function can be easily determined for any size of the community. Since the two lines obtained for $k = 2$ and 5 channels begin at the point (0,0), it is obvious to conclude that the equation of the dependence has the following form:

$$\mu(k,n,z) = \begin{cases} \text{round}\,(c(k)z) & z \leq z_o(n,k) \\ n & z > z_o(n,k) \end{cases} \tag{A.2}$$

where $z_o(n,k) = \text{round}(n/c(k))$ determines the point where the line leaves the triangular surface that determines the range of possible values that μ can take. In the case when $z > z_o$, the mean value of the probability function $w(t,z)$ is n, however it does not refer to a Gaussian function with $\mu = n$, but to a Kronecker delta function $\delta_{t,n}$.

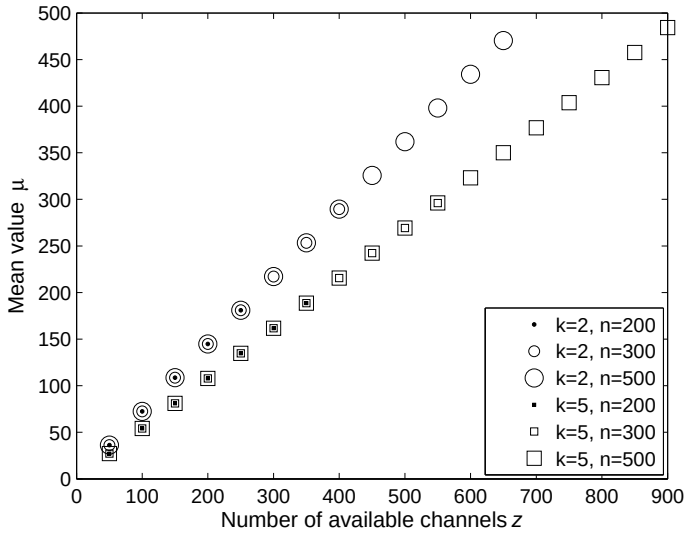


Figure A.5.: Dependence of the mean value μ of a Gaussian function on the community size n, streaming capacity k and number of available channels z.

The coefficient $c(k)$ depends only on the streaming capacity k. Using a linear fitting function on the data sets obtained for different values of k, by means of the generic functions, the dependence of the coefficient of the lines c on the streaming capacity k is obtained as shown in Figure A.6. The exact values of the coefficient $c(k)$ are also presented in Table A.1. The

figure shows that, as the number of channels increases, the coefficient c converges to value 0.5. For values of $k > 10$ channels it can be assumed that the coefficient c has the limit value without significant errors in the calculations. The great significance of this finding is that the coefficient c is independent on the size of the community n and is accurately determined for any value of k.

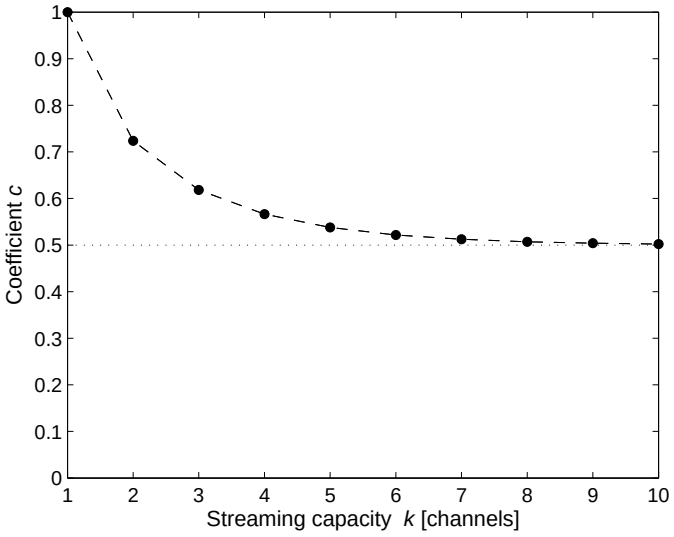


Figure A.6.: Dependence of the value of the linear coefficient c on the streaming capacity k.

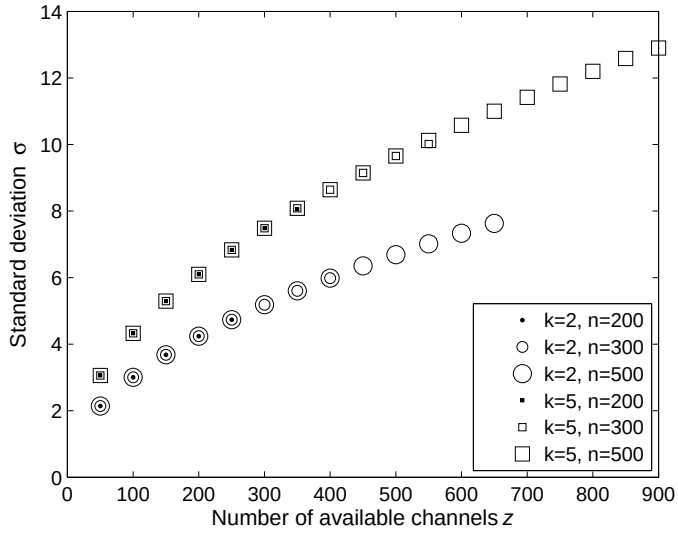


Figure A.7.: Dependence of the standard deviation σ of a Gaussian function on the community size n, streaming capacity k and number of available channels z.

Figure A.7 shows the dependence of the standard deviation σ on the number of available channels z for communities with sizes $n = 200$, 300 and 500 peers, and streaming capacities

$k = 2$ and 5 channels. Similarly like in the representation of the mean value μ, it can be concluded that the standard deviation, if exists, for a given streaming capacity k is independent on the size of the community. From the shape of the curves that interconnect the points for a single streaming capacity k, it can be concluded that they have exponential dependence, i.e., they can be presented in the following form:

$$\sigma(k,n,z) = \begin{cases} a(k)z^{b(k)} & z \leq z_o(n,k) \\ 0 & z > z_o(n,k) \end{cases} \tag{A.3}$$

After applying a fitting function on the set of values obtained by means of the generic functions for various values of the streaming capacity k, the dependency of the coefficients a and b is obtained as shown in Figure A.8 and Table A.1. It can be seen that the coefficient b is independent on the streaming capacity k because it has a constant value which can be approximated to 0.5. The only exception from this value is obtained for $k = 1$ because when there is only one streaming channel on the peers, for any number of available channels z, there is only one value in the range R. Therefore, the probability function $w(t,z)$ has the shape of a Kronecker delta function $\delta_{t,n}$. The value of the coefficient a increases with the increment of the number of streaming channels k and asymptotically approaches the value 0.5. This is another important result for the calculation of the probability $w(t,z)$ for values of the streaming capacity k higher than the values presented in the figure ($k > 10$).

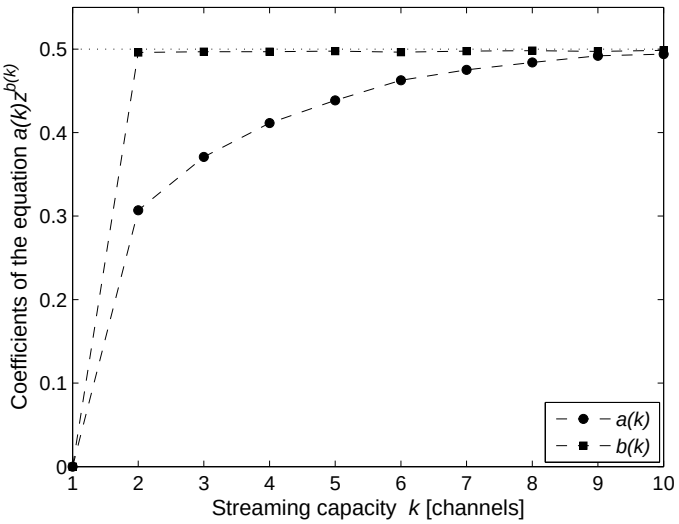


Figure A.8.: Dependence of the value of the linear coefficients a and b of the equation $a(k)z^{b(k)}$ on the streaming capacity k.

Table A.1.: Values of the coefficients for obtaining the dependence curves of μ and σ.

k	$c(k)$	$a(k)$	$b(k)$
1	1.0000	0	0
2	0.7236	0.3069	0.4960
3	0.6184	0.3708	0.4969
4	0.5663	0.4114	0.4968
5	0.5379	0.4385	0.4975
6	0.5217	0.4626	0.4964
7	0.5124	0.4751	0.4976
8	0.5071	0.4840	0.4982
9	0.5040	0.4920	0.4972
10	0.5021	0.4940	0.4985

The proposed numerical method significantly accelerates the calculation of the probability function w which is further used for calculation of the probability that a request for a strip will be served by a peer. It is important to emphasize that the calculation time with the proposed method is independent on the size of the local community n, the streaming capacity of the peers k and the number of available channels z. The calculation of any function $w(t, z)$ is done by a look-up of the coefficients in Table A.1 for a given value of k. The dependencies of the mean value μ and the standard deviation σ are determined based on the value of n according to (A.2) and (A.3). Then, the value of z is substituted in the equations, and the exact shape of the Gaussian function is obtained. Substituting every value of the range R obtained for the given values of n, k and z in the Gaussian function will give the probability of distribution of z available channels among every possible number of peers defined in R.

Let us consider an example of determining the shape of $w(t, z)$. In a system with $n = 250$ peers with streaming capacity of $k = 8$ channels and a state when the system has $z = 400$ available channels, according to Table A.1, the values of the coefficients c, a and b will be 0.5071, 0.4840 and 0.4982. The critical value of z is $z_o = n/c = 493$, and hence $\mu = \text{round}(0.5071z) = 203$, and $\sigma = 0.484z^{0.4981} = 9.57$. The final expression for the function w will be:

$$w(t, z) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(t-\mu)^2}{2\sigma^2}} = 0.3989 e^{-\frac{(t-203)^2}{183.1698}},\ t \in [50, 200]$$

Substituting the values of the range $R = [50, 200]$ in the expression above will give the exact shape of the probability function $w(t, z)$.

The importance of the proposed method is not only in the extreme acceleration of the calculation of the entire range of the probability function $w(t, z)$, but also in the contribution to the calculation of the global problem of finding the popularity of distribution of z items among t objects with capacity of k items.

Appendix B

Sampling Methods for Size Reduction of the Stochastic Models

The main approach that used in the mathematical models for precisely estimating the usage of the peers' uplink capacity in the streaming of the VoD contents is defining a set of states of the entire system that include the states of each peer in the local community and the states of the servers. These states determine the number of channels, both on the peers and the server, occupied for streaming strips of the videos. Additionally, the system for non-cooperative streaming extends this set of states by including the number of failed channels. In order to avoid the enormous set of states obtained with this approach, the models are based on simplifications which consist in representing all the peers with one peer that has the cumulative streaming and storage capacity of the entire local community. Although this approach significantly reduces the number of states necessary for precise description of the system, yet, the obtained models include huge set of states. The size of this set largely depends on the size of the local community n, the number of strips that compose the videos m and the streaming capacity of the peers k. The main issue of both the mathematical models is that for the higher values of these parameters, which are the values used in real-time simulation scenarios, the size of the set of states is so huge that that systems of linear equations used for determining the probability of each state cannot be computed by a computer with decent processing power that can be provided with the current computer technology. This issue makes the mathematical models unusable for real-sized systems. Therefore, the following sections propose sampling methods for reduction the size of the set of states which accelerate the computation of the utilization of the peers' uplink capacity. These methods significantly reduce the size of the diagrams by joining several states into one state and modifying the values of the arcs between the sate probabilities. The sampling introduces certain errors in the results, however, throughout the text, it will be clarified that this error is minor and

insignificant for the analyses.

B.1. Model for cooperative peer-assisted streaming

The size of the set of states in the mathematical model for the system for cooperative peer-assisted streaming is a serious issue from a computation point of view. This can be concluded from the expression that determines the number of states of the flow of probabilities $\mathcal{D}$ of the system, further referred to as state diagram. Its size is obtained by summing the number of states of each row of the diagram in Figure 5.2. The first row of the diagram has $nm + 1$ states, and each following row has one state less than the previous one. There are $nk + 1$ rows, obtained for each possible number of busy channels on the peers from 0 to nk. By using the expression for the sum of the first n natural numbers $\sum_{i=1}^{n} i = n(n+1)/2$, the expression for the size of the state diagram is obtained as:

$$\begin{aligned} \text{size}(\mathcal{D}) &= \sum_{r=0}^{nk}(nm+1-r) = (nk+1)^2 - \sum_{r=0}^{nk}(r) = \\ &= (kn+1)\left(\left(m-\frac{k}{2}\right)n+1\right) \approx kn^2(m-k/2) \end{aligned} \tag{B.1}$$

For typical values of the system parameters used in the analyses, e.g., $n = 200$ peers, $m = 10$ strips and $k = 5$ channels, the size of the state diagram will be approximately $1.5 \cdot 10^6$ states. This is a considerably high number, especially from a computation point of view. Taking into consideration the fact that the size of the coefficient matrix B is determined by the number of states of the system, its size for this specific case of values would be $1.5 \cdot 10^6 \times 1.5 \cdot 10^6$. For a representation of the values of the matrix with double precision numbers that occupies 8 bytes, the overall coefficient matrix B would occupy approximately 11.3 TB of memory space, which is hardly a feasible size of RAM memory with the current computer technology. Moreover, the multiplication $B^{-1}\mathbf{b}$ in (5.28) would also require high computational power. Consequently, the computation of the probabilities of the states would be a very time-consuming and resource-demanding process.

In order to accelerate the computation and reduce the required resources, the size of the original state diagram $\mathcal{D}$ of the system is reduced by implementation of a sampling method. The sampling consists in forming a new state diagram by choosing each h-th state of the original state diagram shown in Figure 5.2, starting from the top-left state and going in both right and down direction. In the process, the states that are not chosen are omitted, however, the probability flows that leave or arrive at these states are included in the new probability flows that connect the chosen states. The sampling of the original state diagram and the

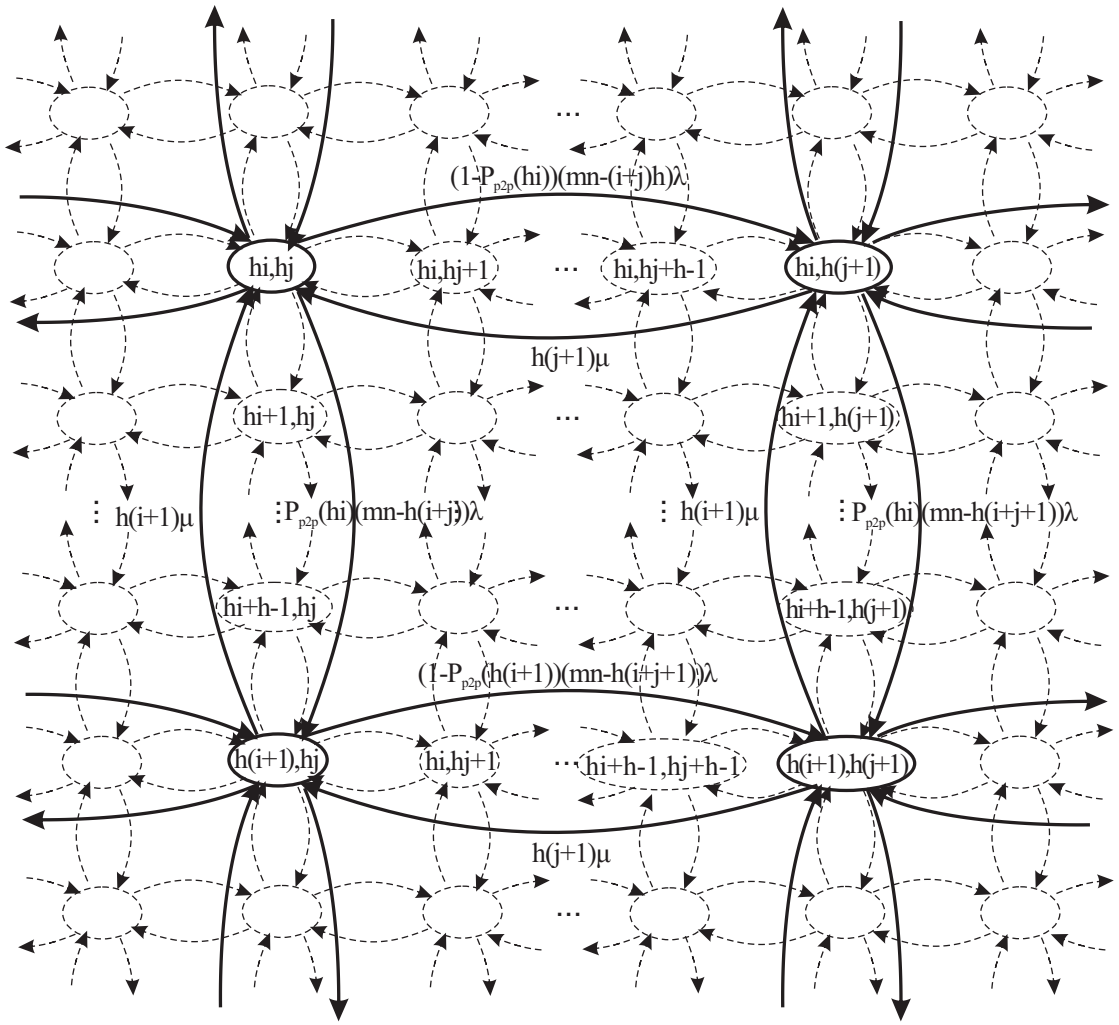


Figure B.1.: Sampling of the original diagram of flow of probabilities of a system with cooperative peers.

connection of the sampled states with the new probability flows is shown in Figure B.1. The new, reduced diagram $\widetilde{\mathcal{D}}$ includes only those states with indexes that are multiple of the sampling step h, provided that the number of channels that can be requested in the system nm and the number of channels on the peers nk are multiples of the sampling step h. The sampling method also includes the right-most and down-most states of the diagram in the case when nm and nk are not divisible by the sampling constant h.

A global picture of the reduced state diagram $\widetilde{\mathcal{D}}$ is shown in Figure B.2. The overall size of the reduced diagram is:

$$\text{size}(\widetilde{\mathcal{D}}) = \frac{(2J - I + 3)(I + 2)}{2} \tag{B.2}$$

where

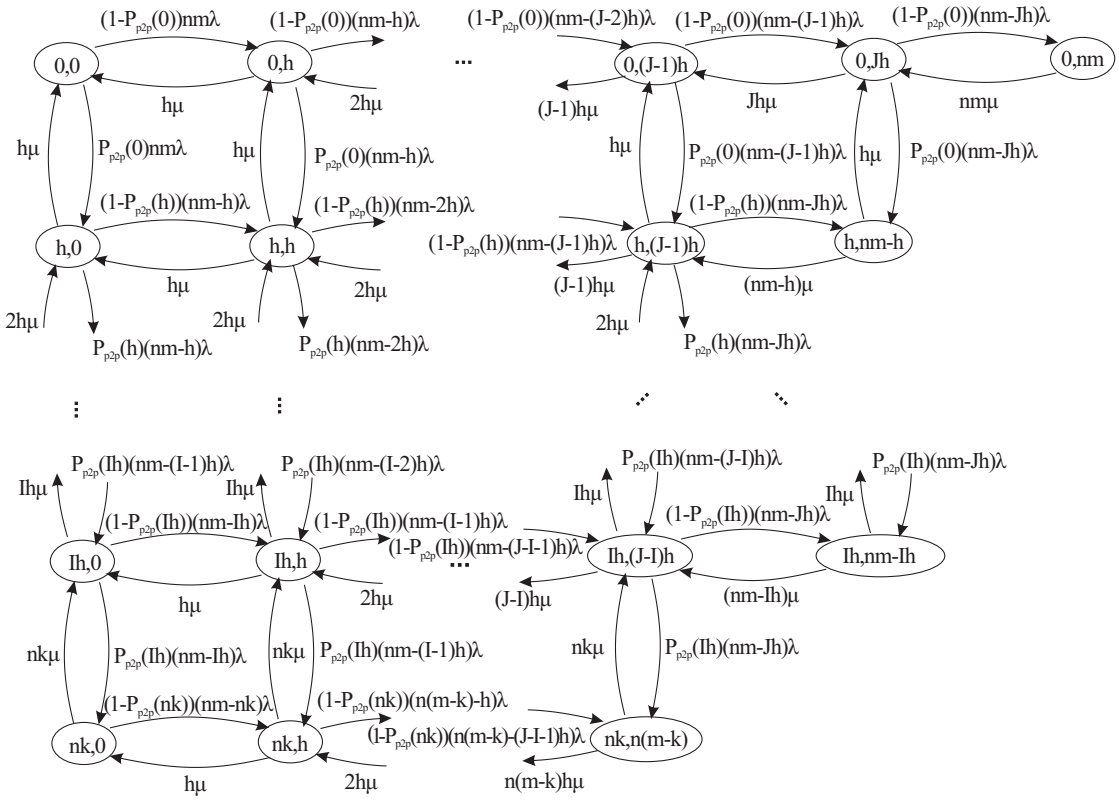


Figure B.2.: Reduced diagram of flow of probabilities of a system with cooperative peers.

$$I = \lceil nk/h \rceil - 1 \tag{B.3}$$

and

$$J = \lceil nm/h \rceil - 1 \tag{B.4}$$

In order to present a clearer picture of the sampling process, Figure B.3 shows an example of a small-scale system with $n = 6$ peers, with capacity of $k = 3$ channels, videos divided into $m = 5$ strips and sampling step $h = 7$ states, obtained by substituting the values in the general reduced state diagram shown in Figure B.2. Instead of approximately 300 states obtained from (B.1), by using the sampling method, the new size of the diagram $\widetilde{\mathcal{D}}$ according to (B.2) is only 18 states.

The system presented in the previous example has very small size, which is not of interest for the analysis of real systems. The advantages of the sampling method are more evident in the case of the previously consider system with $n = 200$ peers, $m = 10$ strips and $k = 5$ channels. By choosing a sampling constant with value $h = 20$ states and using it in (B.3) and (B.4), I and J would be 49 and 99, which substituted in (B.2) would give a size of approximately 4000 states. The new reduced diagram has approximately 400 times less states compared to

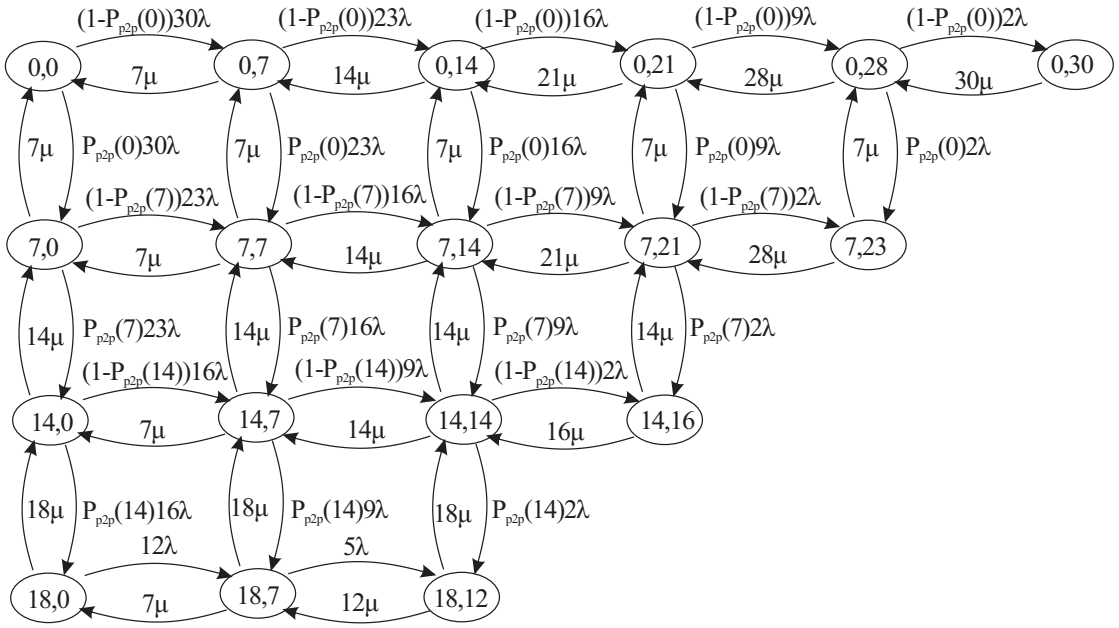


Figure B.3.: Example of a reduced diagram of flow of probabilities of a system with cooperative peers.

the original diagram with size of $1.5 \cdot 10^6$ states. Increasing the value of the sampling step to $h = 50$ states, would give approximately 650 states, which is equivalent of a gain of 2300 times in the size reduction.

Naturally, it is expected that the reduction of the size would introduce errors in the results, the value of which will depend on the size of the sampling step h. For that purpose, Figure B.4 shows the dependence of the relative error δ of the peer-assisted traffic θ for a system with size $n = 200$ peers, $m = 10$ strips and storage capacity $s = 100$ strips for variable values of the streaming capacity k and the sampling step h. The figure shows that, apart from the sampling step h, the relative error largely depends on the streaming capacity of the peers k. A critical value of the streaming capacity is $k = 8$ when the relative error δ reaches the maximum value. However, it can be concluded that up to $h = 50$ states, the relative error is below 1% for almost the entire range, expect for the critical value $k = 8$ channels, when the relative error has slightly bigger value than 1%.

Choosing the value of the sampling step h will depend on the maximum size of the system that the operator can afford for computation of the results and on the maximum relative error it can tolerate. The exact value has to be a compromise between the resources and the correctness of the results. Table B.1 shows an overview of the sizes of the system obtained for various values of the sampling constant h and the streaming capacity k, obtained from B.2. If these results are compared to the relative error in Figure B.4, it can be concluded that choosing a sampling value $h = 20$ considerably reduces the size of the system and at

the same time keeps satisfactory level of error in the results for all values of the streaming capacity k.

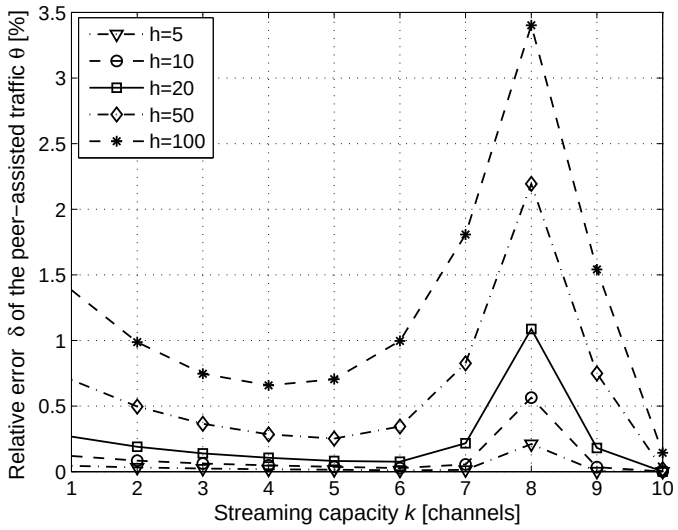


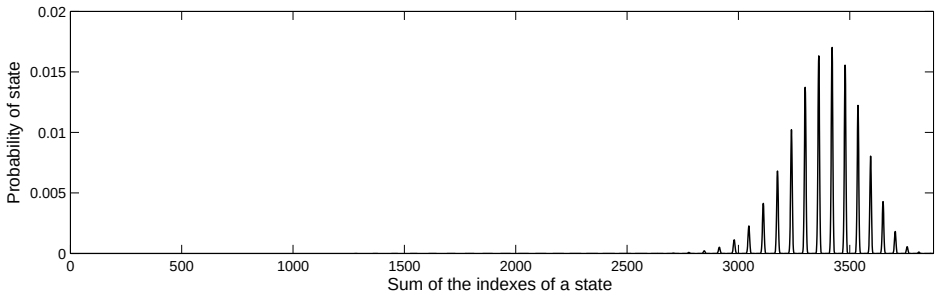
Figure B.4.: Dependence of the relative error δ of the peer-assisted traffic θ in a system with cooperative peers on the streaming capacity of the peers k and the sampling constant h for community with size $n = 200$ peers and number of strips per video $m = 10$.

Table B.1.: Overview of the size of the system with $n = 200$ cooperative peers and $m = 10$ strips for various values of the sampling step h and the streaming capacity k.

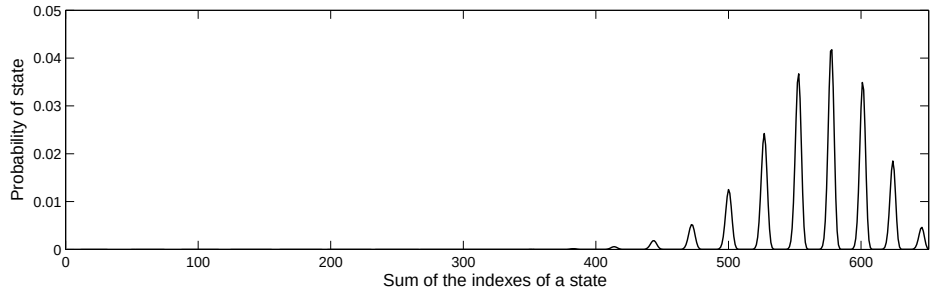
h	Size		
	$k = 2$	$k = 5$	$k = 10$
1	722201	1502501	2003001
2	181101	376251	501501
5	29241	60501	80601
10	7421	15251	20301
20	1911	3876	5151
50	333	651	861
100	95	176	231

One possible solution for a further reduction of the relative error with the same or even smaller size of the system obtained for a certain value of the sampling step h can be proposed if the values of each state probability $p_{i,j}$ shown in Figure B.5 are more thoroughly analyzed. The figure shows two cases of the probabilities of the states of a system with $n = 200$ peers with capacity of $k = 5$ channels and sampling step $h = 20$ and 50 states. The sum of all the values

in both the figures equals 1, however, Figure B.5a has more states and more bell-shaped curves than Figure B.5b. An important conclusion that can be made from the figures is that each local bell-shaped curve represents the state probabilities of one row of the diagram in Figure B.2. The bell-shaped curves in the figures presented from left to right are obtained for the rows of the state diagram when moving from the first row, downwards, i.e., the left curves refer to small values of the index i and the right curves refer to large values of the index i. The number of curves is determined by the sampling step in a vertical direction. The movement within a local curve from left to right refers to increasing the value of the index j. The number of values within a curve is determined by the value of the sampling step in horizontal direction. Therefore, the sampling step in vertical direction h_v can have a different value from the step in the horizontal direction h_h. Since the average uplink utilization η, and hence the peer-assisted traffic θ, are obtained by multiplying the value of the index i with the sum of the probabilities of its corresponding local curve in (5.29), it is more important to have more rows in the diagram. Consequently, the value of the vertical sampling constant h_v can be decreased and the value of the horizontal sampling constant h_h can be increased.



(a) $h = 20$



(b) $h = 50$

Figure B.5.: Probability of the states (i, j) of a system with $n = 200$ peers, $m = 10$ strips and $k = 5$ channels for sampling constant (a) $h = 20$ and (b) $h = 50$.

Figure B.5 also shows that for small values of the index i, the local bell-shaped curves are not visible, which means that these states have probability close to value 0, and therefore, can be ignored without introducing noticeable error in the calculations. Therefore, the vertical sampling constant h_v for the small values of i can be increased in order to join the least

probable states in the first rows into fewer rows. Although these states will be represented by fewer states, the sum of their probabilities will be again close to value 0, and the increasing of the sampling constant would not introduce errors. On the contrary, for the higher values of the index i, the vertical sampling constant can be reduced so that the probabilities of the more popular states in the last rows of the diagram are calculated with more accuracy. The horizontal sampling constant h_h can also have variable values if the local curves are further analyzed in order to determine the positions of the states in the diagram in the popular rows that have probability close to 0. This method and the previously proposed methods are left for a future work since the simple method where $h_h = h_v = h$ gives satisfactory results for the analyses of the system for peer-assisted VoD streaming proposed in Chapter 5.

B.2. Model for non-cooperative peer-assisted streaming

The size of the set of states of the system for non-cooperative peer-assisted streaming is considerably higher than the size of set of states of the previously considered system because the failures of the peers add another dimension of the diagram of the probabilities of the states (further referred to as state diagram). The size of the state diagram in this case can be determined by summing the number of states of each plane in the diagram in Figure 6.4, which corresponds to all the states that have the same number of busy channels on the peers. Each plane represents a local diagram that consists of rows of states that decrease in size by one state. The last row of the local diagram has size of 1 state. In the diagram $\mathcal{D}$ in Figure 6.4, the size of the first row of the local diagram $\mathcal{D}(i)$, which corresponds to the states of i busy channels on the peers, is $mn + 1 - i$. Hence, by using the expression for the sum of the first n natural numbers $\sum_{i=1}^{n} i = n(n+1)/2$, the size of this local diagram $\mathcal{D}(i)$ is obtained as:

$$\begin{aligned} \text{size}(\mathcal{D}(i)) &= \sum_{r=0}^{nm-i} (nm + 1 - i - r) = \\ &= \frac{1}{2}(n^2m^2 + 3nm + 2) + 2i^2 - \frac{3}{2}(nm+1)i \end{aligned} \tag{B.5}$$

There are $nk + 1$ different values that i can take, starting from 0 to nk, which implies that the overall size of the diagram $\mathcal{D}$ is:

$$\begin{aligned} \text{size}(\mathcal{D}) &= \sum_{i=0}^{nk} \text{size}(\mathcal{D}(i)) = \\ &= \frac{1}{2}(n^2m^2 + 3nm + 2)(nk + 1) + \sum_{i=0}^{nk} i^2 - \frac{3}{2}(nm + 1)\sum_{i=0}^{nk} i \end{aligned} \tag{B.6}$$

Using the expression for the sum of the squares of the first n natural numbers $\sum_{i=1}^{n} i^2 = n(n+1)(2n+1)/6$ and the previously mentioned expression for the sum of the first n natural numbers in (B.6), yields:

$$\text{size}(\mathcal{D}) = \frac{1}{12}(nk + 1)\left(n^2(6m^2 + 2k^2 - 9mk) + n(18m - 8k) + 12\right) \tag{B.7}$$

If this equation is used for calculating the size of a typical system considered in this work, e.g., $n = 200$ peers, $m = 10$ strips and $k = 5$ channels, the size of the state diagram $\mathcal{D}$ will be approximately $6.697 \cdot 10^8$ states. Speaking in terms of memory capacity for storing the values after the calculation of the inverse modified coefficient matrix B^{-1} with double precision numbers of 8 bytes, it will be necessary a space of 3.18 EB. A matrix with such an enormous size could not be even stored in the hard disk of the modern personal computers. The large size of the coefficient matrix also speaks about the huge processing power necessary for the computation of the inverse matrix B^{-1} and the multiplication for obtaining the values of the state probabilities defined in (6.24). Therefore, in the calculations of the results, it is applied the same sampling method, previously used in the case of the cooperative streaming. Since the state diagram has three dimensions, the sampling method considers every h-th plane starting from the bottom of the diagram, and for every chosen plane, it takes every h-th state in horizontal and vertical direction, starting from the upper-right corner of the plane (Figure 6.4). Figure B.6 shows the sampling method applied to a portion of the state diagram $\mathcal{D}$ of the system. The reduced diagram $\widetilde{\mathcal{D}}$ is obtained by simply changing the unit that is added or subtracted from the indexes of the central state with the value of the sampling step h.

Using the same definition of I and J from (B.3) and (B.4), and applying the same procedure for obtaining the size of the original diagram $\mathcal{D}$, the size of the reduced diagram $\widetilde{\mathcal{D}}$ is obtained as:

$$\text{size}(\widetilde{\mathcal{D}}) = \frac{1}{12}(I + 2)\left(6J^2 + 24J + 2I^2 - 10I - 6IJ + 24\right) \tag{B.8}$$

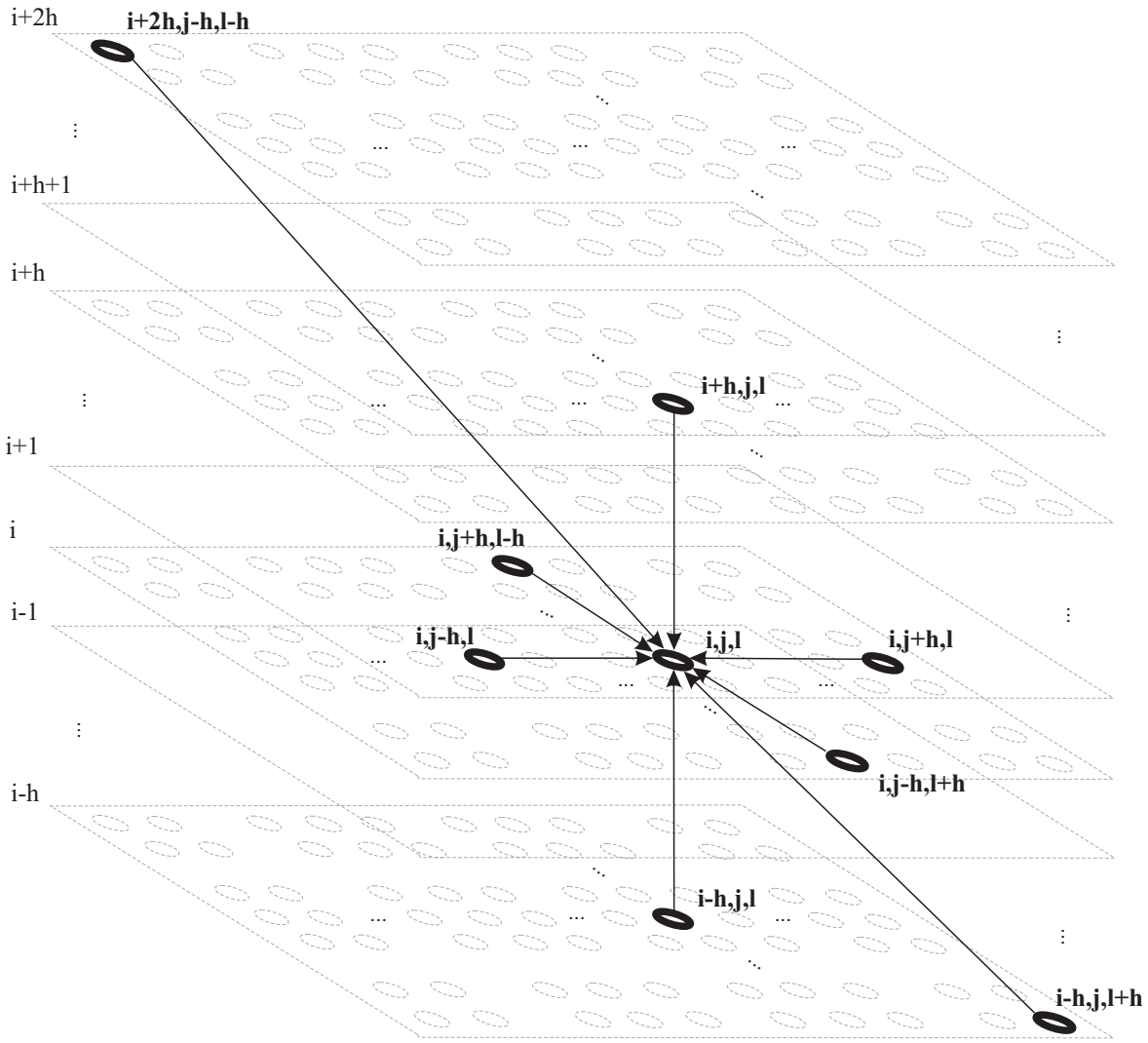


Figure B.6.: Sampling of the original diagram of flow of probabilities for a system with non-cooperative peers.

Substituting the previously defined values of the system with $n = 200$ peers, $m = 10$ strips and $k = 5$ channels with sampling step $h = 20$ states, the size of the system reduces to $1.547 \cdot 10^5$ states. For a higher value of the sampling step, e.g., $h = 50$ states, the size of the system is $1.08 \cdot 10^4$ states. Although the sampling method significantly reduces the size of the system, the reduced diagrams still have a large size. Therefore, the computation of the results is still a time and resource-consuming process. The size of the sampling step h can be further increased for reducing the size of the system, however, it will also increase the error introduced with the sampling. Figure B.7 shows the relative error of the peer-assisted traffic θ introduced in the system as a result of the sampling with steps $h = 50$ and 100 states. Since the size of the obtained diagrams for values of $h < 20$ are impossible to be computed with a single personal computer, the results obtained for $h = 20$ states are taken as a referent

point for comparison of the error introduced with the higher values of the sampling step h. The figure shows the errors obtained for streaming capacity of up to $k = 7$ channels because the computations are difficult to be obtained for the higher values due to the huge size of the system. In the figure, the relative error δ for $k = 1$ channel is higher compared to the error for the higher streaming capacity. However, one should note that when there is only one streaming channel on the peers, the peer-assisted traffic has small values. For this specific case the peer-assisted traffic is only 10% of the overall streaming traffic, so the introduced error will not have big importance on a global scale. The relative error also reaches high values for streaming capacity higher than $k = 7$ strips. This is quite a worrying fact since for the higher streaming capacity, the peer-assisted traffic takes a significant part of the overall traffic. Therefore, the introduced error is too high for the peer-assisted traffic to be taken as a relevant value on a global scale. For a streaming capacity in the range $k \in [2, 6]$ channels, the relative error has acceptable value, especially for the case when the sampling constant is $h = 50$ states which introduces a maximum error of 2%.

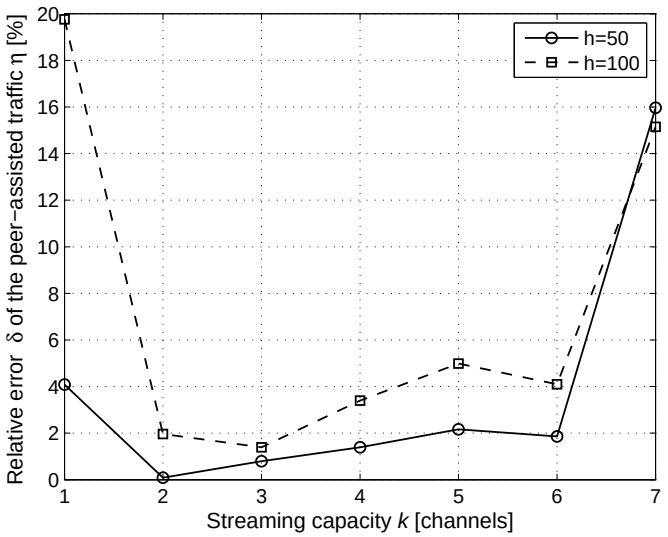
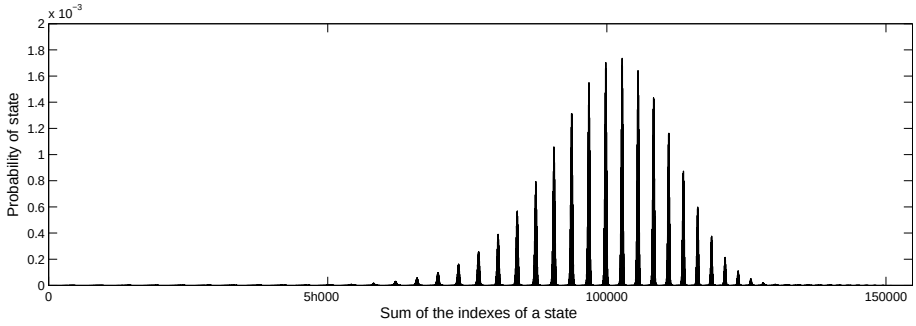


Figure B.7.: Dependence of the relative error δ of the peer-assisted traffic θ in a system with non-cooperative peers on the streaming capacity of the peers k and the sampling constant h for community with size $n = 200$ peers, number of strips per video $m = 10$, failure probability $p_{off} = 0.3$ and recovery time $T_{off} = 150$ min.

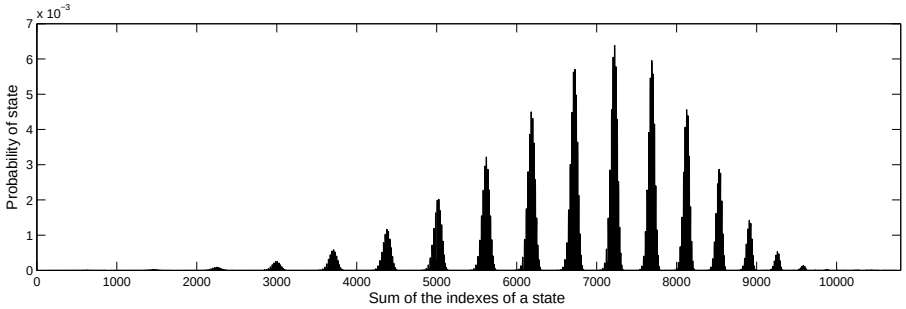
Figure B.7 shows that the mathematical model with high value of the sampling step h gives erroneous results for higher values of the streaming capacity k. The proposed idea for improving the accuracy of the system with cooperative peers by choosing different values of the sampling step could not be applied in the system with cooperative peers. Using different values of the sampling step for sampling the planes, e.g., h_v and another value for sampling the states within the planes h_p is not an acceptable option for further reducing the state of

the diagram because of the specific structure of the diagram. That can be concluded from Figure B.6 which shows that if there are chosen states that are h_p positions from the central state in the plane i, there have to be chosen states from exactly h_p and $2h_p$ planes above and h_p planes below. If $h_p \neq h_v$, then, the required planes above and below would not exist. Therefore, other approaches for reduction of the size of the state diagram and parallelization of the computation should be considered.

One of the approaches that can be implemented for improving the accuracy of the system can be derived by examining the Figure B.8, which shows the value of the probability of each state in the system when the sampling constant is $h = 20$ and 50 states.



(a) $h = 20$ states



(b) $h = 50$ states

Figure B.8.: Probability of the states of a system with non-cooperative peers with $n = 200$ peers, $m = 10$ strips, $k = 5$ channels, failure probability $p_{off} = 0.3$ and recovery time $T_{off} = 150$ min for sampling step (a) $h = 20$ and (b) $h = 50$ states.

In the figure, each local bell-shaped curve represents the probabilities of the states that lie in the same plane of the state diagram. The left-most bell-shaped curve in the figure refers to the bottom plane of the state diagram, and the right-most refers to the top plane of the diagram. Therefore, the plot in Figure B.8a, obtained for $h = 20$, has more bell-shaped curves than the plot in Figure B.8b, obtained for $h = 50$. From Figure B.8a it can be concluded that there is a significant number of states that have probability close to value 0. These states, further referred to as unimportant, could be reduced into a few sates that can be obtained with higher value of the sampling constant, e.g., $h = 50$ states because they

have infinitesimal influence on the values of the probabilities of the other states placed in the middle of the range of states. Reducing the number of these unimportant states enables reducing the size of the system, which gives the possibility to increase the number of the so called important states in the middle of the range that are crucial for obtaining an accurate value of the number of busy channels on the peers. Therefore, in the state diagram, the important states can be chosen with a smaller sampling constant, e.g., $h = 20$ states, which will result with a diagram with high number of planes only of those probabilities that take significant part in the final results. This approach would enable obtaining accurate results with smaller size of the state diagram. The figure shows that there is also a possibility to further localize the important probabilities within a plane (local bell-shaped curve), however, all these issues are left for the future work.

Another trivial, but effective solution is dividing the videos into smaller number of strips m when a model for higher streaming capacity k of the peers is needed. The lower number of strips implies higher streaming rate of each channel, which implies less channels for expressing the streaming capacity of the peers. The reduced sizes of m and k would significantly reduce the overall size of the system which would enable using smaller sampling step in the calculations. An example of this simple solution is a system with $n = 200$ peers where the videos are divided into $m = 10$ strips and their playback rate is $r = 2$ Mbps, which implies that the steaming rate for one strip is $r/m = 200$ kbps. If the streaming capacity of the peers is $k = 8$ strips (1600 kbps), for a sampling step $h = 50$ states, the size of the system according to (B.8) would be size$(\widetilde{\mathcal{D}}) = 12221$ states. Although the reduced size is acceptable for computation, the results proved that the introduced error for $h = 50$ and values of $k > 5$ is unacceptable. Now, if the video is divided into $m = 5$ strips, each strip would have playback rate of 400 kbps and the streaming capacity of the peer would be $k = 4$ channels (1600/400). Dividing the number of strips by 2 imposes dividing the storage capacity of the peers by 2 because, in the model, the streaming capacity is expressed as number of strips that can be stored in a peer. As the size of the strip is increased twice, the number of strips that can be stored in the peer has to be reduced by half. The simulations show that the number of strips of a video m does not influence the obtained results if the streaming capacity k and the storage capacity s are adjusted to the streaming rate of the strip and its size. For these values of n, m and k and sampling step $h = 10$ states, the size of the system would be e size$(\widetilde{\mathcal{D}}) = 1.547 \cdot 10^5$ states. A system whit such number of states can be easily solved in decent time and the obtained results would be with acceptable error. The downside of this approach is that it can be used only for specific values of the streaming capacity, the streaming rate of one streaming channel, the number of strips of a video m and the number of channels per peer k. They have to be such that the playback rate r of the video is m times the streaming capacity of the channel and the streaming capacity of the peer has to be k times the streaming capacity

of the channel.

Because of the computational limitations of the system, this work only considers systems with streaming capacity of the clients k up to 5 channels, obtained by sampling the original state diagram of the system with sampling step $h = 50$. The analysis of systems with higher streaming rates and lower values of the sampling step remains an open field of future work.

Bibliography

Banerjee, S., Bhattacharjee, B. & Kommareddy, C. (2002). Scalable Application Layer Multicast. *SIGCOMM Comput. Commun. Rev.*, 32(4), pp. 205–217.

Bartolini, N., Casalicchio, E. & Tucci, S. (2003). A Walk through Content Delivery Networks. In: *MASCOTS Tutorials*, volume 2965 of *Lecture Notes in Computer Science*, Springer, pp. 1–25.

BitTorrent (2013). BitTorrent. Available at: http://www.bittorrent.com.

Borst, S., Gupta, V. & Walid, A. (2009). Self-organizing Algorithms for Cache Cooperation in Content Distribution Networks. *Bell Labs Technical Journal,*, 14(3), pp. 113–125.

Breslau, L., Cao, P., Fan, L., Phillips, G. & Shenker, S. (1999). Web Caching and Zipf-like Distributions: Evidence and Implications. In: *Proceedings of IEEE INFOCOM*, volume 1, pp. 126–134.

Brosh, E., Agastya, C. & Morales, J. (2009). Serving Niche Video-on-Demand Content in a Managed P2P Environment. *Architecture*, pp. 1–17.

Buyya, R., Pathan, M. & Vakali, A. (2008). *Content Delivery Networks (Lecture Notes Electrical Engineering)*. Springer-Verlag Gmbh, 1st edition.

Carlsson, N., Eager, D.L. & Mahanti, A. (2009). Peer-assisted On-demand Video Streaming with Selfish Peers. In: *Proceedings of Networking*, volume 5550, pp. 586–599.

Castro, M., Druschel, P., Kermarrec, A.M., Nandi, A., Rowstron, A. & Singh, A. (2003). SplitStream: High-Bandwidth Multicast in Cooperative Environments. In: *Proceedings of SOSP*, pp. 298–313.

Cha, M., Rodriguez, P., Moon, S. & Crowcroft, J. (2008). On Next-generation Telco-managed P2P TV architectures. In: *Proceedings of IPTPS*, pp. 1–6.

Chae, Y., Guo, K., Buddhikot, M., Suri, S. & Zegura, E. (2002). Silo, Rainbow, and Caching Token: Schemes for Scalable, Fault Tolerant Stream Caching. *IEEE Journal on Selected Areas in Communications*, 20(7).

Chen, S., Shen, B., Wee, S. & Zhang, X. (2003). Adaptive and Lazy Segmentation Based Proxy Caching for Streaming Media Delivery. In: *Proceedings of NOSSDAV*, ACM, pp. 22–31.

Chen, Y., Huang, Y. & Jana, R. (2007). When is P2P Technology Beneficial for IPTV Services. In: *Proceedings of ACM NOSSDAV*.

Chen, Y.F., Huang, Y., Jana, R., Jiang, H., Rabinovich, M., Rahe, J., Wei, B. & Xiao, Z. (2009). Towards Capacity and Profit Optimization of Video-on-demand Services in a Peer-assisted IPTV Platform. *Multimedia Systems*, 15(1), pp. 19–32.

Chen, Y.F., Jana, R., Stern, D., Wei, B., Yang, M., Sun, H. & Dyaberi, J. (2010). Zebroid: Using IPTV Data to Support STB-Assisted VoD Content Delivery. *Multimedia Systems,*, 16(3), pp. 199–214.

Chesire, M., Wolman, A., Voelker, G.M. & Levy, H.M. (2001). Measurement and Analysis of a Streaming-Media Workload. In: *Proceedings of USITS*, pp. 1–12.

Choe, Y.R., Schuff, D.L., Dyaberi, J.M. & Pai, V.S. (2007). Improving VoD Server Efficiency with Bittorrent. In: *Proceedings of Multimedia*, pp. 117–126.

Cisco Systems (2013). *Cisco Visual Networking Index: Forecast and Methodology, 2012-2017*.

CoDeeN (2013). CoDeeN: A CDN for PlanetLab. Available at: http://codeen.cs.princeton.edu.

CORAL CDN (2013). CORAL CDN. Available at: http://coralcdn.org.

Dana, C., Li, D., Harrison, D. & Chuah, C.n. (2005). BASS: BitTorrent Assisted Streaming System for Video-on-Demand. In: *Proceedings of MMSP*, IEEE, pp. 1–4.

Deshpande, H., Bawa, M. & Garcia-Molina, H. (2001). *Streaming Live Media over a Peer-to-Peer Network*. Technical Report 2001-30, Stanford InfoLab.

Do, T.T., Hua, K.A. & Tantaoui, M.A. (2004). P2VoD: Providing Fault Tolerant Video-on-Demand Streaming in Peer-to-Peer Environment. In: *Proceedings of IEEE ICC*, pp. 1467–1472.

Dukes, J. & Jones, J. (2002). Dynamic RePacking: a Content Replication Policy for Clustered Multimedia Servers. In: *Proceedings of the Microsoft Research Summer Workshop*.

Dyaberi, J.M., Kannan, K. & Pai, V.S. (2010). Storage Optimization for a Peer-to-Peer Video-on-Demand Network. In: *Proceedings of ACM MMSys*, pp. 59–70.

Fallica, B., Lu, Y., Kuipers, F., Kooij, R. & Mieghem, P.V. (2008). On the Quality of Experience of SopCast. In: *Proceedings of NGMAST*, IEEE, pp. 501–506.

Fan, B., Chiu, D.M. & Lui, J. (2006). Stochastic Differential Equation Approach to Model BitTorrent-like P2P Systems. In: *Proceedings of IEEE ICC*, volume 2, pp. 915–920.

Flajolet, P. & Sedgewick, R. (2009). *Analytic Combinatorics*. Cambridge University Press.

Fu, W., Xiao, N. & Lu, X. (2008). A Quantitative Survey on QoS-Aware Replica Placement. In: *Proceedings of GCC*, IEEE Computer Society, pp. 281–286.

Garey, M.R. & Johnson, D.S. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness (Series of Books in the Mathematical Sciences)*. W. H. Freeman, 1st edition.

Ge, Z., Figueiredo, D., Jaiswal, S., Kurose, J. & Towsley, D. (2003). Modeling Peer-Peer File Sharing Systems. In: *Proceedings of IEEE INFOCOM*, volume 3, pp. 2188–2198.

Gopalakrishnan, V., Bhattacharjee, B., Ramakrishnan, K., Jana, R. & Srivastava, D. (2009). CPM: Adaptive Video-on-Demand with Cooperative Peer Assists and Multicast. In: *Proceedings of IEEE INFOCOM*, pp. 91–99.

Gramatikov, S. & Jaureguizar, F. (2014). Stochastic modelling of non-cooperative peer-assisted VoD streaming in managed networks (Under review). *Comput. Netw.*

Gramatikov, S., Jaureguizar, F., Cabrera, J. & Garcia, N. (2011). Content Delivery System for Optimal VoD Streaming. In: *Proceedings of IEEE ConTEL*, IEEE, pp. 487–494.

Gramatikov, S., Jaureguizar, F., Cabrera, J. & García, N. (2012a). Popularity Based Distribution Schemes for P2P Assisted Streaming of VoD Contents. In: *Proceedings of MMEDIA*, pp. 14–19.

Gramatikov, S., Jaureguizar, F., Cabrera, J. & García, N. (2013). Stochastic modelling of peer-assisted VoD streaming in managed networks. *Comput. Netw.*, 57(9), pp. 2058–2074.

Gramatikov, S., Jaureguizar, F., Mishkovski, I., Cabrera, J. & García, N. (2012b). P2P Assisted Streaming for Low Popularity VoD Contents. In: *Proceedings of ICT Innovations*, volume 207, Springer, pp. 23–33.

Guo, L., Chen, S., Ren, S., Chen, X. & Jiang, S. (2004). PROP: A scalable and reliable P2P assisted proxy streaming system. In: *Proceedings of ICDCS*, IEEE, pp. 778–786.

Guo, Y., Suh, K., Kurose, J. & Towsley, D. (2003a). A Peer-to-Peer On-Demand Streaming Service and Its Performance Evaluation. In: *Proceedings of ICME*, IEEE, ISBN 0-7803-7965-9, pp. 649–652.

Guo, Y., Suh, K., Kurose, J. & Towsley, D. (2003b). P2Cast: Peer-to-Peer Patching Scheme for VoD Service. In: *Proceedings of WWW*, ACM, pp. 301–309.

Hei, X., Liang, C., Liang, J., Liu, Y. & Ross, K.W. (2007). A Measurement Study of a Large-Scale P2P IPTV System. IEEE Press, pp. 1672–1687.

Hei, X., Liu, Y. & Ross, K. (2008). IPTV over P2P Streaming Networks: the Mesh-pull Approach. *IEEE Communications Magazine,*, 46(2), pp. pp. 86–92.

Huang, C., Li, J. & Ross, K.W. (2007). Peer-Assisted VoD: Making Internet Video Distribution Cheap. In: *Proceedings of IPTPS*, pp. 1–6.

Huang, C., Zhou, G., Abdelzaher, T.F., Son, S.H. & Stankovic, J.A. (2005). Load Balancing in Bounded-Latency Content Distribution. In: *Proceedings of RTSS*, IEEE, pp. 50–61.

Janardhan, V. & Schulzrinne, H. (2007). Peer Assisted VoD for Set-Top Box Based IP Network. In: *Proceedings of P2P-TV*, pp. 1–5.

Jayasundara, C., Nirmalathas, A., Wong, E. & Chan, C.A. (2011). Localized P2P VoD Delivery Scheme with Pre-Fetching for Broadband Access Networks. In: *Proceedings of IEEE GLOBECOM*, pp. 1–5.

Jia, J., Li, C. & Chen, C. (2007). Characterizing PPStream across Internet. In: *Proceedings of NPC*, IEEE, pp. 413–418.

Jin, S., Bestavros, A. & Iyengar, A. (2002). Accelerating Internet Streaming Media Delivery Using Network-aware Partial Caching. In: *Proceedings of ICDCS*, pp. 153–160.

JOOST (2013). JOOST. Available at: http://www.joost.com.

Jung, J., Krishnamurthy, B. & Rabinovich, M. (2002). Flash Crowds and Denial of Service Attacks: Characterization and Implications for CDNs and Web Sites. In: *Proceedings of WWW*, ACM, pp. 293–304.

Kangasharju, J., Roberts, J. & Ross, K. (2002). Object Replication Strategies in Content Distribution Networks. *Computer Communications*, 25(4), pp. 376–383.

Karlsson, M. & Mahalingam, M. (2002). Do We Need Replica Placement Algorithms in Content Delivery Networks. In: *In Proceedings of WCW*, pp. 117–128.

Kawarasaki, M. & Suzuki, K. (2009). Performance Evaluation of Cooperative Peer Selection Methods for P2P Video-on-Demand. In: *Proceedings of ICUMT*, IEEE, pp. 1–6.

Kerpez, K., Luo, Y. & Effenberger, F.J. (2010). Bandwidth Reduction via Localized Peer-to-Peer (P2P) Video. *Digital Multimedia Broadcasting,*, pp. 1–10.

Kim, H. & Yeom, H.Y. (2007). P-chaining: a Practical VoD Service Scheme Autonomically Handling Interactive Operations. *Multimedia Tools and Applications*, 39(1), pp. 117–142.

Kleinrock, L. (1975). *Queuing Systems*, volume I: Theory. Wiley Interscience.

Korosi, A., Lukovszki, C., Szekely, B. & Csaszar, A. (2009). High Quality P2P Video-on-Demand with Download Bandwidth Limitation. In: *Proceedings of IWQoS*, pp. 1–9.

Kostić, D., Rodriguez, A., Albrecht, J. & Vahdat, A. (2003). Bullet: High Bandwidth Data Dissemination Using an Overlay Mesh. *SIGOPS Oper. Syst. Rev.*, 37(5), pp. 282–297.

Kozat, U., Harmanci, O., Kanumuri, S., Demircin, M. & Civanlar, M. (2009). Peer Assisted Video Streaming With Supply-Demand-Based Cache Optimization. *IEEE Transactions on Multimedia*, 11(3), pp. 494–508.

Krogfoss, B., Sofman, L. & Agrawal, A. (2008). Caching Architectures and Optimization Strategies for IPTV Networks. *Bell Labs Technical Journal,*, 13(3), pp. 13–28.

Kulatunga, C., Kandavanam, G., Rana, A., Balasubramaniam, S. & Botvich, D. (2011). HySAC: A Hybrid Delivery System with Adaptive Content Management for IPTV Networks. In: *Proceedings of IEEE ICC*, pp. 1–5.

Kumar, R., Liu, Y. & Ross, K. (2007). Stochastic Fluid Theory for P2P Streaming Systems. In: *Proceedings of IEEE INFOCOM*, pp. 919–927.

Lee, S.B., Muntean, G.M. & Smeaton, A. (2009). Performance-Aware Replication of Distributed Pre-Recorded IPTV Content. *IEEE Transactions on Broadcasting,*, 55(2), pp. 516–526.

Li, B., Xie, S., Qu, Y., Keung, G.Y., Lin, C., Liu, J. & Zhang, X. (2008). Inside the New Coolstreaming: Principles, Measurements and Performance Implications. In: *Proceedings of IEEE INFOCOM*, IEEE, pp. 1031–1039.

Li, B. & Yin, H. (2007). Peer-to-Peer Live Video Streaming on the Internet: Issues, Existing Approaches, and Challenges. *IEEE Communications Magazine,*, 45(6), pp. pp. 94–99.

Li, J. (2008). On Peer-to-Peer (P2P) Content Delivery. *Peer-to-Peer Networking and Applications*, 1(1), pp. 45–63.

Li, T., Chen, M., Chiu, D.M. & Chen, M. (2009). Queuing Models for Peer-to-Peer Systems. In: *Proceedings of IPTPS*, pp. 1–6.

Liao, X., Jin, H., Liu, Y., Ni, L. & Deng, D. (2006). AnySee: Peer-to-Peer Live Streaming. In: *IEEE INFOCOM*, volume 6.

Limelight (2013). Limelight Networks. Available at: `http://www.limelightnetworks.com`.

Liu, J. (2004). Proxy Caching for Media Streaming over the Internet. *IEEE Communications Magazine*, 42(8), pp. 88–94.

Liu, Y., Guo, Y. & Liang, C. (2008). A Survey on Peer-to-Peer Video Streaming Systems. *Peer-to-Peer Networking and Applications,*, 1(1), pp. pp. 18–28.

Loeser, C., Schomaker, G., Brinkmann, A., Vodisek, M. & Heidebuer, M. (2005). Content Distribution in Heterogeneous Video-on-Demand P2P Networks with ARIMA Forecasts. In: *Proceedings of ICN*, volume 3421, Springer, pp. 800–809.

Lu, Y., Zhang, A., He, H. & Deng, Z. (2005). Stochastic fluid model for p2p content distribution networks. In: *Proceedings of ISADS*, pp. 707–712.

Lu, Y., Mol, J.D., Kuipers, F. & Mieghem, P.V. (2008). Analytical Model for Mesh-Based P2PVoD. In: *Proceedings of IEEE ISM*, IEEE Computer Society, pp. 364–371.

Muñoz Gea, J., Nafaa, A., Malgosa-Sanahuja, J. & Rohmer, T. (2012). Design and Analysis of a Peer-Assisted VoD Provisioning System for Managed Networks. *Multimedia Tools and Applications,*, pp. 1–36.

Nygren, E., Sitaraman, R.K. & Sun, J. (2010). The Akamai Network: a Platform for High-performance Internet Applications. *SIGOPS Oper. Syst. Rev.*, 44(3), pp. 2–19.

Padmanabhan, V.N., Wang, H.J., Chou, P.A. & Sripanidkulchai, K. (2002). Distributing Streaming Media Content Using Cooperative Networking. In: *Proceedings of NOSSDAV*, ACM, pp. 177–186.

Pierre, G. & van Steen, M. (2006). Globule: a Collaborative Content Delivery Network. *IEEE Communications Magazine*, 44(8), pp. 127–133.

PPlive (2013a). PPlive. Available at: http://www.coolstreaming.com.

PPlive (2013b). PPlive. Available at: http://www.pplive.com.

PPstream (2013). PPstream. Available at: http://www.ppstream.com.

Presti, F.L., Petrioli, C. & Vicari, C. (2007). Distributed Dynamic Replica Placement and Request Redirection in Content Delivery Networks. In: *Proceedings of MASCOTS*, IEEE Computer Society, pp. 366–373.

Qiu, D. & Srikant, R. (2004). Modeling and Performance Analysis of BitTorrent-like Peer-to-Peer Networks. In: *Proceedings of SIGCOMM*, pp. 367–378.

Qiu, L., Padmanabhan, V.N. & Voelker, G.M. (2001). On the Placement of Web Server Replicas. In: *Proceedings of IEEE INFOCOM*, volume 3, IEEE, pp. 1587–1596.

Radoslavov, P., Govindan, R. & Estrin, D. (2002). Topology-informed Internet Replica Placement. *Computer Communications*, 25(4), pp. 384–392.

Ramachandran, K.K. & Sikdar, B. (2005). An Analytic Framework for Modeling Peer to Peer Networks. In: *Proceedings of IEEE INFOCOM*, volume 3, IEEE, pp. 2159–2169.

Rao, S. & Seshan, S. (2002). A Case for End System Multicast. *IEEE Journal on Selected Areas in Communications*, 20(8), pp. 1456–1471.

Rimac, I., Elwalid, A. & Borst, S. (2008). On Server Dimensioning for Hybrid P2P Content Distribution Networks. In: *Proceedings of P2P*, IEEE, pp. 321–330.

Rodriguez, P., Spanner, C. & Biersack, E. (2001). Analysis of Web Caching Architectures: Hierarchical and Distributed Caching. *IEEE/ACM Trans. Netw.*, 9(4), pp. 404–418.

Simpson, W. & Greenfield, H. (2007). IPTV and Internet Video. *P14*.

Simsarian, J. & Duelk, M. (2007). IPTV Bandwidth Demands in Metropolitan Area Networks. In: *Proceedings of LANMAN*, pp. 31–36.

SopCast (2013). SopCast. Available at: http://www.sopcast.org.

Spoto, S., Gaeta, R., Grangetto, M. & Sereno, M. (2009). Analysis of PPLive Through Active and Passive Measurements. In: *Proceedings of IEEE IPDPS*, pp. 1–7.

Stockhammer, T. & Heiles, J. (2009). DVB-IPTV Content Download Services-IPTV Services Anytime and Anywhere. In: *Proceedings of ConTEL*, IEEE, pp. 413–420.

Sugavanam, P. (2007). Robust Static Allocation of Resources for Independent Tasks Under Makespan and Dollar Cost Constraints. *Journal of Parallel and Distributed Computing*, 67(4), pp. 400–416.

Suh, K., Diot, C., Kurose, J., Massoulie, L., Neumann, C., Towsley, D.F. & Varvello, M. (2007). Push-to-Peer Video-on-Demand System: Design and Evaluation. *IEEE Journal on Selected Areas in Communications,*, 25(9), pp. 1706–1716.

Tran, D., Hua, K., Zigzag, T.D. & An (2003). ZIGZAG: an Efficient Peer-to-Peer Scheme for Media Streaming. *Proceedings of IEEE INFOCOM*.

Tu, Y.C., Sun, J., Hefeeda, M. & Prabhakar, S. (2005). An Analytical Study of Peer-to-Peer Media Streaming Systems. *ACM Trans. Multimedia Comput. Commun. Appl.*, 1(4), pp. 354–376.

TVAnts (2013). TVAnts. Available at: http://www.tvants.com.

UUSee (2013). UUSee. Available at: http://www.uusee.com.

Varga, A. (2013). INET Framework. Available at: http://inet.omnetpp.org.

Varga, A. & Hornig, R. (2008). An Overview of the OMNeT++ Simulation Environment. In: *Proceedings of Simutools*, pp. 1–10.

Verhoeyen, M., De Vleeschauwer, D. & Robinson, D. (2008). Content Storage Architectures for Boosted IPTV Service. *Bell Labs Technical Journal,*, 13(3), pp. 29–43.

Vlavianos, A., Iliofotou, M. & Faloutsos, M. (2006). BiToS: Enhancing BitTorrent for Supporting Streaming Applications. In: *Proceedings IEEE INFOCOM*, IEEE, pp. 1–6.

Wang, H., Liu, P. & Wu, J.J. (2006). A QoS-Aware Heuristic Algorithm for Replica Placement. In: *Proceedings of GRID*, IEEE, pp. 96–103.

Wu, D., Liu, Y. & Ross, K. (2009). Queuing Network Models for Multi-channel P2P Live Streaming Systems. In: *Proceedings of IEEE INFOCOM*, IEEE, pp. 73–81.

Wu, D., Liu, Y. & Ross, K. (2010). Modeling and Analysis of Multichannel P2P Live Video Systems. *IEEE/ACM Transactions on Networking,*, 18(4), pp. pp. 1248–1260.

Wu, K.L., Yu, P. & Wolf, J. (2004). Segmentation of Multimedia Streams for Proxy Caching. *IEEE Transactions on Multimedia*, 6(5), pp. 770–780.

Xie, S., Li, B., Keung, G.Y. & Zhang, X. (2007). Coolstreaming : Design, Theory, and Practice. *IEEE Transactions on Multimedia*, 9(8), pp. 1661–1671.

Yang, M. & Fei, Z. (2003). A Model for Replica Placement in Content Distribution Networks for Multimedia Applications. In: *Proceedings of IEEE ICC*, volume 1, pp. 557–561.

Yiu, W.P. & Chan, S.H. (2007). VMesh: Distributed Segment Storage for Peer-to-Peer Interactive Video Streaming. *IEEE Journal on Selected Areas in Communications*, 25(9), pp. 1717–1731.

Yu, H., Zheng, D., Zhao, B.Y. & Zheng, W. (2006). Understanding User Behavior in Large-scale Video-on-Demand Systems. *ACM SIGOPS Operating Systems Review*, 40(4), pp. pp. 333–344.

Zhou, X. & Xu, C. (2007). Efficient Algorithms of Video Replication and Placement on a Cluster of Streaming Servers. *Journal of Network and Computer Applications*, 30(2), pp. 515–540.

Zhou, Y., Fu, T. & Chiu, D.M. (2011). Statistical Modeling and Analysis of P2P Replication to Support VoD Service. In: *Proceedings of IEEE INFOCOM*, pp. 945–953.

Zhu, P., Yoshiuchi, H. & Yoshizawa, S. (2010). P2P-Based VOD Content Distribution Platform with Guaranteed Video Quality. In: *Proceedings of IEEE CCNC*, pp. 1–5.