



**Julius-Maximilians-Universität Würzburg**

Institut für Informatik  
Lehrstuhl für Verteilte Systeme  
Prof. Dr. P. Tran-Gia

# **Performance Evaluation of Future Internet Applications and Emerging User Behavior**

**Tobias Hoßfeld**

Würzburger Beiträge zur  
Leistungsbewertung Verteilter Systeme

Bericht 01/09

# **Würzburger Beiträge zur Leistungsbewertung Verteilter Systeme**

## **Herausgeber**

Prof. Dr. P. Tran-Gia  
Universität Würzburg  
Institut für Informatik  
Lehrstuhl für Verteilte Systeme  
Am Hubland  
D-97074 Würzburg  
Tel.: +49-931-31-86630  
Fax.: +49-931-888-6632  
email: [trangia@informatik.uni-wuerzburg.de](mailto:trangia@informatik.uni-wuerzburg.de)

## **Satz**

Reproduktionsfähige Vorlage vom Autor.  
Gesetzt in L<sup>A</sup>T<sub>E</sub>X Computer Modern 9pt.

**ISSN 1432-8801**

# **Performance Evaluation of Future Internet Applications and Emerging User Behavior**

Dissertation zur Erlangung des  
naturwissenschaftlichen Doktorgrades  
der Julius–Maximilians–Universität Würzburg

vorgelegt von

**Tobias Hoßfeld**

aus

Fulda

Würzburg 2009

Eingereicht am: 22.05.2009

bei der Fakultät für Mathematik und Informatik

1. Gutachter: Prof. Dr.-Ing. P. Tran-Gia

2. Gutachter: Prof. Dr. K. Pawlikowski

Tag der mündlichen Prüfung: 04.08.2009

# Danksagung

*Leider läßt sich eine wahrhafte Dankbarkeit mit Worten nicht ausdrücken.* Johann Wolfgang von Goethe (1749-1832)

Nach fünf Jahren intensiver Forschung am Lehrstuhl für Verteilte Systeme möchte ich allen Personen danken, die während dieser Zeit dazu beigetragen haben, diese Arbeit erfolgreich abzuschließen.

Allen voran gilt mein besonderer Dank meinem Doktorvater und Betreuer, Herrn Prof. Dr.-Ing. Phuoc Tran-Gia, der es mir ermöglicht hat, in einem spannenden und aktuellen Forschungsgebiet zu promovieren. Dabei hat er mich stets in meinen Forschungen unterstützt und stand mir mit Rat und Tat beiseite, um die eine oder andere wissenschaftliche Fragestellung in Angriff zu nehmen. Ich danke ihm insbesondere dafür, daß ich während meiner Promotion zahlreiche internationale Konferenzen und Kollegen im In- und Ausland besuchen durfte, durch die ich immer wieder neue Ideen und Impulse bekommen habe. Gerade durch die engen Kooperationen mit den Partnern aus der Industrie und Forschung entstanden neue Fragestellungen, die in meine Dissertation eingegangen sind. Vor allem aber möchte ich mich für das Vertrauen bedanken, welches Prof. Tran-Gia mir entgegengebracht hat. Durch dieses Vertrauen konnte ich Projekte selbstständig und in eigener Verantwortung durchführen und konnte so neben den wertvollen Erfahrungen in meinem Forschungsbereich noch darüber hinausgehende Erfahrungen sammeln. Prof. Tran-Gia schafft durch sein unermüdliches Engagement eine angenehme Quality of Experience am Lehrstuhl und ein positives Klima unter den Kollegen und Studenten, die wesentlich zum Gelingen der Arbeiten und der Freude an der Forschung beitragen. Ganz besonders herzlich

möchte ich mich bei Prof. Tran-Gia für seine menschliche Nähe bedanken und daß er mir in schwierigen persönlichen Situation geholfen hat.

My deepest appreciation and gratitude goes to Prof. Dr. Krzysztof Pawlikowski, who acted as a reviewer of this thesis and provided me with valuable comments on my research. Ein herzliches Dankeschön auch an Dr. Wolfgang Kellerer, der mir gute Ratschläge beim Schreiben der Dissertation gegeben hat. Danken möchte ich auch Prof. Dr. Klaus Schilling und Prof. Dr. Frank Puppe, die als Prüfer für meine Disputation zur Verfügung standen.

Ein besonderes Dankeschön geht an Prof. Dr. Kenji Leibnitz, mit dem mich eine enge Freundschaft – auch über geographische Grenzen hinaus – verbindet und von dem ich über die Jahre sehr viel in Bezug auf wissenschaftliches Arbeiten gelernt habe. Ich bin zuversichtlich, dass die enge wissenschaftliche Zusammenarbeit auch in Zukunft sich so erfolgreich gestalten wird. Gleiches gilt für Prof. Dr. Markus Fiedler, von dem ich in zahlreichen Diskussionen – über wissenschaftliche wie auch andere Probleme – viele Anregungen erhalten habe. Auf die fachlicher Kompetenz beider konnte ich mich stets stützen und beide haben wesentlich dazu beigetragen, dass diese Arbeit in der vorliegenden Form entstanden ist. In the same way, I would like to thank Prof. Dr. Marie-Ange Remiche for the fruitful collaboration and her engagement from which I benefit very much.

Weiterhin möchte ich mich bei Prof. Hermann de Meer, Frank-Uwe Andersen und Cornel Pampu für die gute Zusammenarbeit in den Forschungsprojekten bedanken. Rege Diskussion gab und gibt es auch immer wieder mit den Kollegen aus den Projekten G-Lab, Euro-NF und SmoothIt, sowie aus den Projekten mit den Firmen Bosch, Siemens, Bertelsmann Arvato und Datev.

Wichtig für mein erfolgreiches wissenschaftliches Arbeiten war jedoch immer mein persönliches Umfeld und der Kontakt zu den Menschen am Lehrstuhl. Neben den fachlichen Diskussionen mit den Kollegen im Kaffeeraum oder auf Dienstreisen, blieb vor allem immer noch genügend Zeit für nicht-fachliche Gespräche und gemeinsame Aktivitäten, die das Leben am Lehrstuhl so angenehm machen. So entstanden im Laufe der Jahre aus den Peer-to-Peer

Beziehungen echte Freundschaften und ich bin sicher, mit einigen noch den einen oder anderen Lauf zu bestreiten, ein paar lustige Kickerabende zu verbringen oder sich gemütlich dem gemeinsamen Urlaub zu widmen. Ich möchte allen Kollegen, die mich im Laufe der Jahre begleitet haben, meinen aufrichtigen Dank aussprechen: Dr. Andreas Binzenhöfer, Michael Duelli, Dr. Mathias Dümmler, Matthias Hartmann, Dr. Klaus Heck, Robert Henjes, David Hock, Alexander Klein, Dr. Stefan Köhler, Frank Lehrieder, Dr. Andreas Mäder, Dr. Rüdiger Martin, Dr. Michael Menth, Dr. Jens Milbrandt, Simon Oechsner, Rastin Pries, Prof. Dr. Oliver Rose, Daniel Schlosser, Barbara Staehle, Prof. Dr. Kurt Tutschku, Dr. Norbert Vicari, Florian Wamser und Thomas Zinner.

Bedanken möchte ich mich auch bei Frau Gisela Förster für ihre Unterstützung bei der Verwaltungsarbeit und der Organisation des EuroView-Workshops.

Weiterhin möchte ich mich bei meinen Diplomanden, Daniel Schlosser, Thomas Zinner, Michael Duelli, Matthias Kuhnert und Christian Bergner, sowie alle Studenten bedanken, die mich bei der Projektarbeit und der Betreuung von Vorlesungen unterstützt haben.

Zum Schluß möchte ich mich ganz besonders bei meiner lieben Familie bedanken, meinen Eltern Ewald und Angela Hoßfeld, sowie meiner Schwester Dr. Susanne Marx und ihrem Ehemann Christian Marx. Meine Eltern haben mir das Studium der Informatik ermöglicht und mir auch während meiner Promotionszeit alle Unterstützung zukommen lassen. Am wichtigsten war jedoch, dass sie immer an mich geglaubt haben und mir die Liebe und Kraft gaben, dass ich irgendwann selber daran geglaubt habe. Meiner Schwester Susanne danke ich für ihr Mentaltraining in sämtlichen Disziplinen und dass sie immer für mich da war, wenn ich sie gebraucht habe.





# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Scientific Contribution . . . . .	3
1.2	Outline of Thesis . . . . .	7
<b>2</b>	<b>Cooperation in Mobile Peer-to-Peer</b>	<b>11</b>
2.1	Background: Cooperation Strategies for Content Distribution . . .	13
2.1.1	Content Distribution with Multi-Source Download . . . . .	15
2.1.2	Common Cooperation Strategies . . . . .	17
2.1.3	Key Performance Characteristics . . . . .	21
2.1.4	Related Work on Cooperation Strategies . . . . .	23
2.2	Selfishness of Users and Robustness of the System . . . . .	26
2.2.1	Last Chunk Problem . . . . .	27
2.2.2	CycPriM Strategy . . . . .	29
2.2.3	Investigated Scenarios of User Behavior . . . . .	33
2.2.4	Performance Comparison . . . . .	36
2.3	Content Distribution in Heterogeneous Cellular Networks . . . . .	43
2.3.1	Effects of Mobility . . . . .	43
2.3.2	Modeling Mobility and VHO in Cellular Networks . . . . .	45
2.3.3	Impact on Today's and Future Cellular Networks . . . . .	47
2.3.4	Mastering Mobility with Time-based Data Exchange . . . . .	50
2.3.5	Utilization of Scarce Resources . . . . .	53
2.4	Future Trends in Mobile Peer-to-Peer . . . . .	57
2.5	Lessons Learned . . . . .	59

<b>3</b>	<b>Modeling of Online TV Recording Services</b>	<b>61</b>
3.1	Background: Video Content Delivery . . . . .	65
3.1.1	Classification of Video Delivery Services . . . . .	66
3.1.2	Online TV Recording Service . . . . .	67
3.1.3	Related Work . . . . .	69
3.2	Measurement of Video Contents . . . . .	73
3.2.1	Network-based Personal Video Recorder – OTR . . . . .	74
3.2.2	Server-based Video-on-Demand – YouTube . . . . .	76
3.3	High-Performance Server Clusters . . . . .	77
3.3.1	Model of Centralized OTR System . . . . .	78
3.3.2	Time-Dynamic Evaluation with Fluid Model . . . . .	79
3.3.3	Derivation of the Stationary Sojourn Time Distribution . . . . .	82
3.3.4	Understanding the Key Influence Factors . . . . .	95
3.4	Pollution of P2P Content Distribution Service . . . . .	101
3.4.1	Epidemic Model of File Diffusion . . . . .	102
3.4.2	Analytic Modeling of Pollution in the P2P Network . . . . .	109
3.4.3	Solving the Differential Equation System . . . . .	114
3.5	Comparison of Client/Server and P2P . . . . .	119
3.5.1	Success Ratio . . . . .	120
3.5.2	Download Duration . . . . .	121
3.5.3	Fairness Issues . . . . .	122
3.6	Lessons Learned . . . . .	123
<b>4</b>	<b>QoE of Edge-Based VoIP Applications</b>	<b>127</b>
4.1	Background: Assessment of Quality . . . . .	130
4.1.1	Quality Comparisons and Classification of Metrics . . . . .	131
4.1.2	Qualitative Relationship Between QoE and QoS . . . . .	135
4.1.3	The Exponential Interdependency of QoE and QoS Hypothesis . . . . .	137
4.1.4	Related Work . . . . .	138

4.2	Measurement Testbed and Setup . . . . .	144
4.2.1	Computation of QoS and QoE Parameters . . . . .	146
4.2.2	Verification of the Emulation of Network Conditions . . . . .	151
4.3	QoE of Voice Codecs iLBC and G.711 . . . . .	155
4.3.1	Approach to Test the IQX Hypothesis . . . . .	158
4.3.2	Voice Quality Affected by Loss . . . . .	159
4.3.3	Jitter and Reordering . . . . .	160
4.3.4	Autocorrelated Packet Streams . . . . .	164
4.4	Skype VoIP Traffic in UMTS . . . . .	168
4.4.1	Emulated Rate-Controlled DCH in UMTS . . . . .	170
4.4.2	Replication of Voice Data . . . . .	173
4.4.3	Measurement in a Public UMTS Network . . . . .	180
4.4.4	Emulate Dynamic Changes in UMTS . . . . .	186
4.5	QoE Management and Provisioning . . . . .	191
4.6	Lessons Learned . . . . .	194
<b>5</b>	<b>Conclusions</b>	<b>199</b>
	<b>Nomenclature</b>	<b>203</b>
	<b>List of Acronyms</b>	<b>209</b>
	<b>Bibliography and References</b>	<b>213</b>



# 1 Introduction

Over the last years, new paradigms and concepts have emerged in telecommunication systems that are currently being realized in the Internet. Among those are the overlay, Peer-to-Peer (P2P), and the Quality of Experience (QoE) paradigms. An *overlay* or an overlay network is a flexible, logical network that is built on top of an existing substrate network. Overlays are used to overcome prevailing technical limitations of the Internet, e.g. multicast, or to facilitate simplified implementation of sophisticated new mechanisms on a logical layer, e.g. re-routing on application layer in case of congested end-to-end paths. Note that the Internet itself has evolved as an overlay on top of the plain old telephone system to support new packet-switched data services.

In a *Peer-to-Peer* (P2P) network, the nodes of this network, called peers, share common resources, e.g. bandwidth or memory, in order to provide or support a certain service, like content distribution networks (CDN) or distributed lookup systems. Typically, the peers form an overlay for communicating with each other. The capabilities of P2P facilitate the deployment of new functionalities, like direct any-to-any communication or sharing of user-generated contents, as well as help to overcome restrictions on resources, e.g. in terms of storage capacity for a CDN. To this end, the application of the fundamental P2P paradigm fosters the realization of future Internet applications and allows saving infrastructure costs by using existing resources in a more efficient way.

Furthermore, the technological advancements in high-speed Internet access enable the realization of the P2P potential and propel the use of the Internet into a new era. New applications have emerged that are bandwidth intensive or have strict Quality of Service (QoS) requirements. The most popular applications up

to now are P2P file sharing applications that serve as a new medium for CDNs like eDonkey or BitTorrent. Recently, new types of overlay applications have appeared and gained popularity, such as P2P-based voice and video services. Examples are the popular Skype Voice-over-IP application or online video recording systems.

The user's satisfaction with a particular application is expressed by the *Quality of Experience* (QoE) measure. Degradation in QoS, like packet loss, packet reordering, and large jitter in the network, may lead to strong decrease in QoE, which is the case for VoIP applications for instance. Beside such objective end-to-end QoS parameters, QoE focuses rather on subjective evaluations of service delivery by the end users. It addresses service reliability comprising service availability, accessibility, access time and continuity, as well as service comfort including session quality, ease of use and level of support. From this perspective, QoE will be the major criterion for the subscriber to select a specific service.

The composition of these paradigms may result in multi-network services with edge-based intelligence. In future telecommunication systems, we observe an increasing diversity of access networks and the fixed to mobile convergence between wireline and wireless networks. This implies an increasingly heterogeneous networking environment for applications and services. The separation of transport services and applications, or between different services leads to *multi-network services*. A future service has to work transparently to the underlying network infrastructure and independently of the user's current location and access technology. In this sense, a multi-network service establishes a logical overlay on top of different access networks.

The Internet Protocol is currently the smallest common denominator for such multi-network services. Still, roaming users expect these services to work in a satisfactory way, i.e. a good QoE, regardless of the currently available access technology. Thus, a true multi-network service must be able to adapt itself to its environment to a much stronger degree than what is supported by the Internet protocol suite. Streaming multimedia applications for example face the problem that their predominant transport protocol UDP does not take any feedback from

the network into account. Consequently, any quality control and adaptation has to be applied by the application itself at the edge of the network. The network providers have to cope with the fact that these edge-based applications dynamically determine the amount of consumed bandwidth. In particular, applications such as Skype do their own network quality measurements and react to quality changes in order to keep their users satisfied. This *edge-based intelligence* is established via traffic control on application layer.

The shift of the control intelligence to the edge is accompanied by the fact that the observed user's behavior also changes. A user can appear either altruistic or selfish. Selfish user behavior means that the user or the application tries to maximize the user-perceived QoE rather than to optimize the overall network QoS. Very often such selfish behavior is implemented in the software downloaded by the user without his explicit notice. In contrast, altruistic users, whose behavior is mostly influenced by the network provider's traffic control protocols (like TCP) help to maximize the overall system performance in a fair manner. In the case of file sharing platforms, an altruistic user is willing to upload data to other users, while a selfish user only wants to download without contributing to the network. For VoIP, altruistic users would reduce the consumed bandwidth in the case of facing congestion, while selfish users would continuously try to achieve a high goodput and QoE, irrespective of the consequences for the other users.

## 1.1 Scientific Contribution

The intention of this thesis is threefold. First, we aim at modeling and evaluating future Internet applications from a user-centric view instead of using a classic network-centric view. Next, the identified problems and challenges as well as the emerging user behavior are highlighted, which go along with the realization of the upcoming overlay, P2P, and QoE paradigms. The observation of changes in the user behavior is important for the performance evaluation of future services and also for their dimensioning. However, the changing user behavior affects not only the performance of the investigated systems, but requires also to develop a

methodology and to derive appropriate models for analyzing their performance. Finally, this performance modeling permits a proper design of future Internet applications that are beneficial for its users.

For this purpose, we look at currently existing applications to estimate those that may become relevant in the future and identify and model the user behavior. Beside the paradigm changes, the available technologies and environments also affect user behavior. For instance, let us consider a mobile subscriber of a P2P-based file sharing service. The advances in wireless technology may allow for user mobility, even perhaps between different network access types. However, this also introduces heterogeneity according to the capabilities of the access network the user is currently connected to. As a result, the complexity to coordinate the users and resources in the P2P network increases, while trying to maintain providing an efficient and fair file sharing service. As a result of mobility, the user behavior will also change according to the radio coverage. In case of coverage loss, the user appears to be offline and after getting network access again he might appear as an entirely new user in the overlay, e.g. when receiving a new IP address. To save battery power, a mobile user might additionally switch more often to offline mode. Thus, increased dynamics in the user behavior and in the overlay topology are observed, which results in higher churn in the P2P system. Due to the popularity of the contents, the dynamics of a P2P file sharing service is further increased, as download requests may occur as flash crowds, i.e., a large number of users requests a certain file within short time.

The expensive upload capacity of a mobile user may also cause selfish user behavior by reducing the amount of uploaded data to other users in the P2P network. Since the users in a P2P network act as servers, the willingness of users to share resources has to be considered. Furthermore, it has to be taken into account, that in contrast to classic client/server systems a shared file is no longer at a single trusted server location. Thus, malicious users may offer a corrupted version of a file or parts of it to disturb the service. This is referred to as poisoning or pollution depending on whether the decoy was offered deliberately or not. As effect of pollution or poisoning, the download times are prolonged and



the QoE of users may be decreased. When the user's patience is exceeded, he will abort the download and abandon the service. The degree of satisfaction determines the user's impatience. Edge-based intelligence taking into account QoE feedback will additionally affect the network traffic and the user behavior pattern.

As consequences of the emerging user behavior and the user's perception of the service quality, a methodology has to be developed and appropriate models have to be provided for evaluating the performance of future Internet applications. The obtained results from performance evaluation allow to quantify their technical impact and to derive solutions to overcome problems. Returning to the example of mobile P2P file sharing, large-scale systems with a high number of users have to be analyzed. The complex interaction of mobile and heterogeneous users in the overlay has to be modeled in such a way, that all relevant effects are captured within reasonable computational time. To this end, we provide a semi-Markov model for user mobility in cellular wireless networks which enables us to simulate large-scale P2P networks with mobile users. This allows investigating e.g. the application of Mobile IP techniques and to study and predict the performance of common P2P cooperation strategies, as applied by eDonkey or BitTorrent, in current and future cellular networks. A recommendation for the usage of Mobile IP in different scenarios is given. As a result of the identified problems we derive a novel cooperation strategy to master mobility and an adaptive strategy to utilize the scarce resources in such heterogeneous networks.

Figure 1.1 gives an overview of the contribution in this thesis. The various research studies carried out during the course of this work are classified according to the major methodology on the x-axis and the main focus or mainly investigated technology of the study on the y-axis. The methodology is distinguished between measurement studies, simulative performance evaluation, mathematical analysis, and design of new mechanisms, services or applications. The main focus of the research study considers overlays, P2P, QoE, and wireless systems. It has to be noted that this classification aims at highlighting only the main contributions. However, some studies cover several areas, e.g. mobile P2P file sharing which is therefore placed between "wireless" and "P2P". The same is true regarding the

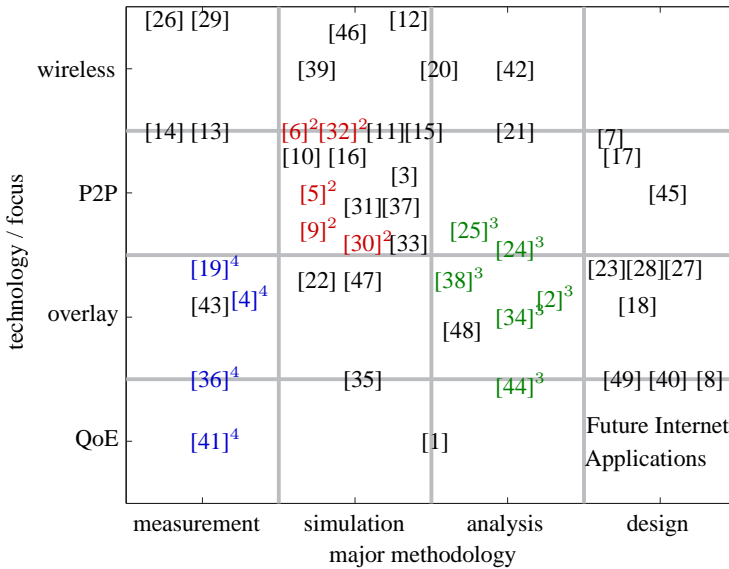


Figure 1.1: Contribution in thesis is illustrated by a cartography of the research studies carried out. The notion  $[x]^y$  indicates that the scientific publication  $[x]$  is discussed in Chapter  $y$  of this monograph.

applied methodology for some studies discussed in this thesis.

In this monograph, three important issues in the future Internet are selected which cover a broad area in the classification scheme of Figure 1.1. The corresponding chapter number marks the related scientific publications of the three examples. The first issue covers mobile P2P file sharing as discussed above. The second issue addresses modeling of online TV recording services and aims at a performance comparison of a high-performance server cluster and a P2P-based system in terms of reliability, efficiency and fairness. As a result of the performance study, the high-performance server cluster can be properly dimensioned. In the case of the P2P system, the model allows investigating the impact of ma-

licious or fake peers and their impact on the impatience of regular users. This can be exploited for two reasons. First, the disturbance of the P2P system due to malicious peers is quantified when the service provider relies on P2P technology. Second, it allows dimensioning the number of fake peers to protect copyrighted contents from illegally being distributed in a file sharing system.

The third important topic considers the QoE of edge-based VoIP applications, since the QoE mainly determines the behavior of a user. In particular, it is investigated how the current network conditions described as QoS parameters influence the QoE of a VoIP user. As a major contribution of this example, the IQX hypothesis is formulated and derived as an exponential functional relationship between QoE and QoS disturbance. It is tested and validated for existing measurement studies in web browsing, as well as for VoIP applications based on own extensive measurement studies in a controlled testbed. In addition, it is studied in how far an edge-based application like Skype reacts to quality degradations. Starting from measurements of the Skype application, we show the basic properties of selfish and altruistic user behavior in accordance to edge-based intelligence.

## 1.2 Outline of Thesis

The organization of this monograph and the contributions in the individual chapters are illustrated in Figure 1.2. For each of the three selected examples an individual chapter is devoted with a similar structure. Each chapter has a background and related work section and summarizes the lessons learned. Figure 1.2 shows for each chapter three different columns which are related to (1) the impact of user behavior and perception, (2) its consequences for the applied methodology, and (3) its technical impact and the derived solutions to overcome the identified problems. Arrows between the building blocks within the diagram show that either fundamental background is introduced or that the findings are utilized in later sections. The section numbers of the building blocks are given in parentheses.

The remainder of this monograph is organized as follows. In Chapter 2, cooperation in mobile P2P networks for content distribution is investigated. A com-

prehensive background on the multi-source download mechanism as key feature of P2P CDNs and common cooperation strategies, as used by eDonkey or BitTorrent, are given. Additionally, we review related work which addresses heterogeneity and selfishness in general. As investigated user behavior, we consider the impact of selfishness, altruism, and mobility. Furthermore, the heterogeneity of users stemming from different access technologies is taken into account. For the quantification of the performance, we define key metrics like reliability or chunk availability. They reveal the fundamental last chunk problem and the need to derive appropriate cooperation strategies to overcome this problem. As solution, we propose the so-called CycPriM cooperation strategy. Its performance is compared with common strategies in different user behavior scenarios. However, the simulation of mobile users with heterogeneous network access requires too much computational effort and is not feasible in practice. To this end, a new semi-Markov model is proposed which allows investigating the impact of the user behavior. We consider the application of Mobile IP techniques and study the performance of common cooperation strategies in contemporary and future cellular networks for different load scenarios. Again, as a result of the identified problems, we derive a time-based cooperation strategy to master mobility and an adaptive strategy to utilize the scarce resources in such heterogeneous networks. Finally, important developments and future trends in the area of mobile P2P are shown.

Chapter 3 addresses the second example on modeling an online TV recording service. It aims at a performance comparison of a high-performance server cluster and an eDonkey-based P2P system for delivery of OTR video contents in terms of reliability, efficiency and fairness. We provide appropriate queueing and fluid models to describe pollution by malicious peers and time dynamics, e.g. due to flash crowd effects. Pollution in a P2P system may result in prolonged download times, while flash crowds may overburden server clusters. As a result in both cases the users may get impatient. Thus, we consider performance measures as introduced in Chapter 2, but due to user impatience we also have to take into account success ratio as essential QoE indicator. For obtaining realistic file sizes of

video contents available in the Internet, a comprehensive measurement study was conducted. To get numerical values from the proposed analytical models, an approximation of matrix exponentials was applied in case of the queueing system, while the Runge-Kutta method was applied to approximate solutions of differential equations systems. Together with the measured video data, we compare the performance of both systems and dimension them according to their desired purpose.

Chapter 4 focuses on the user perception of the quality of a VoIP application, as a user will react according to the actual QoE. In particular, it is investigated how the current network conditions described as QoS parameters influence the QoE of a VoIP user. As a major contribution of this chapter, the IQX hypothesis is formulated and derived as an exponential functional relationship between QoE and QoS disturbance. To quantify the influence of QoS problems on the QoE for VoIP applications and to test the IQX hypothesis, a measurement study in a controlled testbed was carried out to measure the quality of VoIP traffic. Thereby, the applied methodology comprises measurements on network and application level, emulation of network conditions, as well as the validation of the measurement testbed. Furthermore, related work dealing with user experience in web browsing is reviewed and we demonstrate that the exponential interdependency is also valid there. Non-linear regression analysis was used to test the hypothesis. As a result of the study, simple mapping functions between QoE and QoS parameters are derived which can be used in edge-based applications to control and adapt the QoE. Next, the edge-based Skype VoIP application is investigated which tries to maintain the QoE of its user and makes the observed user behavior appear selfish from network traffic's point of view. This selfish user behavior by means of replicated sending of voice datagrams is analytically investigated with respect to the obtained QoE of a single user. This demonstrates the usability of the derived hypothesis. After that, QoE management and provisioning is discussed in general.

Finally, Chapter 5 summarizes the main findings gained throughout the course of this work.

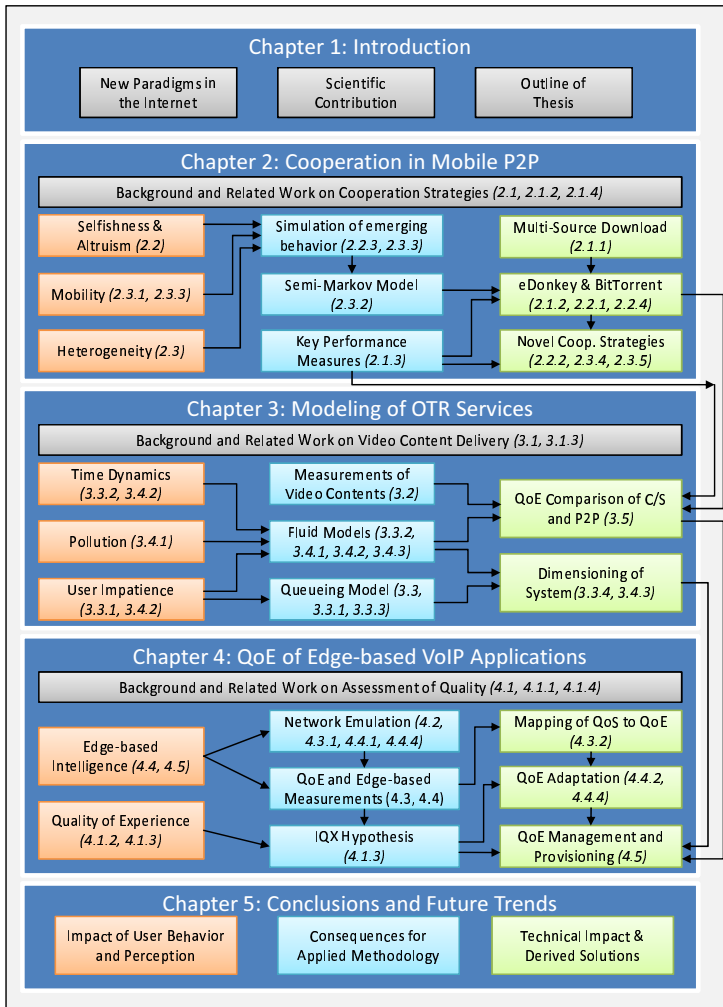


Figure 1.2: Organization and contribution of this monograph

## 2 Cooperation in Mobile Peer-to-Peer

P2P file sharing systems contribute to the majority of traffic volume currently being transported in the Internet. Applications like eDonkey or BitTorrent are used to share large volume content and alleviate the problem of overloaded servers by distributing the load among all sharing peers, which makes P2P systems scalable and resilient. The performance of such P2P content distribution networks (CDN) in cellular networks depends highly on the coordination of heterogeneous and often selfish mobile users. Sophisticated cooperation strategies, such as the multi-source download (MSD) and tit-for-tat principle, are the foundation of the extreme efficiency of P2P content distribution networks. Multi-source download means the simultaneous download of parts of a file, referred to as chunk, from several sources in parallel. The cooperation strategies applied in popular P2P CDN platforms such as eDonkey or BitTorrent, rely on the fundamental P2P assumption that all peers are equal. In cellular networks, however, the peers differ significantly in their characteristics, e.g. their access system and bandwidth which might change over time or their on-line behavior, thus introducing heterogeneity and even selfishness in the peer community. Hence, the P2P assumption of equal peers is not valid any more. In addition, the dynamics and heterogeneity in cellular mobile networks is further increased by the mobility of users.

Although most P2P CDNs use the benefits of multi-source downloads, the various platforms differ significantly in the actual implementation of the cooperation algorithms. In particular, the peer selection as well as the chunk selection mechanisms lead to different system behaviors and performance results. The detailed

performance of the strategies is further determined by the actual peer characteristics and the peer behavior. The peer characteristic includes, among others the available upload and download bandwidth, as well as the number of parallel upload and download connections. The mobility of a user makes these peer characteristics change over time. Thus, the performance depends considerably on the heterogeneity. The peer behavior is mainly described by churn, i.e. the switching of a user between offline and online state, and by the willingness of a user to participate in the CDN. A user may behave selfish and tries to minimize the upload of data or he may redistribute the data in an altruistic way. In the context of cellular mobile networks, churn and selfish behavior appear even more distinctive, e.g. to save battery resources or scarce and expensive uplink capacities. As a result, the so-called “last chunk” problem might arise which inhibits the data dissemination process and makes individual chunks starve in the network.

Additional challenges and influence factors on the performance of the system arise in a heterogeneous, wireless cellular network. We consider a beyond third generation (B3G) network with different infrastructure-based radio access technologies, in particular UMTS and WLAN. Due to the user mobility, vertical handovers (VHO) between the different wireless access technologies are required which may result in transmission delays and IP address changes of the switching peer. We investigate whether it is recommended to use mechanisms like Mobile IP in the context of P2P-based content distribution in cellular environments, since such mechanisms also introduce additional delays. Another important phenomenon occurring with VHOs is the abrupt change of available bandwidth, e.g., from a fast WLAN connection to a rather slow UMTS connection.

There are several possibilities to improve the performance of content distribution in cellular networks. Those are (a) particular architecture concepts introducing special entities like caches for storing contents or crawlers for locating sources, e.g. [17], (b) the optimization of parameters, like the size of chunks, as proposed by [16], (c) incentives to motivate the users to share files and to contribute to the system, and (d) cooperation strategies for the coordination among peers. From these possibilities, we will focus on the cooperation strategies in this



chapter. The goal is (i) to describe how to model a P2P content distribution system with multi-source download in a cellular environment, (ii) to identify the fundamental problems of typical cooperation strategies, (iii) to investigate the impact of user behavior and heterogeneity, in particular selfishness, mobility and VHO, and (iv) to propose solutions to overcome the derived problems.

This chapter is organized as follows. Section 2.1 gives comprehensive background on the multi-source download mechanism and common cooperation strategies, as used by eDonkey or BitTorrent. We define key metrics for evaluating the performance of such systems and review related work which addresses heterogeneity and selfishness in general. In the Section 2.2, we discuss the fundamental last chunk problem and show how the proposed CycPriM cooperation strategy allows overcoming this. Its performance is compared with common strategies in different user behavior scenarios. In Section 2.3, the effects of user mobility in a B3G network on the traffic characteristics are revealed. This understanding makes us derive an abstract mobility model subsuming the network layout and the user mobility using a semi-Markov model. We consider the application of Mobile IP techniques and investigate the performance of common cooperation strategies in today's and future cellular networks for different load scenarios. Again, as a result of identified problems we derive a time-based cooperation strategy to master mobility and an adaptive strategy to utilize the scarce resources in such heterogeneous networks. Finally, Section 2.4 shows our particular viewpoint on important developments and necessary future work in this area, before Section 2.5 summarizes the lessons learned in this chapter.

## **2.1 Background: Cooperation Strategies in Content Distribution Networks**

The mechanisms to control and manage content distribution in P2P networks can be distinguished in two major categories: (a) resource mediation mechanisms, which are functions for searching and locating resources and (b) resource access

control mechanisms, i.e. functions for exchanging files or parts of it. There are several approaches focusing on resource mediation mechanisms. They vary from centralized concepts such as index servers, as in eDonkey, to highly decentralized approaches such as flooding protocols, as in the Gnutella network, or distributed hash tables, as used in the Chord protocol. Especially, the DHTs and hierarchical derivatives have gained a lot of scientific interest addressing refinements to cope with reliability and efficiency in cellular environments [122]. Special architectural entities like crawlers are used to locate files and sources of files on behalf of other users to improve the performance. This is especially important in mobile environments with scarce and expensive resources of users, cf. [11].

The resource access control mechanisms determine the coordination and cooperation among peers which means to permit, prioritize, and schedule the access to shared resources. In this context, incentive mechanisms are implemented to promote cooperative behavior. This means they try to make peers participate in the network and share their resources. Examples are credit point systems as used in eDonkey or tit-for-tat strategies like in BitTorrent. However, in this chapter, we consider a different approach, the so-called cooperation strategies, to overcome problems, like the last chunk problem caused by selfishness or inefficient usage of scarce resources in heterogeneous environments. In particular, we investigate different cooperation strategies and derive solutions for specific problems. The coordination of the peers to enable the efficient, fair and robust distribution of contents in a CDN is realized by a cooperation strategy. Its task is to decide (a) which peers requesting for blocks are served by an uploading peer using a priority function, like first-come-first-serve, and (b) which is the next chunk to download by a downloading peer. These two decisions undertaken by a cooperation strategy are referred to as peer selection and chunk selection, respectively. The question arises whether a cooperation strategy can leverage the effects of selfishness or heterogeneity and establish an efficient, fair and robust CDN.

### 2.1.1 Content Distribution with Multi-Source Download

An efficient and robust way of cooperative content delivery is the multi-source download (MSD), which means that the recipient peer orders and downloads the desired data from many providing peers instead from a single one. The efficiency of MSD was demonstrated by the success of the P2P files sharing platforms eDonkey and BitTorrent and was scientifically researched e.g. in [99, 103].

P2P content distribution mechanisms which apply MSD split files into chunks and blocks which are subparts of chunks. For the eDonkey application for example, the chunk size is typically 9.5 MB and the block size is 180 kB. A downloading peer requests blocks from serving peers, i.e. sources of that file, and might download from these sources in parallel. As soon as a peer has downloaded a complete chunk, it becomes a source for the file, i.e. it can redistribute the already received chunks. The benefit of MSD lies in the speed-up via the parallel download of data and the faster creation of additional sources for chunks. As a result, MSD does not rely on a single source and can therefore avoid bottlenecks and overcome churn.

A peer can download from an arbitrary number of sources in parallel. While the number of parallel download connections is typically not limited, the number of parallel upload connections at a peer is restricted to a maximum of  $n$  in order to guarantee a certain minimal bandwidth. Requesting peers being served simultaneously share the uploading bandwidth of the providing peer. However, if a downloading peer cannot handle the offered bandwidth due to restrictions of his own download bandwidth, the surplus is equally divided among the other peer connections. In heterogeneous environments, this effect is emphasized, especially due to capacity changes over time due to mobility and VHOs. The resulting bandwidth sharing discipline is referred to as max-min fair share [10].

Reducing the number of parallel uploads to one,  $n = 1$ , which means no parallel uploads at all, could possibly enhance the diffusion. This is reasonable by considering the following scenario. At time  $t_0$ , only a single initial source exists which provides a file consisting of one chunk. All peers are assumed to

have the same upload bandwidth, which allows them to upload one chunk within a time  $T$  using the complete bandwidth. Thus, the number of available chunks after time  $t$  is  $2^{t/T}$  in the case of one upload and  $(k + 1)^{t/kT}$  in the case of  $k$  parallel uploads. It holds  $(k + 1)^{T/k} \leq 2^T$  for  $k > 0$ , i.e., an outbound degree of one performs best in theory. However, these assumptions are not valid in practice. Selfish user behavior, churn, or heterogeneous peer capabilities will lead to other results which will be discussed later.

A user interested in a particular content sends a download request to a peer providing the desired content. If the provider already serves  $n$  peers, it pushes the request into its uplink waiting queue. As soon as an upload connection becomes available, the first peer in the uplink waiting queue is served. However, this waiting queue can be ordered according to a certain priority function. In eDonkey for example, the credit point system is used to determine a peer's position within an uplink queue. This credit point system might take into account the popularity of a file or the actual upload to download ratio of exchanged data with this peer. The simplest priority function is a first-come-first-serve (FCFS) which means the uplink waiting queue is served in FCFS manner. While being served, each peer downloads a specific amount of data in a row. In the current eMule application which is a popular client for eDonkey, these are three blocks of 180 kB, resulting in a so-called download unit (DU) of size 540 kB. After completing the download of a DU, a peer will either re-enter the waiting queue at the end or leave this peer, if it has already finished downloading the desired data. The upload queue model is demonstrated in Figure 2.1. It has to be noted that if a peer goes offline, the existing data connections are dropped, but the already downloaded part of a DU is stored and does not get lost.

In the studies presented in this paper, we assume a hybrid P2P architecture. That means, the information where resources are located is offered by a central entity, which we call the index server in reference to the eDonkey network. The index server keeps track of the peers being connected to the CDN. We focus on the resource access control mechanisms and the sharing behavior of CDNs and therefore assume that global information about chunks shared in the network is

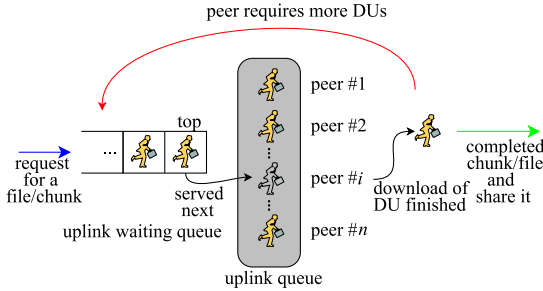


Figure 2.1: Upload queue of a providing peer

available. A peer who is interested in a file, requests all available sources at an index server. Therefore, each peer knows all sources which are connected to the network at the moment of the request. New sources will be discovered by periodical source request messages of a downloading peer which are sent every ten minutes. Every time a peer receives a new source, it sends a download request containing an identifier for all required chunks. If the peer addressed by the request has none of the required chunks, the request is neglected.

### 2.1.2 Common Cooperation Strategies

One of the major influence factors on the performance of a CDN is the applied cooperation strategy. A high level of robustness drives to the success of BitTorrent and is achieved among others by the least-shared first cooperation strategy. In [107], Hamra and Felber identify the principal design choices of content distribution that draw the behavior of the system. In particular, the structure of the P2P overlay and the cooperation strategy are emphasized. According to them, a cooperation strategy is the result of three factors coupled together, the peer selection strategy, the chunk selection strategy, and the network degree.

To define this clearly, a cooperation strategy describes the selection of the next peer being served as well as the choice which chunk should be transferred, i.e. the

Table 2.1: List of requesting peers at seeds and providing peers at the beginning of each round

	$S_1$	$S_2$	$S_3$	$P_0$	$P_1$	$P_9$
no.1	$P_0, P_2, P_7, \dots$	$P_1, P_0, P_3, \dots$	$P_9, P_4, P_0, \dots$	-	-	-
no.2	$P_2, P_7, \dots, P_0$	$P_0, P_3, \dots, P_1$	$P_4, P_0, \dots, P_9$	$P_5, \dots$	$P_6, \dots$	$P_8, \dots$

peer selection and the chunk selection strategy. In this section, we describe two common strategies which are used to identify problems and to compare them as benchmark test with newly proposed cooperation strategies. In particular, we introduce the random chunk strategy and the least-shared first strategy and discuss the chunk dissemination process on an example scenario. For both strategies, the peer selection is assumed to follow a FCFS approach.

As an example, the first two rounds of the distribution process of a file for the two different cooperation strategies are considered. For the sake of simplicity, we consider in this case only a single upload slot of the providing peers and a homogeneous scenario in which all peers require the same amount of time for downloading any chunk.

There are three initial sources  $S_1, S_2, S_3$  which share all chunks of a file. A peer offering all chunks of a file is referred to as a seed for this file. Table 2.1 shows the list of requesting peers at the seeds and the providing peers at the beginning of the first two rounds. At seed  $S_2$ , for example, the requesting peers are  $P_1, P_2, P_3, \dots$  which will be served in FCFS manner order. In the beginning of round no. 1, only the seeds share chunks. After that round, however, the peers which were served by these peers also act as sharing peers and provide the successfully downloaded chunks to the other requesting peers.

In the example the file consists of two chunks. Furthermore there are ten peers  $P_0, \dots, P_9$  who want to download the file. The first step of the download process is without loss of generality assumed to be equal for both strategies: peer  $P_0$  downloads chunk 1 from  $S_1$ , peer  $P_1$  chunk 2 from  $S_2$ , and peer  $P_9$  chunk 1 from  $S_3$ . After the first round of transferring chunks, the dissemination behaves different for each strategy. The strategies will be explained in the following.

Table 2.2: Example of chunk download for common cooperation strategies

	$P_0$	$P_1$	$P_2$	$P_3$	$P_4$	$P_5$	$P_6$	$P_7$	$P_8$	$P_9$	$\#C_1$	$\#C_2$
no.1	$C_1/S_1$	$C_2/S_2$								$C_1/S_3$	5	4
no.2: random	$C_2/S_2$		$C_1/S_1$		$C_1/S_3$	$C_1/P_0$	$C_2/P_1$		$C_1/P_9$		9	6
no.2: LSF	$C_2/S_2$		$C_2/S_1$		$C_2/S_3$	$C_1/P_0$	$C_2/P_1$		$C_1/P_9$		7	8

Table 2.2 illustrated the chunk exchange for the depicted example. The columns  $P_0, \dots, P_9$  shows the file requesting peers and their corresponding actions per round for both strategies. For example, in round 1, peer  $P_0$  downloads chunk 1 from seed  $S_1$ , indicated as  $C_1/S_1$ . Round no. 1 is equal for both strategies and results into five sharing peers of chunk 1 ( $S_1, S_2, S_3, P_0, P_9$ ) and four sharing peers of chunk 2 ( $S_1, S_2, S_3, P_1$ ), cf. Table 2.2. The number of sharing peers of chunk  $i$  is abbreviated as  $\#C_i$  in Table 2.2.

### eDonkey-like Random Chunk Strategy

Applying the random chunk strategy, like the one used by eDonkey, a downloading peer issues a request to a sharing peer. The sharing peer queues this request in a first-come-first-serve (FCFS) manner. As soon as the downloading peer is served, it chooses a random chunk which it has not downloaded yet. In our example, peer  $P_0$  selects its missing part and departs after downloading it from the network. In addition, peer  $P_2$  and peer  $P_4$  choose chunk 1 randomly and independently and download it from  $S_1$  and  $S_3$ , respectively, cf. Table 2.2.

The random chunk strategy relies on the random selection of required chunks. The randomization avoids that all downloading peers select the same chunk. Thus, the simultaneously downloading peers get different chunks and can therefore exchange these different chunks in the further distribution process. This fosters the cooperation among peers. As will be shown in Section 2.2 this strategy performs well as long as peers are altruistic, i.e. as long as peers are willing to share after they have completed their download.

However, if most of the peers are leeching and leave the system shortly after the download of the file or due to churn, the random selection cannot guarantee an even distribution of the chunks. This leads to the situation of one chunk being less shared than the others and the last chunk problem occurs. In our example, chunk 2 is only shared by three peers besides the initial seeds, while chunk 1 is shared by six peers. If a peer sharing chunk 2 leaves the system for any reason, this imbalance is increased further.

### **BitTorrent-like Least-Shared First Strategy**

The least-shared-first (LSF) strategy also uses the same priority function for the peer selection like the random chunk strategy, i.e. requests are served in a first-come-first-serve manner. However, the chunk selection differs. Peers choose as next chunk to be downloaded the one which is least-shared in the P2P network. This means that this chunk has the smallest number of sharing peers, compared to the number of possible sources for other chunks. If there are several chunks fulfilling the least-shared criteria, one of these is chosen randomly. After round no. 1 of the example scenario, chunk 2 is the least shared one. Thus, with the same peers to be served as for the random chunk strategy, the peers  $P_2$  and  $P_4$  choose the least shared chunk 2 not yet being downloaded at the moment of the download. At the end of round no. 2, the least-shared first strategy results in a more equal chunk distribution, that are seven sharing peers of chunk 1 and eight sharing peers of chunk 2 which can also be seen in Table 2.2.

A peer using this strategy selects the required chunk which has the lowest number of providing peers. This mechanism results typically in an evenly spread number of sharing peers for all chunks of the file. However, there are cases in which this is not true. As it will be shown in Section 2.2, this strategy is very efficient as long as the chosen chunk is the least-shared one at the end of the download of this chunk. However, the decision which chunk is the currently least-shared one is done at the beginning of the download. Thus, another chunk can get the least-shared one which undermines the homogeneous chunk dissemination.



In addition, it is necessary that every peer is aware of the numbers of peers sharing a specific chunk in order to know the least-shared chunk. Although, we assume that an index server, as used by eDonkey, keeps track of the peers being connected to the CDN, the index server is not responsible for providing information about the dissemination of chunks. Thus, the evaluation presented later neglects the overhead caused by frequent status update messages or monitoring mechanisms which are necessary to maintain or predict this information. Hence, the LSF strategy might perform worse in practice, since the transmission of the overhead consumes additional resources. As a result, the download time might be longer than discussed here.

### **2.1.3 Key Performance Characteristics**

The performance of a P2P CDN is determined by the implementation of the cooperation strategy, the peer characteristics, i.e. the currently available capacity resources for exchanging files (upload and download bandwidth, maximum number of inbound and outbound connections), and the user behavior. The latter one includes (a) the file request pattern taking into account flash crowd effects and popularity of contents; (b) the churn behavior, i.e. the switching of an user between offline and online state which might be more frequent in mobile environments; (c) mobility which mainly effects the available capacities of a peer; and (d) the willingness to participate in the network, i.e. selfish or altruistic peers. As an extreme case of selfish peers we consider leechers which immediately leave the system after finishing the download of a file. In such a case, the leaving users reduce the availability of chunks. As a result, in the worst case a specific chunk may get rarely in the system.

From the user's perspective, the key performance characteristic of a CDN is the efficiency in terms of download time which is the time from sending the request for a file until successfully receiving the entire content. Additionally, a user wants to minimize its costs in terms of the amount of uploaded data volume which consumes an expensive resource in cellular networks, the upload capacity

of a peer. Beside the efficiency and the costs, in a P2P-based CDN the users are interested in a fair system, i.e., the system should ensure fairness among the peers with respect to efficiency and costs. This is especially important in the presence of selfish peers. In particular, a perfectly fair cooperation strategy makes all peers experience the same download time and upload the same amount of data, although selfish peers try to maximize only their own benefit. We choose the fairness index introduced by Jain [53] to quantify fairness.

Jain's fairness index is defined by

$$J = \frac{(\sum_{i \in M} x_i)^2}{|M| \sum_{i \in M} x_i^2}, \quad (2.1)$$

where  $x_i$  are the values of the considered performance measures,  $M$  is the set of all measurement values, and  $|M|$  is the number of measurement values. It holds  $J = \frac{1}{1+c_x^2}$ , where  $c_x$  is the corresponding coefficient of variance. The fairness index returns values between zero and one, i.e.  $0 \leq J \leq 1$ . Low values of the fairness index indicate an unfair system, while a fairness index of one describe a completely fair system. That is, all users experience deterministically the same performance with respect to the considered measure.

From a global point of view, the robustness of a CDN is of interest which is expressed by the chunk availability and the occurrence of rare chunks. More formally, we define the availability  $A_i$  of a chunk  $i$  in the time interval from  $t_0$  to  $t_1$  as follows

$$A_i = \frac{\int_{t_0}^{t_1} C_i(t) dt}{t_1 - t_0}, \quad (2.2)$$

where  $C_i(t)$  is the number of peers sharing chunk  $i$  at time  $t$ . The chunk availability  $A_i$  reflects the average number of peers sharing chunk  $i$  in the corresponding interval. The rare chunk availability  $A$  is the minimal availability of all chunks normalized by the average availability of all chunks. A low value of  $A$  indicates starving chunks, while a high value around 100 % shows that all chunks are similarly disseminated and available over time within the CDN. Let  $N$  denote the

total number of chunks of a file. Then we define the rare chunk availability as

$$A = \min_{0 \leq i < N} \left\{ \frac{A_i}{\frac{1}{N} \sum_{0 \leq j < N} A_j} \right\}, \quad (2.3)$$

which can take values in  $[0; 1]$ .

According to Birolini [154], robustness is a characteristic of a system, being stable under failure, misuse, and overload. For content distribution networks we see this demand fulfilled if the system is resistant against changes in the user behavior, i.e., the file transfer times and upload volumes are stable even with selfish peers in the network. Hence, a CDN which is efficient, fair and robust can provide a reliable download experience with short download times and small upload volumes. To be more detailed, a cooperation strategy is considered to be robust, if the amount of data uploaded and the time needed to finish the download of an arbitrary peer are close to the values obtained in a diffusion scenario with altruistic peers which will be explained in Section 2.2.3. This implicitly requires a high chunk availability for all chunks.

## 2.1.4 Related Work on Cooperation Strategies

Cooperation strategies define how peers interact with each other. Penserini et al. [85] model peers within a special framework and research methods how to judge the cooperation strategies build up by the reasoning mechanism within the peers for a given task.

If we focus on content distribution the task is to quickly disseminate one or more files to a group of peers. Incentives help these groups to collaborate even if some of the peers behave selfish. In [82], Lai et al. characterize the problem of selfish peers and shown that solutions based on the local knowledge on a peer's behavior does not scale with an increasing peer group size. Thus, other options have to be considered. But it has also been shown in [96] that there are possibilities to reach near optimal sharing behavior even in large groups and with

high churn using incentives. A comparison between different incentive strategies is presented in [111]. Many of these incentive mechanisms are based on the idea of trading upload volume against download volume by using some sort of virtual currency. However, if a new peer without any part of the file enters the system, it has to earn an amount of this currency in order to pay for its download. To encounter this problem Liao, Papadopoulos, and Psounis [135] propose to reward peers for staying in the system instead of endowing new peers the possibility to download parts of the file. In contrast to this, Anagnostakis and Greenwald [87] believe that incentives based on virtual currencies are either ineffective or much too complex. Therefore they propose a strong incentive, based on the idea of barter trade. In their proposed architecture peers prefer to trade parts of files with other peers, which provide them with parts they currently need and vice versa. Incentives might guarantee a good cooperation between peers. But that does not necessarily mean that the exchange of data is fair for all peers, as it is demonstrated by Veciana and Yang [72]. However, all these approaches define incentives in order to stabilize the cooperation of peers. In our work we propose an interaction scheme without incentives and compare it to some of the architectures proposed above. Another proposal for an incentive-less architecture is defined by Hales [131]. But in this work Hales assumes that peers are able to copy the neighborhood and the behavior of other peers, which is very hard to achieve in practice and is not necessary with our approach.

In 2004, Fessant, Handurukande, Kermarrec and Massoulié [97] showed measurement results of several peer-to-peer content distribution systems and concluded that these systems provide the opportunity to gain efficiency by clustering peers with the same interests and regional togetherness. The idea of selecting proper peers in order to increase the efficiency was also discussed in [77]. This contribution proposes to build hierarchical structures in order to cope with problems locally and not to affect the whole network. [74] discusses how a measurement-based optimization may influence bandwidth demanding peer-to-peer systems. The question is addressed in [137] which topologies are created by peers trying to minimize their connections and optimize the response times to

their overlay neighbors. The peer selection may also have an influence and can optimize the dissemination of a file in the P2P system based on random selection of parts of the file which is shown in [138]. In our contribution we do not restrict peers in their communication with other peers. A peer may interact with any other peer in the system. Thus, we focus on the timing when two peers interact and on the information they exchange, i.e. the scheduling within the resource access control and the chunk selection.

With the optimal selection of neighbors in the overlay it may be possible to structure and optimize the peer-to-peer network. However, the data exchanged between two interacting peers also influences the file dissemination. Felber and Biersack [95] discuss which peer and chunk selection strategies are able to cope with flash crowd effects. A more detailed look on the behavior of the content distribution systems which we compare with our solution are presented in [104, 134]. Whereas Tutschku [104] focuses on the eDonkey network, Legout et al. [134] regard the BitTorrent architecture.

Beyond these performance measures it is also crucial that the CDN does not decay in adverse circumstances. This feature is called robustness and is discussed in [86, 140]. While Risson and Moors [140] research the robustness of algorithms that distribute dictionaries over a group of peers, Triantafillou et al. [86] apply the concept of robustness to P2P CDNs. The resulting content distribution system is complex and uses peer clustering. In contrast to this, our proposed architecture is flat and avoids the overhead needed to stabilize a hierarchical architecture.

Despite the large literature on content distribution schemes, there exist only a few works on P2P CDNs in a mobile environment, especially in infrastructure-based wireless networks. Recently, mobile P2P research projects have received high attraction which is reflected by the popularity of the latest IEEE workshops MobiShare and MP2P. However, most of the work addresses structured P2P networks based on distributed hash tables as lookup-service or considers mobile ad hoc networks. For example, Michiardi and Urvoy-Keller [152] propose a cooperative P2P scheme that allows parallel download of the content based on swarming protocols in wireless ad hoc networks.

In the context of infrastructure-based, cellular networks, some investigations on P2P-based content distribution exist. Biström and Partanen [92] propose a JXTA solution to create a mobile file sharing system in 3G environment. The effect of heterogeneous, but fixed link capacities in BitTorrent-like file sharing systems was analytically evaluated with a simple fluid model [102]. It is shown that bandwidth heterogeneity can have a positive effect on content propagation among peers. The cooperation concept proposed in [128] makes peers help each other in downloading data. In [151], the authors propose a network-aware P2P file architecture and related control schemes in cellular systems which divide a P2P file sharing network into multiple network-aware clusters. A file discovery control scheme named Mobility-Aware File Discovery Control (MAFDC) scheme is devised to obtain fresh status of shared peers and find the new resource providing peers in wireless mobile networks. Additionally, a resource provider selection algorithm is devised to enable a mobile peer to select new resource providing peers for continuous file retrieval. However, these strategies do not take into account the effects of mobility and VHO in a heterogeneous, cellular environment.

## 2.2 Selfishness of Users and Robustness of the System

The scope of this section is to show the impact of selfish users on the performance of the system. In particular, we will show that the selfishness of users will decrease the robustness of the system which will lead to the last chunk problem. We will further address if an appropriate cooperation strategy is able to deal with selfish user behavior and makes the system be robust again. As a result of the observation why common cooperation strategies fail to prevent chunks starving in the network, we develop the so-called CycPriM strategy. In a worst-case and a best-case scenario, we compare the performance of the CycPriM strategy with the eDonkey-like random chunk selection strategy and the BitTorrent-like least-shared first (LSF) strategy.

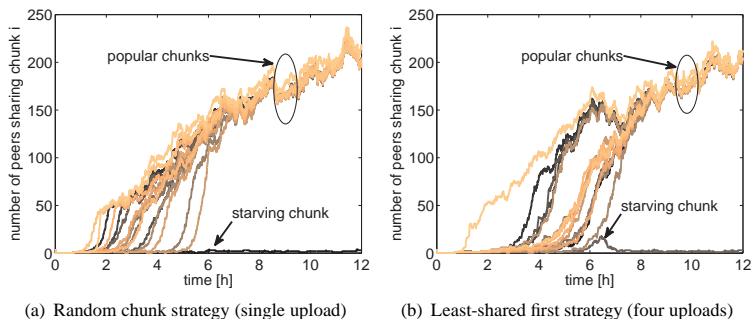


Figure 2.2: Illustration of the last chunk problem in the leeching scenario

### 2.2.1 Last Chunk Problem

The coordination of peers in a P2P file sharing network is no simple task. An inappropriate coordination of the peers may decrease the performance of the strategy, i.e. it may increase the overall download time of a file. A particular problem in P2P file sharing networks is the so-called “last chunk” or “starving chunk” problem [107]. Here, a single chunk of a file may not spread in the file sharing network as the other chunks do. Hence, a shortage of providers for this chunk may arise. As a result, the remaining providing peers may be overloaded and the file exchange is delayed.

The user behavior now decides on the willingness to participate in the network, i.e. to behave as selfish or altruistic peer. Leechers as an extreme case of selfish peers leave the system immediately after finishing the download of a file. As a leaving user sharing some or all chunks reduces the availability of the corresponding chunks, leechers provoke chunks to get rare. As a result, in the worst-case a specific chunk may get rare in the system, i.e. there are only a few sources for this chunk in the CDN. As a consequence, these few sources might not be able to efficiently serve all requesting peers and the entire content distribution process is disturbed.

Figure 2.2 depicts two examples for such a behavior. It shows the spreading of chunks, i.e. the number of sharing peers for each chunk  $i$  of the file, over time in a leeching scenario. Most of the chunks are spreading throughout the system, cf. label 'popular chunks' in Figure 2.2. One of the chunks denoted as 'starving chunk' will not spread due to the leeching behavior of peers, as the downloading peer disappears from the system as soon as it has downloaded this chunk. The only remaining sources of this chunk are the initial seeds. As a result, unfinished peers which seek the final, last chunk have to wait until they receive the chunk from one of the initial sources. This leads to large download times. This problem is called the last chunk problem.

Figure 2.2(a) depicts the number of peers sharing a chunk throughout time using at most one upload connection. In this case, the random chunk strategy is applied. Each of the chunks of the file is represented by one single line in Figure 2.2(a). It is evident that the number of peers sharing a chunk does not rise equally. At the beginning some chunks are reproduced while others are not. A chunk being shared by some more peers than only the seeders becomes independent to churn which allows a faster reproduction of sources for these chunks. Additionally, in the beginning there are several peers that share only one chunk. Therefore, they will distribute this chunk only, as long as they do not download any other chunks. Thus these chunks will be more often downloaded than the other ones.

After the first chunk was distributed among the requesting peers, a peer downloads another chunk and this chunk spreads in the network like the first one. Later this leads to a situation in which nearly all chunks are often shared. This is the point where the leeching behavior harms the system. If a peer downloads this starving chunk there are two possible situations. In the first case, it has the other chunks already. Thus this peer has finished the download and departs from the CDN. In the second case some or all of the other chunks are still needed. Then the peer will be able to download the remaining chunks in a short time, because of the high number of peers sharing the other parts. Afterwards, it leaves the network. In any case, the time this peer provides this rare chunk as an additional source is



very short. As a result, one chunk is shared only by a few peers and required by many peers which forces them to wait for this final chunk to be transferred.

The LSF strategy tries to overcome the last chunk problem by favoring rare chunks. In order to choose the least-shared chunk, it is imperative to know the dissemination of chunks at the moment of the chunk request. Thus, this information has to be up-to-date and globally accessible, e.g., it is provided by a tracker or other more complex distributed schemes.

In Figure 2.2(b), we see the evolution of the number of peers sharing chunk  $i$  for the least-shared first strategy with at most four parallel uploads. In this situation, this cooperation strategy is no longer able to prevent a starving chunk. The reason is that the least-shared chunk is determined at the beginning of the download. However, this is not necessarily the least-shared one when the download ends. With a rising number of parallel uploads it gets more difficult to decide the least-shared chunk at the end of the download before it starts.

The question arises whether a cooperation strategy can leverage the selfish behavior of the peers and avoids the last chunk problem. Within this chapter, different cooperation strategies are evaluated with respect to the last chunk problem by their download performance and their spreading behavior of the chunks. There are many attempts to overcome this problem like the least-shared-first chunk selection of BitTorrent [71] or Avalanche network coding [115]. In this chapter we propose, however, a new cooperation strategy called CycPriM, which has efficient chunk diffusion behavior and unlike other strategies it is based only on existing local information available at the peer.

### **2.2.2 CycPriM Strategy**

A sharing peer should take care of the homogeneous chunk dissemination in the network to avoid the last chunk problem. As we have seen so far, the random distribution of chunks leads to rare chunks in the presence of selfish peers or high churn rates. Although the least-shared first strategy tries to overcome this, it still cannot avoid that chunks get rare in the system. The main problem derives from

the fact that the downloading peers determine which chunk to download next. As selfish peers are only interested in their own download and not in the robustness of the CDN itself, any chunk selection undertaken by the downloading peers to their benefit cannot solve the last chunk problem if the peers are served in a first-come-first-serve manner. As the downloading peer determines which chunk is required and will be downloaded next, our idea is to modify the peer selection strategy of an uploading peer in an appropriate way. The goal of the peer selection strategy is to force the downloading peers to download chunks such that an equal dissemination of all chunk of a file prevails. Thus, the chunk selection strategy of the downloading peer is implicitly determined by the peer selection mechanism of the uploading peer.

If there is only local information available at a providing peer on the availability of chunks in the CDN, the sharing peer should deliver chunks in an ordered way. The basic idea is to distribute the entire file - instead of favoring individual chunks - in upload rounds. In each upload round the mechanisms tries to distribute a sequence of all chunks to requesting peers, as long as requests for these chunks are available. If no request is available for one of the chunks in the sequence, this chunk is skipped and the next chunk of the sequence is chosen to be distributed. After the complete sequence is processed, a new upload round starts. In order to prevent the downloading peer from selecting any other chunk, we propose the following cooperation strategy: The uploading peer offers only this one chunk. If a peer accepts this offer, or no peer wants the chunk, then the next chunk from the cycle is chosen. We call this strategy CycPriM which stands for Cyclic Priority Masking.

It has to be noted that this cooperation strategy does not require any additional information from the CDN. In contrast to the least-shared first strategy, the chunk availability has not to be monitored and signaled to providing peers and seeds. Thus, no additional signaling traffic arises. Each providing peer only has to decide its individual sequence of chunks. According to this sequence, the upload of chunks is determined. However, no coordination among the peers is required to define this sequence. In fact, we use a random sequence of chunk delivery for

Table 2.3: Example of chunk upload for CycPriM and common strategies

	$S_1$	$S_2$	$S_3$	$P_0$	$P_1$	$P_9$	$\#C_1$	$\#C_2$
no.1	$C_1/P_0$	$C_2/P_1$	$C_1/P_9$				5	4
no.2: random	$C_1/P_2$	$C_2/P_0$	$C_1/P_4$	$C_1/P_5$	$C_2/P_6$	$C_1/P_8$	9	6
no.2: LSF	$C_2/P_2$	$C_2/P_0$	$C_2/P_4$	$C_1/P_5$	$C_2/P_6$	$C_1/P_8$	7	8
no.2: CycPriM	$C_2/P_2$	$C_1/P_3^{(1)}$	$C_2/P_4$	$C_1/P_5$	$C_2/P_6$	$C_1/P_8$	8	7

each providing peer. This sequence is locally stored at the providing peer and is kept constant while uploading chunks of this file.

We consider now the same example as in Section 2.1.2. The file consists of two chunks and the chunk upload sequences for each seed are the following. Seed  $S_1$  and  $S_3$  first upload chunk 1 and then chunk 2, while seed  $S_2$  first uploads chunk 2 and then chunk 1, i.e.  $(C_1, C_2)$  for  $S_1$ ;  $(C_2, C_1)$  for  $S_2$ ; and  $(C_1, C_2)$  for  $S_3$ .

The first round of the download process is the same as for the random chunk strategy and the least-shared first strategy. Peer  $P_0$  downloads chunk 1 from  $S_1$ , peer  $P_1$  chunk 2 from  $S_2$ , and peer  $P_9$  chunk 1 from  $S_3$ . After that, however, the CycPriM strategy leads to a different system behavior. The ordered list of requesting peers at seeds and providing peers at the beginning of each round is given in Table 2.2.

Table 2.3 shows which chunk the seeds and the providing peers upload in the two rounds for the random chunk strategy (random), the least-shared first strategy (LSF), and the cyclic priority masking strategy (CycPriM). The number of sharing peers of chunk 1 and chunk 2 is denoted as  $\#C_1$  and  $\#C_2$ , respectively. Note that Table 2.3 shows now the chunk dissemination process from the viewpoint of an uploading peer, in contrast to Table 2.2 showing the download of chunks from the viewpoint of requesting peers. This representation highlights the different peer selection of the CycPriM strategy compared to the FCFS peer selection of the random and the LSF strategy.

<sup>1</sup>Peer  $P_0$  is masked because it already has chunk 1. Thus, the next peer  $P_3$  is served instead.

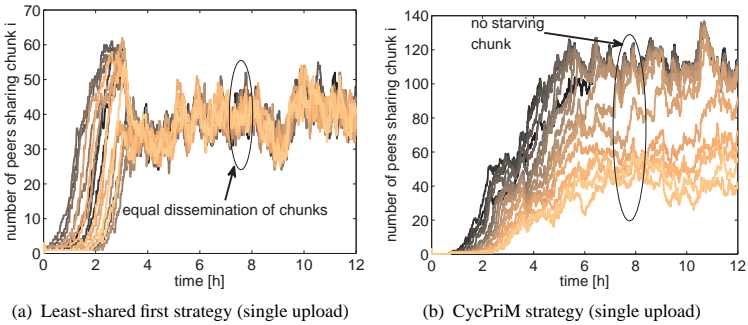


Figure 2.3: Avoidance of starving chunks by equal dissemination and ordered delivery in a leeching scenario with selfish user behavior

In Table 2.3 the example shows, that all seeds share the opposite chunk in the second transfer phase for the CycPriM strategy, as they did in the first. For example seed  $S_2$  has transferred chunk 2 in step 1, thus  $S_2$  will distribute chunk 1 in step 2. Peer  $P_0$  would be served by seed  $S_2$  in the second step. However, peer  $P_0$  has been masked because it already has chunk 1. The next peer wanting to download chunk 1, which is in the example peer  $P_3$ , is served instead.

Figure 2.3 shows the temporal evolution of the number of sharing peers for each chunk for a single simulation run. The considered simulation scenario is the same as described in Section 2.2.1. Thus, we consider a leeching scenario in which the users disappear immediately after downloading a file. Figure 2.3(a) shows the results for the LSF strategy with a single upload per providing peer, while the results for the CycPriM strategy with a single upload is illustrated in Figure 2.3(b). Obviously, in both cases, the cooperation strategy avoids starving chunks which is realized by an equal dissemination of chunks for LSF and the ordered delivery of chunks for CycPriM, respectively. As it will be shown later CycPriM is only a little bit slower than an optimal adjusted LSF. But it neither needs additional signaling traffic nor has the last chunk problem which appears

when using several upload slots for LSF. This means the CycPriM will also lead to a robust system, if the number of upload slots differs and the peers have heterogeneous peer capabilities. Figure 2.3(b) shows in particular that using CycPriM the availability for the different chunks stays relatively close together in the beginning. After this, they spread in the CDN and the number of peers sharing a particular chunk is very dynamic. The most popular chunk is three times more often shared than the most unpopular one which increases slightly the download times.

### 2.2.3 Investigated Scenarios of User Behavior

The performance of the cooperation strategies is evaluated for different scenarios in which the user behavior and the peer capabilities are varied. In particular, we investigate the impact of selfish and altruistic peers, as well as the impact of a single upload and four parallel uploads per peer.

The selfishness of peers is investigated in a worst-case scenario, the leeching scenario, and a best-case scenario, the diffusion scenario, in which the peers are almost altruistic. In the diffusion scenario all peers finishing the file transfer will serve as uploading peers during the rest of the simulation. From the diffusion scenario it can be concluded whether a strategy uses the available resources efficiently or not. Against this, a peer finishing the download will depart from the network shortly after in the second scenario, which is called the leeching scenario. The selfishness in the leeching scenario will demonstrate if a strategy can deal with uncooperative peer behavior.

Another influence factor on the system performance are the peer capabilities. In all scenarios of this section, peers are assumed to have the same bandwidth capabilities. The impact of heterogeneous and changing bandwidths is considered afterwards in the Section 2.3. Here, the peers have GPRS access with an upload bandwidth of 12 kbps and a download bandwidth of 48 kbps. The maximum number of outbound connections, i.e. parallel uploads of a peer, might strongly impact the system and is varied between one and four connections. These settings

are used in this section as a case study to see how the cooperation strategy impacts the performance and how the sophisticated CycPriM strategy improves the system. This mobile P2P CDN scenario is of particular interest to investigate the robustness of a system, due to the increased churn behavior of peers and the poor connectivity of the peers. This may increase the selfishness of peers and clearly reveals the drawbacks of the cooperation strategies.

The considered network consists of 1,000 peers. These peers are interested in one file which is provided by three initial seeds. The file has a size of 8 MB which corresponds roughly to the median size of a YouTube video [43]. At the beginning one peer is chosen randomly to download the file. The interarrival time used to schedule the file requests of the other peers follows an exponential distribution with a mean of 80 seconds.

Although user mobility has an impact on the capacity of the cellular system, the effect of mobility for a P2P user with fixed access bandwidth in a cellular network can be described by a simple ON/OFF process. If the user is granted access to the wireless system, he may start its P2P application. Due to the loss of radio coverage, the peer appears to be offline in the P2P system. Thus, we may subsume (i) the online and offline behavior of a peer due to switching the P2P application on or off and (ii) the effect of mobility of a peer with a fixed bandwidth in a cellular network. In the following, we use the term “churn” which is described by a random variable taking (i) and (ii) into account.

In all scenarios we assume churn. The duration of an ON period and an OFF period of a peer is exponentially distributed with a mean of one hour, respectively. For illustrating the dynamics of the system, we take a closer look at the churn ratio. We define the churn ratio  $\gamma$  as the ratio between the online time  $T_{on}$  and the offline time  $T_{off}$  of an arbitrary peer, formally  $\gamma = \frac{T_{on}}{T_{off}}$ .

In our simulations,  $T_{on}$  and  $T_{off}$  follow an exponential distribution with an average online respective offline time of 1 h. As a result,  $\gamma$  is a random variable

whose cumulative distribution function can be derived as follows

$$P \left[ \frac{T_{on}}{T_{off}} \leq t \right] = \int_{a=0}^{\infty} P \left[ \frac{T_{on}}{a} \leq t \right] P [T_{off} = a] da \quad (2.4)$$

$$= \int_{a=0}^{\infty} (1 - e^{-\lambda a t}) \lambda e^{-\lambda a} da = \frac{t}{1+t} . \quad (2.5)$$

The probability density function of the churn ratio  $\gamma$  is accordingly

$$P [\gamma = t] = \frac{d}{dt} P [\gamma \leq t] = \frac{1}{(1+t)^2} . \quad (2.6)$$

Figure 2.4(a) shows the cumulative distribution function (CDF) of the churn ratio  $\gamma$  for the parameters as used in the simulation scenarios, while the probability density function (PDF) is given in Figure 2.4(b). As can be seen from these results, the dynamics of the peers in the system due to churn might be quite high. This assumption is reasonable because of the considered mobility of the peers.

The dynamics of the system can also be identified when deriving the probabil-

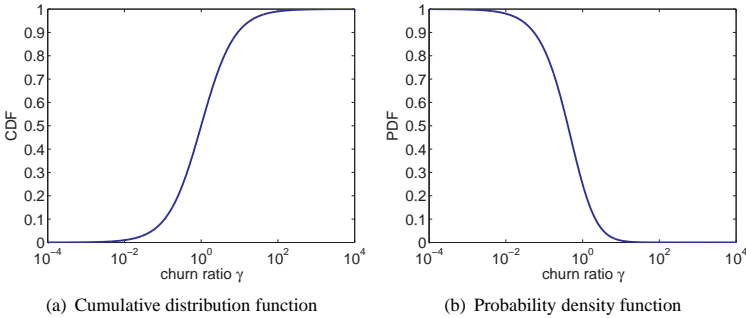


Figure 2.4: Churn ratio  $\gamma$  as used for the comparison of CycPriM with common cooperation strategies

ity that a peer is online in two successive on and off phases, i.e.  $P \left[ \frac{T_{on}}{T_{on} + T_{off}} \leq t \right]$ . In an analogous way, we obtain

$$\begin{aligned} P \left[ \frac{T_{on}}{T_{on} + T_{off}} \leq t \right] &= \int_{a=0}^{\infty} P \left[ \frac{a}{a + T_{off}} \leq t \right] P [T_{on} = a] da \\ &= \int_{a=0}^{\infty} e^{-\lambda a/t - \lambda a} \lambda e^{-\lambda a} da = t, \end{aligned} \quad (2.7)$$

and  $\frac{d}{dt} P \left[ \frac{T_{on}}{T_{on} + T_{off}} \leq t \right] = 1$  as probability density function.

For the numerical evaluation, we simulated ten runs for each scenario and each cooperation strategy. Confidence intervals are omitted for reasons of clarity, when presenting the simulation results in Section 2.2.4. We have indicated in [30] that the confidence intervals are small enough to separate the performance results of the cooperation strategies from each other. Thus, the derived qualitative statements and results regarding the application of cooperation strategies in different scenarios are valid, even when taking significance levels of the simulation results into account. In order to show the statistical credibility, we nevertheless take a closer look on some exemplary scenarios at the end of Section 2.2.4.

## 2.2.4 Performance Comparisons of CycPriM with Common Strategies

Next, we investigate the performance of these cooperation strategies in the diffusion and the leeching scenario. The diffusion scenario represents an ideal system. The scenario is used as a reference scenario for the discussion of the leeching scenario where robustness and fairness is of major interest. The last chunk problem and the associated decrease of efficiency are assumed to be mainly caused by the selfish behavior of peers. Therefore, the robustness and availability of chunks is investigated in the leeching scenario. It has to be noted that the results for the diffusion, as well as for the leeching scenario are combined in Figure 2.5 and Figure 2.6. Next, we start discussing the diffusion scenario.



**Diffusion Scenario**

Figure 2.5 shows the results of the simulation study from the user’s point of view including the uploaded data volume as well as the download times experienced by the user. The left part of Figure 2.5(b) depicts the average download time in the diffusion scenario with the associated 95 % and 99 % quantiles. The number of parallel uploads in the scenario is varied from one to four and is labeled as ‘#PU’ in Figure 2.5, as well as in Figure 2.6. The average download times are in the same order of magnitude for the different cooperation strategies in the diffusion scenario. They show that all cooperation strategies are very efficient and permit a short download time. However, the download time of an arbitrary peer depends highly on the actual number of available sources and their upload bandwidth. Due to the altruistic behavior, a peer that starts the download lately sees many sources for the file where it can choose from. Hence, a peer arriving lately experiences a short download time and has to upload less data.

Figure 2.5(a) shows the average data volume uploaded by peers and the corresponding 95 % and 99 % quantiles. We see, that in the diffusion scenario the

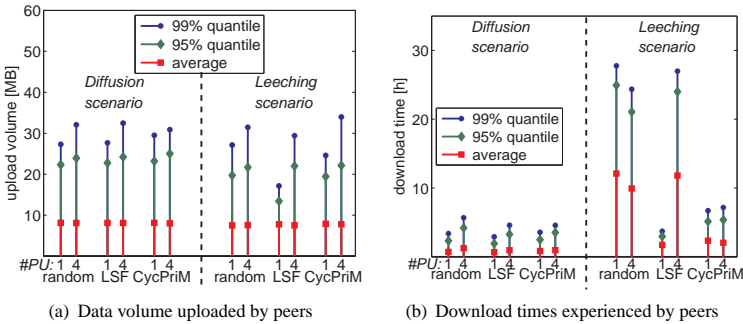


Figure 2.5: Comparison of CycPriM with random chunk and LSF strategy

amount of uploaded data volume significantly differs among the peers which is independent of the cooperation strategy. This is also expressed by Jain's fairness index in Figure 2.6(b) which is around 0.5. The indices are all in the same order, independently of the observed performance measure, i.e. the upload volume or the download time, and the number of parallel uploads. The download time is closely related to the availability of chunks. The more peers provide a chunk the faster a download can be completed. The left part in Figure 2.6(a) shows the rare chunk availability in the diffusion scenario. It nearly reaches the optimal value of 100 % for all strategies due to the altruistic user behavior.

### Leeching Scenario

We continue to investigate the leeching scenario which is depicted in the right part of Figure 2.5 and Figure 2.6. A cooperation strategy is considered to be robust, if the amount of data uploaded and the time needed to finish the download are close to the values obtained in the diffusion scenario. Figure 2.5(a) shows the data volume the peers have uploaded. In all scenarios, the mean value of uploaded data volume is roughly the same. The 95 % quantile and the 99 % quantile show how much individual peers have contributed in uploading. From this figure the high fairness index of the LSF strategy with a single upload becomes evident, cf. Figure 2.6(b). This variant is the only strategy that assures single peers not to upload much more data than two times of the download. All other strategy variants have much higher values for these quantiles.

The right part of Figure 2.5(b) shows the average download time of the peers in the leeching scenario. The LSF strategy with one parallel upload has download times which are in the same order as in the diffusion scenario, cf. left part of Figure 2.5(b). This feature and the low upload volume for each peer demonstrate the very good robustness of the LSF strategy with one parallel upload against leeching behavior. The CycPriM strategy permits fast download times in the leeching scenario as well. It cannot provide always the short download times of the LSF strategy. However, the robustness of the CycPriM strategy does not depend on

the peer capabilities. In contrast, the download times for the random chunk strategy and the LSF strategy with four parallel uploads are three times higher. Only CycPriM is robust against selfish behavior independent of the peer capabilities.

Figure 2.6(a) shows the availability of rare chunks. For the leeching scenario, the last chunk problem occurs at the random chunk strategy and the LSF strategy with a high outband degree. In our simulation study, we found out that already four parallel uploads make one chunk starve in a LSF-based CDN. This is reflected by the low values presented in the right part of Figure 2.6. The CycPriM strategy shows a rare chunk availability of about 50 % which is a result of the wide spectrum of number of sharing peers as already discussed in the CycPriM Strategy section. The LSF strategy with a single upload leads to the best results regarding this rare chunk availability in the leeching scenario. However, in this case, it is necessary to update or estimate the global information about the number of sharing peers for every chunk. Furthermore, the LSF strategy has to prevent changes to the peer capabilities, ie., it must not allow parallel uploads at a providing peer. The CycPriM strategy is robust against leeching as well as against changes of peer capabilities while still avoiding the last chunk problem.

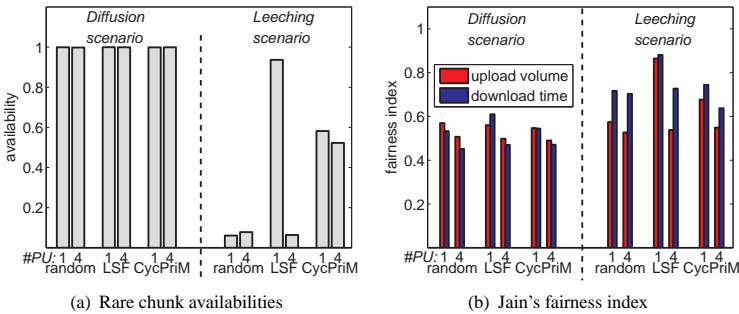


Figure 2.6: Robustness and fairness of cooperation strategies

Figure 2.6(b) visualizes Jain’s fairness index for the upload volume and download times experienced by the peers. The fairness index of all cooperation strategies in the leeching scenario is mostly on the same level, see right part of Figure 2.6(b). Only, the LSF strategy with one parallel upload has a fairness index that is significantly higher. The fairness indices of the other strategies are more or less the same.

As a result of this performance evaluation, we have seen that in cases where most of the peers are selfish, i.e. show a leeching behavior, the performance of the CDN can be significantly improved with an appropriate cooperation strategy. The results proposed so far are also valid in a more general context of P2P-based CDNs, however, the typical features of wireless networks emphasize strongly the effects and the performance influence factors.

### Statistical Credibility of Simulation Results

The statistical credibility of the simulation results presented in the previous section is investigated next. We focus here on the diffusion and leeching scenario with one parallel upload. For the scenarios with four parallel uploads, we obtain

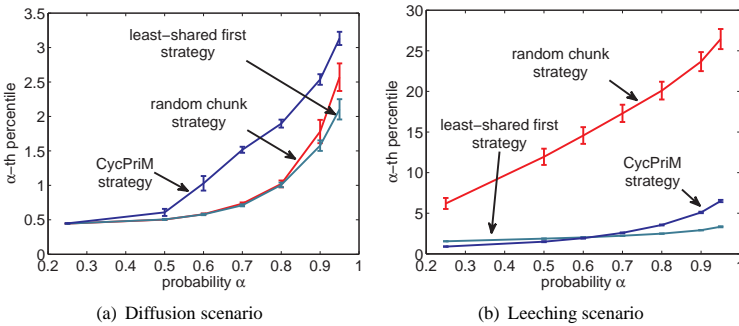


Figure 2.7: Confidence intervals at significance level of 95 % for  $\alpha$ -th percentile

similar results. The investigated performance measure is the download time experienced by a peer in the system. Different performance metrics like rare chunk availability also show similar statistical results. This can be explained, e.g., by the fact that the download time is strongly correlated to the rare chunk availability.

Figure 2.7 shows the confidence intervals at a significance level of 95 % for the  $\alpha$ -th percentile  $x_\alpha$  of the download time  $X$  from ten conducted simulation runs, i.e.  $P[X \leq \alpha] = x_\alpha$ . Although the number of simulation runs is quite low, we see that the confidence intervals are rather small. In particular, the drawn conclusions at the end of the previous section are not affected. For the random chunk strategy, the confidence intervals are larger which is simply caused by the pure random dissemination of chunks in the network and the resulting potential risk of starving chunks. Especially in the leeching scenario, cf. Figure 2.7(b), this can be observed clearly. However, as this strategy leads to much worse results than the LSF and the CycPriM strategy using one parallel upload, this has again no impact on the derived conclusions. Next, we consider the relative error of key performance measures of the download time  $X$ . They include the average value  $E[X]$ , the coefficient of variation  $COEF[X]$ , the minimum download

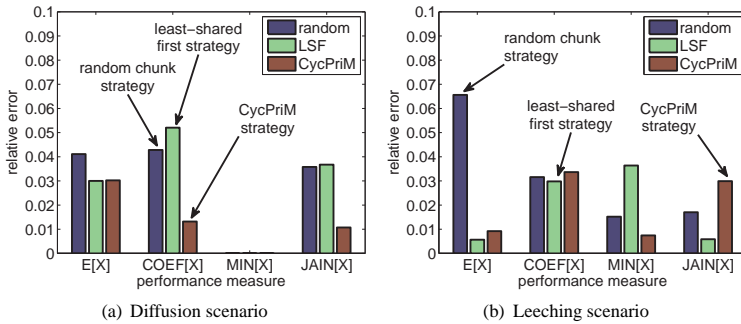


Figure 2.8: Relative error of key performance measures from several runs

time  $\text{MIN}[X]$ , as well as Jain's fairness index  $\text{JAIN}[X]$ , which are plotted in Figure 2.8. The relative error is computed as the length of the confidence interval at a significance level of 95 %, normalized by the average value of the performance measure from the ten individual simulation runs. It has to be noted that the restriction to ten runs has been done because of the excessive computational effort. Nevertheless, we see that the relative error is below 10 % for the considered performance measures in these scenarios and we can conclude that the results are statistically credible for validating our reasoning.

## 2.3 Content Distribution in Heterogeneous Cellular Networks

In this section, we in-depth investigate the impact of vertical handover (VHO) in Beyond 3G (B3G) networks on cooperative content distribution systems. We consider now mobile users moving through a heterogeneous cellular network consisting of WLAN hotspots and UMTS cells. First, we clearly reveal the effects of user mobility on application layer. This allows us to model mobility and VHO for a P2P-based CDN in a cellular network appropriately. In a performance study, we analyze the impact of mobility in different load situations and emphasize the effects of mobility and VHO on the system's performance. Additionally, we consider a today's and a future network layout of the cellular network. In the future network layout, we assume a better WLAN coverage than in today's network layout. The question arises whether the increased capacity due to the higher WLAN density dominates the drawbacks of VHOs on P2P CDNs. As a result of the performance evaluation, we develop new cooperation strategies to cope with the identified problems.

### 2.3.1 Effects of Mobility

A mobile user moving through such a B3G networks needs to perform vertical handovers. This means the ongoing connection is passed from one wireless access system to another and might also include the passing from one operator to another. A VHO implies some delay  $\Delta t_{VHO}$  to reestablish the connection. During this period of time, no application data is transferred. Additionally, the switching between radio access technologies results in an abrupt and dramatic change of the mobile peer's uplink and downlink capacity.

Registering to a new access technology might also change the peer's IP address which leads to the loss of all TCP connections currently opened for file transfer. This concerns the peer's ongoing upload and download connections. But even worse, on application layer, when contacting a providing peer with a new IP ad-

dress, the peer might not keep its old position in the providing peer's waiting queue but reenters at the end of the queue and waits to be served. In addition, a peer  $P$  performing a VHO might serve as a providing peer. The IP address change results in lost connections and the peers served by peer  $P$  need to rediscover  $P$  by asking the index server for new sources of a file. In standard eMule implementation, this is done periodically every ten minutes. In the following, we will refer to this technique as *requeueing w/o refill*.

An alternative method is called *requeueing with refill*. It introduces a minor modification of the peer's cooperation strategy to improve the system's performance and utilizes the fact that a providing peer knows all peers in its uplink waiting queue before and after the VHO. Thus, the providing peer simply reidentifies itself at the served peers with its new IP address and invites them to continue the download. Thus, it can speed up the recovery after a VHO.

Previously, we focused on the situation that a VHO implies an IP address change. However, approaches like Mobile IP preserve the peer's IP address and allow TCP connections to continue after the VHO. These mechanisms lead to an additional delay  $\Delta t_{MIP}$  which we assume to be static. On application layer, a peer keeps its current connections running which means that it also maintains the position in the uplink waiting queue or is still served. However, the total transmission delay during which no application data is exchanged is now  $\Delta t_{VHO} + \Delta t_{MIP}$ . Such a mechanism is denoted as *non-requeueing technique*. The VHO delay can be assumed to be rather small, especially compared to the additional delay caused by the non-requeueing technique, and we will use  $\Delta t_{VHO} = 100$  ms in the simulation studies.

Summarizing, we focus on three effects that VHO have on application layer: abrupt bandwidth change, transmission delay, and change of IP address. In particular, we investigate the impact of requeueing at a providing peer's uplink waiting queue with each VHO, as well as the use of mechanisms that preserve the IP address and connections beyond VHOs, like Mobile IP, at the cost of additional transmission delays.



### 2.3.2 Modeling Mobility and VHO in Cellular Networks

The performance evaluation of a P2P-based CDN with mobile users in a cellular network requires to model the mobility of the users and the above mentioned effects of mobility. In the run-up of our study, we investigated different mobility models, like the random direction mobility model (RDMM) and the Manhattan mobility model (MMM). Such a mobility model is a set of rules which determines the next point on a user's track at each decision point. We found two inadequacies for our simulation purposes. First, the effort for modeling mobility with a "classic" mobility model is not suitable with a large number of simulated users in a discrete event simulation, since it produces a lot of events and thus increases significantly simulation run time. Second, the choice of a particular mobility model is difficult and conveys a lot of following up problems, like the design of a suitable simulation plane for the users which also includes the distribution of coverage areas of the wireless cells and the choice of the corresponding technology.

Therefore, we model mobility in a more abstract way. We propose an abstract mobility model (AMM) which subsumes the network layout and the user mobility by a semi-Markov model. In [32], we showed that the abstract mobility model (with appropriately derived parameters) leads to the same results as the simulation of detailed mobility models, like Random Direction Mobility Model or Manhattan Mobility Model, and detailed network layouts, i.e. simulation of the individual location and coverage of any UMTS node-B and WLAN access point. The simulation of the abstract mobility model is about 50 times faster than the detailed simulation which is required in order to obtain statistically significant data and to be able to investigate a large variety of scenarios and parameters.

#### **Semi-Markov Model**

We now present an approach that releases the discrete event simulation from those events that do not affect the content distribution process at all. This is possible since the mobility of a user is only perceived on application layer when perform-

ing a VHO in the B3G network. Therefore, our approach describes a user's mobility by the user's sojourn time within a certain wireless technology and transition probabilities to other technologies. This abstract mobility model can be modeled as a semi-Markovian finite state machine as defined by Lee and Hou [133] with the wireless technologies as states, general independent sojourn times, and the transition probabilities  $p_{ij}$  for switching from technology  $i$  to technology  $j$ . Note that the transition from one WLAN cell to another is also considered as a VHO, as the WLAN cells are assumed to be operated as individual hotspots.

The distribution of the technology specific sojourn times and the transition probabilities for the AMM are obtained by means of simulation using the RDMM and the MMM with different network layouts, separately. We simulated a single user moving through the simulation plane for 100 days to get statistically significant data. The technical details on the extraction process of sojourn times and probabilities can be found in [148].

The rule set a user has to obey for the AMM still includes decision points, but these are not geographical anymore but merely time dependent. At each decision point in the AMM, a value is chose from the sojourn time distribution function of the current access technology, and then the next access technology is randomly determined according to the transition probabilities.

### **Simulation Scenario Description**

We consider a content distribution system in a heterogeneous wireless environment. In particular, we focus again on the multi-source download mechanism which is based on the eDonkey protocol as implemented in the eMule application and described in Section 2.1. As we focus on the heterogeneity instead of the user behavior in this section, the random chunk strategy is considered now with peers being served in FCFS order. The investigated radio access technologies comprise an area-wide UMTS network and WLAN hotspots which may overlap. The mobile users move in the landscape and perform VHOs between both technologies or between different WLAN cells. In this context, the switch from one

WLAN cell to another is also denoted as VHO, as it might cause an additional delay and the re-assignment of IP addresses.

The UMTS users have a fixed transmission rate of 384 kbps in downlink and 64 kbps in uplink direction. For the WLAN technology, we assign a fixed symmetric bandwidth of 1 Mbps for up- and downlink each. Note, that we do not consider radio resource management mechanisms of the wireless network, like admission, power, or rate control, as we aim at the qualitative evaluation of the effect of VHO on the P2P system. In addition, we do neither consider background traffic in the wireless network nor the case that multiple peers share the capacity of one cell. Including these effects into the simulation would on the one hand lead to unbearable simulation times and on the other hand blur the clear impact of the VHO only. A detailed description of the derived parameters for the abstract mobility model can be found in [148].

The WLAN cells are randomly uniformly distributed within the considered area. We use the disc model with a radius of 50 m to describe the coverage area of a single WLAN cell. In our simulations, we consider a typical city center which is modeled as a square of length 2400 m. According to the investigated scenario, we distinguish between a today's and a future network layout which only differ in the WLAN coverage. In today's network layout, we assume 19 WLAN cells according to the current number of public WLAN cells in Würzburg's city center of a German operator providing UMTS as well as WLAN. In the future network layout, we assume a much better WLAN coverage with 200 WLAN access points.

#### **2.3.3 Impact of Mobility in Today's and Future Networks**

In the investigated scenario, a single file of size 9500 kB is considered. There are 100 mobile peers that want to download this file and altruistically share this file after download. Every 120 seconds, a random peer sends a request to the sources currently available for this file until all peers have placed their request. At the beginning, the P2P network consists of a number of Internet peers with a constant uplink capacity of 768 kbps that serve as initial sources, and keep serving

throughout the simulation. This ensures that the mobile peers always find equal conditions on simulation start-up. The number of these initial peers controls the load of the P2P system. Few Internet peers lead to a high load, since the first downloads may take a long time, and the file only slowly diffuses. All stochastic influences except for the mobility pattern are avoided, so, the impact of VHOs is not tampered by stochastic fluctuations not caused by mobility. For the same reason, we kept to a single set of parameters defining the network and traffic. We performed 20 repetitions with different seeds for the random number generator in every simulation run. For the sake of readability, we omitted confidence intervals and show them only when necessary.

For our analysis, we consider four scenarios: today's network with a low load, today's network with a high load, a future network with a low load, and a future network with a high load. A high load corresponds to a single Internet peer and a low load to ten Internet peers.

In today's network, preserving the IP address outperforms losing the IP address in a high load situation. A peer that loses its IP address is forced to reenter the uplink waiting queues of its sources and therefore has to wait much longer until it is allowed to download for the next time. There is, however, no clear impact of the non-requeueing delay even if the non-requeueing delay is extremely high around 10 seconds, since there are simply too few VHOs in today's network layout. The low load scenario in today's network nullifies the impact of the different IP address handling mechanisms, since even less VHOs occur during the shorter download time in this scenario, and the waiting queues are almost empty. Thus, the average download times are nearly the same. Detailed results and numerical values can be found in [148].

Let us next investigate the situation in future networks with higher WLAN hotspot density. Figure 2.9 show CDFs for requeueing with and w/o refill as well as CDFs for non-requeueing with delays of 0 s, 1 s, 5 s, 10 s, and 100 s in the future network layout. Figure 2.9(a) shows the results for the high load scenario. Analogous to the results in today's network layout, non-requeueing is better than the two requeueing variants, but the difference between requeueing and non-

requeueing increased from a factor of two in today's network layout to a factor of ten in the future network layout. The higher WLAN density in the future layout has two effects, a higher network capacity and more VHOs. The higher available amount of bandwidth leads to an average download time of 82.7 minutes in the future layout compared to 175.6 minutes in today's layout for the non-requeueing technique with  $\Delta t_{MIP} = 1$  s. However, the higher number of VHOs in the future layout increases the relative impact of the non-requeueing delay, compared to  $\Delta t_{MIP} = 0$  s, expressed by larger differences in download times.

Using the requeueing technique, the peer changes its IP address at every VHO. Thus, it is often losing its connections, is removed from being served, and shifted back to the end of the waiting queue. Together with frequent VHOs, this technique has to be avoided for an efficient content distribution service in a future network layout. Only for unrealistic VHO delays of 100 seconds, the requeueing and the non-requeueing technique show the same download performance in a high load scenario as can be seen in Figure 2.9(a).

In the following, we focus on the low load scenario in future networks for which Figure 2.9(b) shows the equivalent CDFs as before. We can still see a dif-

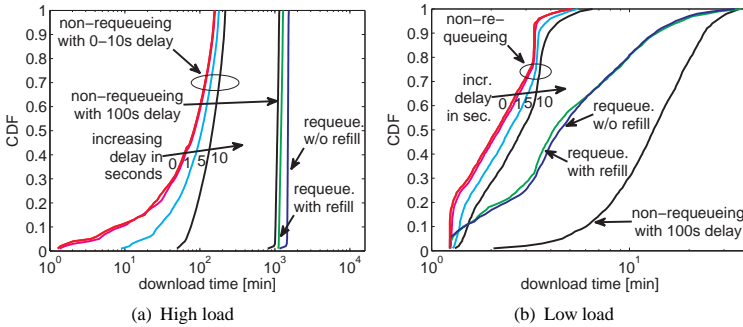


Figure 2.9: Requeueing and non-requeueing technique in future network layout

ference in requeueing and non-requeueing as well as the non-requeueing delays even with a low load as opposed to today's network layout, since more VHOs occur even in the shorter download times. If the load in the P2P system is low, then downloads take less time which leads to less VHOs occurring during the downloading time. In general, the impact of mobility decreases with load and vice versa.

In both load scenarios, preserving the IP address with non-requeueing outperforms requeueing techniques. Nevertheless, the performance gain of non-requeueing melts in the low load scenario, since the waiting queues at the providing peers are almost empty and hence the waiting times are almost negligible. In such a scenario, a delay  $\Delta t_{MIP}$  exists such that the download performance is even worse than with requeueing techniques. However, this only happens for unrealistic large delays.

As a result of the performance evaluation, we see that non-requeueing techniques, like Mobile IP, are recommended in mobile P2P file sharing systems with respect to download performance, if this technique only requires a small transmission delay below a few seconds. In future network layouts, the increased uplink capacity due to the higher WLAN density leads to smaller download times. In order to foster the download from such high-capacity peers, a new cooperation strategy is proposed in the next section which tries to smoothen changes in the available uplink capacity as a consequence of the user's mobility and the resulting VHOs. This means it tries to overcome the drawbacks of heterogeneity.

### 2.3.4 Mastering Mobility with Time-based Data Exchange

In this section, we introduce a new cooperation strategy that affects the duration a user is allowed to access the uplink capacity of a providing peer. In common P2P networks like eDonkey, the resource exchange is volume-based, i.e., each peer is allowed to download the same amount of data in a row, introduced as download unit (DU). We will further speak of volume-based cooperation (VBC). The

problem of VBC is that a peer with a high-capacity technology, like WLAN, is thwarted by peers with smaller bandwidths, like UMTS, if these peers wait to be served by the same source. Thus, a user connected to a high capacity technology cannot finish its download quickly and serve as a new seed for other peers.

As an illustration, we imagine three peers  $P_0, P_1, P_2$  with download capacities  $C_0, C_1, C_2$ . The ratio of the corresponding downlink capacities may be the following,  $C_0 : C_1 : C_2 = 3 : 2 : 1$ . If the peer with the highest capacity, i.e.,  $P_0$ , requires a download time  $\Delta t$  to download a DU, then it takes  $2\Delta t$  for  $P_1$  and  $3\Delta t$  for  $P_2$ . If these three peers start downloading at the same time from the same source, then  $P_0$  will have to wait for  $5\Delta t$ , i.e., the time  $P_1$  and  $P_2$  are served until  $P_0$  is served next. Thus, it is thwarted by these two peers and the P2P network cannot fully profit by its higher capacities. As a consequence, the whole content distribution process is slowed down.

Our new approach avoids this thwarting due to heterogeneity by not restricting the amount of data, but the time a peer is allowed to download in a row. This approach is called time-based cooperation (TBC). Thus, peers with a higher capacity will serve earlier as new sources, since they are able to download more data in the same time. Alas, the effectiveness of this approach heavily relies on the peers' altruism to behave cooperative. The basic principle of this TBC approach is a time-out  $\Delta t$  which is the maximum time a user is allowed to download from a providing peer. Additionally, we still need a limitation of transferred volume, since MSD needs a reservation mechanism for the data currently downloaded to prevent downloading data twice. We set this limit to be  $V = 540$  kB. The providing peer stops serving the downloading peer if either the time  $\Delta t$  is spent or the volume  $V$  is uploaded. In particular, the downloading peer is interrupted after time  $\Delta t' = \min(\Delta t, \Delta t_V)$  while  $\Delta t_V$  is the duration a peer needs to download  $V$ . Note, that  $\Delta t_V$  might vary due to new file requests, churn, and VHOs of the downloading or uploading peers.

For the analysis of TBC, we consider the following scenario which makes greater demands on optimization. There are 100 mobile peers which move around in the future network layout. There are a total of 20 different files, each of size

9500 kB. On average, each peer shares a single file at the beginning. The peers want to download all remaining files they not already have, i.e., 19 files on average. The interarrival time between two file requests is exponentially distributed with a mean  $\mu_F = 40$  s. Additionally, we consider churn here. The peers switch from online to offline with exponentially distributed lengths of the online and offline phases, each with a mean  $\mu_C = 1$  h.

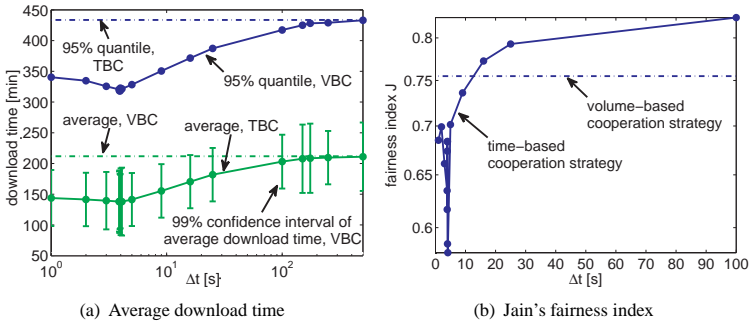


Figure 2.10: Volume-based (VBC) and time-based (TBC) cooperation strategy

Figure 2.10(a) shows the average download time and the 95 % quantile of the download time of the VBC and TBC approach. The latter's performance depends on the choice of the time  $\Delta t$ , a peer is allowed to download. The figure illustrates that the performance of TBC is always at least as good as of VBC. We see that the larger  $\Delta t$  the smaller is the performance gain. This results from the peers with fast technologies having to wait the longer on peers in slower technologies the larger  $\Delta t$ . We can see that there is an upper bound for  $\Delta t$  beyond which the two approaches give the same results since even a peer in the slower technology is able to finish its download before the time-limit is exceeded. Figure 2.10(a) suggests that there is an optimal value of the allowed download time, roughly at  $\Delta t = 4$  s. However, the size of the 99 % confidence intervals of the average



download times, indicated by error bars, is quite large. Hence, it's difficult to find an optimum. This results from the fact that we are investigating a highly dynamical and complex system. The behavior of such a system can vary largely depending on small changes in the overall situation as, e.g., a peer that stayed within WLAN for a longer time and/or became a new seed for a file faster.

A second relevant aspect of a P2P CDN is fairness, i.e., whether all peers are treated equally. Figure 2.10(b) shows Jain's fairness index of the download time for VBC as well as TBC in dependence of  $\Delta t$ . The figure reveals that the fairness is lowest if the performance of TBC is best. This is due to high-capacity peers being preferred by TBC and being able to download more data in the same time. The dots in Figure 2.10(b) represent the average fairness index of 20 simulation runs. It has to be noted, however, that due to the highly dynamical system the fairness indices of different simulation runs of the same scenario are varying strongly. This explains, for example, the fluctuations of the fairness index of TBC for  $\Delta t \approx 4$  s. However, a clear trend can be observed for the fairness index in dependence of the parameter  $\Delta t$ .

#### 2.3.5 Utilization of Scarce Resources in Heterogeneous Networks

In theory, a single uplink performs best in distributing a file over a P2P file sharing network under certain assumptions. In practice, these modeling assumptions are broken. E.g. by churn which lets peers go offline, by MSD which enables peers to download files from different sources in parallel, by the fact that a peer usually downloads not only a single file in a row, and by the heterogeneity of the B3G network as well as the mobility of the peer which break the equality and constancy of down- and uplink capacities, respectively.

By the use of a single uplink, another problem emerges in our simulation scenario, the waste of uplink capacity. With a single uplink, the WLAN capacity cannot be always utilized. E.g., if a peer in WLAN lets download a peer in UMTS. With an uplink capacity of 1 Mbps, this peer could almost saturate the downlinks

of three peers in UMTS, with a downlink capacity of 384 kbps. Instead, it wastes almost  $2/3$  of its uplink capacity, when it is restricted to a single uplink. This effect is even intensified, since we investigate a P2P network that makes use of MSD. Thus, the downloading peers may not even use their total downlink capacity. This leads to an increased waste of uplink capacities.

In order to utilize the scarce resources in a heterogeneous network, we develop a simple, but effective algorithm that is trivial to implement in an existing P2P network. The main feature of this algorithm is the iterative adaption of the number  $N$  of parallel uplinks. To ensure the performance improvement by this algorithm, we also introduced an upper bound  $N_{max}$  for  $N$ , since allowing an unlimited value of  $N$  can be negative for the P2P system. This can be explained by the following scenario. A peer in WLAN is able to serve several UMTS downlinks in parallel. A sudden switch to UMTS causes that the downloading peers will be further served with a rather small bandwidth, which is only a  $64 \text{ kbps}/1,024 \text{ kbps} = 1/16$ -th part of the original WLAN uplink capacity. Hence, we ensure by setting an appropriate  $N_{max}$ , that the minimal bandwidth each connection can be assigned cannot become too small as well as that the time until  $N$  has re-adapted to a sensible value keeps short.

The actual implementation of this cooperation strategy only effects the peer selection of an uploading peer, but not the chunk selection. Therein, each peer is initialized with a single uplink, i.e.  $N = 1$ , whenever joining the P2P network. The peer periodically accumulates the current bandwidths of the active downloading connections. Then, it checks whether the downloading peers have left over some capacity, i.e., the uplink of this peer is not completely utilized. In that case, the number  $N$  is increased as long as  $N_{max}$  is not exceeded, or until the capacity of the uploading peer is utilized. In contrast, if the result of the capacity check comprises that there is no uplink capacity left, i.e. the downloading peers use the uplink completely, then the number of uplinks is decreased by one. Thus, the remaining  $N - 1$  peers can increase their bandwidth per connection, if they have downlink bandwidth left. It has to be noted that the increase or decrease of  $N$  is not applied until a peer has finished its current download volume, and a

single new peer would enter the uplink. This is done in order to avoid that the uplink capacity of the peer is overbooked and download connections are cancelled. In particular, all connections are allowed to finish their current download of the DU. Thus, no connection is aborted just because a sudden overbooking situation emerges.

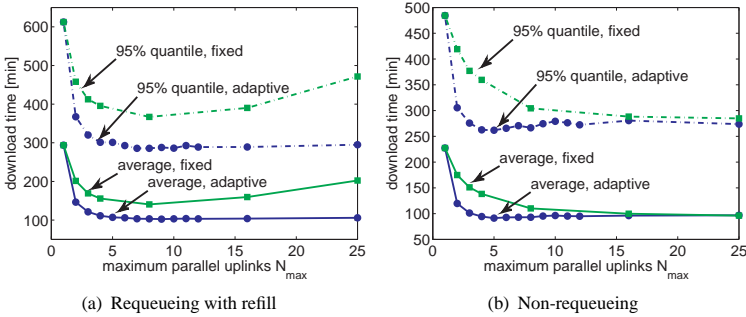


Figure 2.11: Adapting the number of parallel upload connections

Figure 2.11 shows the average download time for the future network layout when varying the maximum number of parallel uplinks  $N_{max}$  of the proposed adaptive approach. For comparison, we also consider the random chunk strategy with a fixed number of parallel uploads. Figure 2.11(a) shows the means and 95 %-quantiles of the average download time when using queueing techniques, while Figure 2.11(b) refers to the case of using non-queueing techniques like Mobile IP .

For  $N_{max} = 1$ , both approaches return the same result, since there is no possibility to adapt with a single uplink. We can also see that the download performance increases for both approaches for  $N_{max} > 1$ . This is due to the fact that a peer's uplink is not saturated with a single downlink connection, especially in the future network layout where peers are often connected to WLAN. Furthermore,

we see that the adaptive approach shows a significantly better download performance than the fixed approach. In detail, the average download time reduced from around 150 minutes with the fixed approach to around 100 minutes with the adaptive approach. This improvement is caused by the advantages of the adaptability, especially when switching between technologies with a big difference in the uplink capacities as it is common in the future network layout.

For  $N_{max} \geq 8$ , the download performance of the adaptive approach becomes invariant to further increases of  $N_{max}$ . If the uplink is already saturated by a certain number of downloading peers, the adaptive approach makes the current number  $N$  of parallel upload not exceed a certain threshold. Thus, increasing  $N_{max}$  has no more impact. In contrary, the download performance of the fixed approach decreases with increasing  $N_{max} > 8$  when using requeueing techniques, as we can see in Figure 2.11. The reason is that with a higher number of parallel upload the download bandwidth per peer decreases, the download times for files increases, and thus sources for these files are available later. In Figure 2.11(b), the average download times are given when using non-requeueing techniques. For large  $N_{max}$  the adaptive and the fixed approach converge. The reason for this simple derives from the actual implementation of this approach. We do not apply an adaptation of the number  $N$  of parallel uplinks, until the download of the DU is successfully finished and another peer enters the uplink queue. Since the non-requeueing technique keeps the current connection although a peer conducts a VHO,  $N$  is not adapted until the DU download. Obviously, this algorithm can be further optimized to utilize efficiently the scarce resources, even in the presence of Mobile IP or other techniques. However, this is part of on-going research.

Besides the general performance improvement of the adaptive algorithm, there is another advantage. The cooperation strategy does not need to dimension an appropriate number  $N_{max}$  of parallel uplinks a priori. This is especially useful, when the CDN is established in more complex and heterogeneous environments with unknown peer characteristics.

## **2.4 Future Trends in Mobile Peer-to-Peer**

A variety of peer-to-peer content distribution systems have emerged in recent years that use logical overlays on top of the physical Internet infrastructure for distributing content among the users. In these peer-to-peer systems the available resources of the end users are utilized to help the content dissemination process: the end users assist the system by storing data locally and by uploading data to other users. Peer-to-peer file sharing systems have been the dominant source of traffic in the Internet over the past years. The recent rise of the popularity of streaming video services, like YouTube, shows however that there is sufficient interest for video-on-demand and hence it can become a major source of Internet traffic in the future. Consequently, peer-to-peer video streaming applications, which offer either video-on-demand, live streaming or both, are expected to gain popularity. With the increasing capabilities of mobile devices in terms of computational power and graphical displays, it is expected that these applications which are quite popular in the Internet will also be implemented successfully for mobile end users. As video streaming applications have more strict requirements in terms of timely delivery of data packets for a smooth playout of the video, it is expected that the user behavior for such applications will change.

The large amount of data exchanged in overlay applications is a significant source of costs for Internet service providers (ISPs) and also mobile operators. Overlays typically span the networks of several ISPs and operators, and due to the logical separation of the overlay and the physical network topology, content is often exchanged between end users that reside in different ISPs. Such inter-domain traffic leads to interconnection costs for the ISPs. Consequently, an ISP would like to (i) control and manage the traffic from overlay applications in order to reduce its traffic costs and (ii) compete with the existing applications by offering data distribution services himself instead of being just a 'bit pipe'. An ISP or operator has different options to control the overlay traffic. For example, the ISP can provide the means so that sophisticated cooperation strategies can be employed among the peers, e.g., they can take into account the network topology

or any other useful information, and this cooperation might lead to a more ISP-friendly way of content distribution. The consideration of inter-domain traffic in the context of mobile P2P applications is in the focus of future investigations.

In order to meet the user's demands and requirements, the Quality of Experience, of P2P applications has to be fulfilled. Sophisticated cooperation strategies are the foundation for efficient and robust content distribution systems. In the context of mobile and heterogeneous environments, a future key topic is multi-homing for mobile P2P applications. This means that the users can utilize several access technologies at the same time. Thus, a cooperation strategy might consider several access technologies and uses the most appropriate one for specific applications. For example, a video-on-demand service requires only a low bandwidth when displayed at a mobile device, but has strict quality of service requirements in terms of delay and jitter. In this case, a UMTS connection via a dedicated channel providing a constant bandwidth might be more appropriate than a WLAN access, although the available capacity is higher for WLAN. Furthermore, it is interesting how to realize multi-homing for P2P-based system. As incentive mechanisms of existing P2P CDNs are usually based on tit-for-tat or credit point systems and identify a user by its IP address, the simultaneous access of a user with different IP addresses per interface will lead to problems. The question arises how to implement multi-homing transparently to existing P2P protocols such that the user will achieve a performance gain.

The combination of different cooperation strategies is also a topic of future work. Cooperation strategies are often optimized to reach a particular goal. CycPriM tries to overcome the drawbacks of selfishness, while the time-based data exchange and the adaptive parallel upload strategy aim at mastering mobility and utilizing scarce resources. However, in an heterogeneous environment with selfish peer, the advantages of both strategies have to be integrated into a common cooperation strategy. The investigation of the combination of different strategies has to reveal whether they can additionally support each other and mutually improve the user's gain.

## **2.5 Lessons Learned**

The performance of P2P content distribution in cellular mobile networks is determined mainly by the implemented cooperation strategy at a local peer. Major challenges which arise typically in cellular networks are the selfishness and the heterogeneity of peers. The selfishness of users leads to the last chunk problem, while the heterogeneity wastes expensive resources in the system. In this chapter, a background on common cooperation strategies which are based on the multi-source download is given. For performance evaluation purposes, the key performance characteristics are formally defined, before related work in the field of P2P content distribution is reviewed.

In this chapter, we have shown that in cases where most of the peers are selfish, i.e., show a leeching behavior, a chunk selection strategy like least-shared first is able to overcome the last chunk problem. However, it is necessary to update the global information about the number of sharing peers for every chunk. Furthermore, the LSF strategy has to prevent changes to the peer capabilities, i.e., it must not allow parallel upload at a providing peer. A more sophisticated cooperation strategy, the CycPriM strategy, has proven to be robust against leeching as well as against changes of peer capabilities. The basic idea is to modify the peer selection strategy of uploading peers which implicitly determines the chunk selection strategy of the downloading peers. This seems to be more appropriate to deal with selfish peers in a heterogeneous environment.

An adequate peer selection mechanism has also been shown to be efficient in a B3G network with mobile users conducting vertical handover between different wireless access technologies. In particular, the adaptation of the number of parallel upload slots of a multi-source download mechanism has shown to efficiently utilize the available resources. Common cooperation strategies waste these resources, as the heterogeneity of peers is not considered.

The comparison of today's and future network layouts in different load scenarios showed that non-requeueing techniques like Mobile IP are recommended in mobile P2P CDNs if this technique only requires additional delays in the order of

a few seconds. In future network layouts, the increased uplink capacity e.g. due to better WLAN coverage leads to smaller download times. In order to foster the download from such high-capacity peers, a time-based cooperation strategy is introduced. Although the fairness of the system is decreased, as high-capacity peers are able to download more content in the same time, this completely different approach allows for efficient exchange of files for all users. The implementation of particular cooperation strategies to overcome problems in mobile environments is an important problem and will surely constitute the basis of further studies.

The lessons learned in this chapter on cooperation in mobile P2P cover different aspects, which are the emerging user behavior, the identification of problems, and the design of new mechanism. In particular, due to the application of the P2P paradigm for content distribution, different user behavior emerges. Users may appear selfish or altruistic and show churn behavior, which impacts significantly the performance of the P2P CDN. The mobile environment introduces additional problems and requires the determination of major influence factor, like mobility and heterogeneity of users, and their impact. The identification of problems due to the emerging user behavior and the new technical challenges guides to the application of existing mechanisms, e.g. using mobileIP to overcome drawbacks on application layer due to vertical handover of users. The performance evaluation of various scenarios, however, requires a framework for also considering and predicting future scenarios, as well as a methodology to investigate them, e.g. modeling mobility of users in the context of P2P systems in cellular networks. Finally, this performance evaluation fosters the design of new mechanisms to overcome the identified problems and to improve the performance of the system and the experienced quality of the end user.



## 3 Modeling of Online TV Recording Services

Television has traditionally been an entirely broadcast-oriented medium. However, nowadays, new technologies delivering packetized digital video data are on the brink of replacing conventional television broadcasts via terrestrial, satellite or cable transmissions. With increasing access bandwidth speeds for end users, the new *Internet Protocol-based television* (IPTV) has gained popularity as a means of delivering high-quality video images. The technological advancement in high-speed Internet accesses facilitates such possibilities. Meanwhile, a large coverage of DSL or fiber-to-the-home (FTTH) is available and improved video encoders like H.264 permit the transmission of clear high-resolution video images at half the bitrate of MPEG2 current on DVDs.

An important distinction of IPTV systems is by their content distribution method: network-based video recorders (NVR), video-on-demand (VoD), and live TV streaming. In this chapter we focus on network-based TV recorders. They operate basically in the same way as home hard-disc video recorders with the only difference that the content is recorded and stored at some remote machines in the Internet. An example for such a service is *OnlineTVRecorder* (OTR). The live TV program is recorded at the OTR server and registered users can download their previously programmed shows and later view them offline.

The volume of such video traffic transported over the Internet has drastically increased over the last few years. In the context of OTR, the download of multimedia contents may consist of large files imposing high requirements on the bandwidth of the file servers.

**Delivery via P2P or server clusters** In conventional systems this means that the servers must be properly dimensioned with sufficient capacity in order to service all incoming file requests from clients. On the other hand, P2P technology offers a simple and cost-effective way for sharing content. Providers offering large volume distributions (e.g. Linux) have recognized the potential of P2P and increasingly offer downloads via eDonkey or BitTorrent.

As explicated in Chapter 2, in P2P all participating peers act simultaneously as clients and as servers, and the file is not offered at a single server location, but by multiple sharing peers. Since the load is distributed among all sharing peers, the risk of overloading servers with requests is reduced, especially in the presence of flash crowd arrivals. However, this flexibility comes at a slight risk. Since the shared file is no longer at a single trusted server location, peers may offer a corrupted version of a file or parts of it. This is referred to as *poisoning* or *pollution* [110] depending on whether the decoy was offered deliberately or not. When the number of fake peers is large, the dissemination of the file may be severely disrupted. All of this leads to a trade-off consideration between high reliability at the risk of overloaded servers and good scalability where the received data may be corrupt.

From the view point of a content provider, the P2P technology faces, however, another major challenge. P2P file sharing platforms are often used for illegal distribution of copyright-protected contents. In order to protect now its own contents, a content provider may utilize the fact, that files in P2P are not offered by single trusted server locations, and inserts some fake peers offering corrupted versions of its contents. After a chunk is downloaded, it is checked for consistency via MD5 hashes and in case an error is detected, the chunk is discarded and downloaded again. Thus, if a user downloads some data from a fake peer, the entire download is prolonged. The hope of the content provider is to heavily disturb the data dissemination, such that the user's impatience is exceeded after some time and the user gives up downloading via the P2P system. In that case, the user may use the content's provider platform instead of P2P to download the video data.

---

Currently, most IPTV service providers offering copyright-protected contents use high-performance server farms in order to manage and control the provided service with respect to security or AAA, i.e. authentication, authorization and accounting. The provider aims at satisfying its customers, i.e. to provide a good QoE. At the same time, the operator may also profit indirectly from a good media consumption experience for end users, as the increased customers' fidelity will also attract other potential users to use this service. In the context of video delivery via OTR, the QoE takes into account three tasks which are independent of the technical realization: (a) efficient download of the content, (b) reliability of the system, (c) fairness among users. They can be quantified in terms of download time, the number of download aborts / the success ratio, and the variation of download times among different users. In this context, we define reliability as the availability of a single file over time in a disruptive environment which is expressed by the success ratio of downloads. In order to provide a good QoE, however, such an IPTV system has to be dimensioned properly.

**Goal and structure of chapter** The goal of this chapter is twofold. First, we provide appropriate models to describe impatience and flash crowd effects in a high-performance server cluster and an eDonkey-based P2P systems for delivery of OTR video contents. Second, these models allow evaluating the impact of user behavior and dimensioning of system parameters. In case of the high-performance server cluster, for example, the number of available download slots has to be estimated. In case of the P2P system, the model allows investigating the impact of fake peers. This can be used either (i) to quantify the disturbance of the P2P system due to malicious peers when the service provider relies on P2P technology or (ii) to dimension the number of fake peers to save copyright-protected contents for being distributed in illegal file sharing system.

The chapter is organized as follows. Section 3.1 gives a comprehensive background on video content delivery. This includes a short overview and a classification on IPTV and P2PTV systems. We explain the OTR service in more detail, as the investigated high-performance server clusters are based on the ex-

isting OTR service and the disseminated contents in the P2P system are OTR video files. After that, we review related work with a focus on the performance of P2P CDNs, the diffusion of files within them, and epidemic models to describe the file diffusion behavior. In addition, existing work related to the analysis of queueing systems with user impatience is presented, since we apply later basic queueing theory to evaluate the performance of the OTR server clusters. Section 3.2 presents a measurement study of typical video contents in the Internet. First, we take a closer look on OTR TV shows, before video contents provided by YouTube are analyzed.

The model and the analysis of the high-performance server clusters is presented in Section 3.3 which considers user impatience due to waiting times in queues or too long download times. To describe the time-dynamic behavior of the system we develop a deterministic fluid model. This is useful to consider phenomena like flash crowd arrivals which may occur due to the popularity of TV shows. Next, we model the performance of an OTR video delivery service by means of a Markov model to derive the stationary sojourn time, which is used for dimensioning the system. The investigated key influence factors comprise the impatience during downloading and waiting, the number of available download slots and the variation of the file size distribution.

Section 3.4 investigates the pollution of the P2P content distribution service, again, taking into account user impatience as in the server case. We derive an epidemic model for the file diffusion, starting from a simple SIR model from biology and refine this to a detailed file sharing model. After that, we introduce pollution in our model which is then described as a flow model by a differential equations system (DES). Further, we show how to obtain the download duration out of the DES. We investigate some exemplary scenarios to compare simulation results with numerical solutions of the DES and quantify the influence of pollution, selfishness, and supporting servers in the P2P system. This brings us to Section 3.5 in which we compare the QoE in the centralized system and in the P2P system. Finally, the lessons learned in this chapter are summarized in Section 3.6.

## 3.1 Background: Video Content Delivery

The term *IPTV* stands for Internet Protocol-based Television and is often used to refer to IPTV systems with particular characteristics. For example, the ITU focus group on IPTV defines IPTV “[...] as multimedia services such as television/ video/ audio/ text/ graphics/ data delivered over IP based networks managed to provide the required level of QoS/ QoE, security, interactivity and reliability.” However, in general, IPTV means a system where a digital television service is delivered using the Internet Protocol (IP) over a network infrastructure. An even more general definition of IPTV is television content that, instead of being delivered through traditional broadcast and cable formats, is received by the viewer through the technologies used for computer networks.

One of the main features of IPTV is its high degree of interactivity. Users are no longer restricted to the broadcast schedules of TV stations, but can choose the program they wish to see on-demand, whenever, wherever, and on whatever device they want (TV, PC, portable player). Additionally, further value-added services are often included, such as chat functions or other feedback mechanisms allowing the viewers to provide ratings or discussion forums on the shows. Therefore, offering IPTV has become an attractive business model for telecommunication service providers. Many providers no longer limit their offer just on telephone or Internet access, but provide so-called triple play services, integrating Internet, VoIP telephone services, as well as television or movie channels. Furthermore, it is also appealing to businesses, which can offer personalized advertisements, individually tuned to the TV programs the customer is currently watching or localized to his region of access. An IPTV service provider usually delivers the video contents over a well-dimensioned network, which is carefully engineered to ensure bandwidth efficient delivery of vast amounts of video traffic.

In Section 3.1.1, we first give a short overview on the different ways of distributing the video contents. We describe the OTR service as a popular example of network-based video recorder in detail in Section 3.1.2, before we review existing work related to this chapter in Section 3.1.3.

### 3.1.1 Classification of Video Delivery Services

In general, IPTV network architectures can be categorized in two main classes: centralized and distributed. *Centralized systems* follow the traditional client/server (C/S) paradigm, where server farms with access queues balance and manage the load among the content servers. Examples are YouTube and OTR servers. Here, the client directly connects to the server via HTTP and after a queuing/buffering delay directly streams/downloads the contents from the server.

On the other hand, *distributed systems* are usually based on P2P technology, e.g. Zattoo, Joost, PPLive. Each user also automatically acts as a relay for other peers in the network. This means that while downloading/watching a video, the peer provides the already downloaded content to other peers. There are several advantages of using P2P-based content delivery systems, as they react better to sudden bursts in requests arrivals. However, the overlay topology must be dynamically set up first and the network must be adaptive to topology changes due to churn or to selfishness of users. The term *P2PTV* refers to P2P applications designed to distribute video streams via a P2P network.

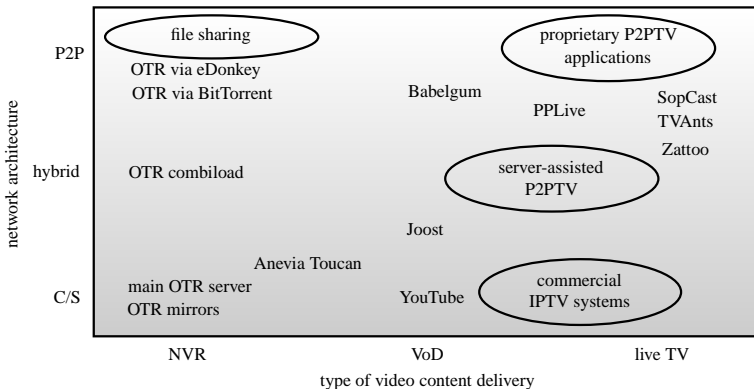


Figure 3.1: Classification of video content delivery

Beside the differences in their network architecture, the various IPTV systems differ significantly in their video delivery mechanism and Figure 3.1 classifies existing IPTV applications accordingly. Basically, there are three major categories of IPTV content distribution methods: live TV streaming, Video-on-Demand, and network-based video recorder.

**Live TV Streaming** describes the streaming of live TV channels over IP-based networks, just as they are being aired over conventional broadcast media. The current live TV program is packetized and often streamed as application-layer multicast to the connected peers in the overlay, which then share the stream with other peers. Live streaming applications require sophisticated mechanisms because all users watching a news broadcast or a popular sports event are typically interested in the same piece of the stream at the same time. Therefore, delays of one or more minutes seem unacceptable.

**Video-on-Demand** permits a user to browse a catalogue of video files and as soon as one is requested its playback is started. Thus, VoD is not restricted to any broadcast schedules, but entirely to the user's demand. Among the available VoD systems, YouTube (<http://www.youtube.com>) enjoys high popularity among Internet users. YouTube is a centralized video sharing website where users can upload, view, and share short video clips.

**Network-based Video Recorders** operate basically in the same way as home hard disc video recorders, only that the content is recorded and stored on a remote server. The live TV program is recorded at a remote machine in the Internet. Users can download their previously programmed shows and later view them offline on a PC or handheld device.

#### 3.1.2 Online TV Recording Service

A popular example for such a video recording service is the OTR service in Germany which is the underlying video content delivery system for our investigations in this chapter. Users can currently select shows covering around 40 channels of German television, but support for other countries is planned. The

access to OTR is provided by a portal at its main Internet homepage <http://www.onlinetvrecorder.com>. There, a registered user has the possibility to choose programs he wishes to record from an electronic program guide (EPG) or can download previously recorded shows. In order to safeguard licensing restrictions and prevent unauthorized access, a user may only download files that he had previously recorded. For this reason, the content is offered encoded which cannot be played directly, but must be decoded prior to playback.

The video files offered by OTR can consist from several hundred megabytes up to 1 GB or more depending on the length of the TV show, as well as on the encoding format, e.g. high quality H.264, standard quality DivX, or MPEG-4 for portable devices (iPod, PSP, etc.). The recordings can be either downloaded directly from the main server, from user-created mirror sites, or alternatively via P2P file sharing networks (eDonkey or BitTorrent). However, the majority of clients are using the HTTP-based server download platforms and in Section 3.3 we will focus on this type of file transfer. In the following, when we refer to OTR server, we treat the main server and the mirror servers in the same way, since their basic operation is the same with the only difference that mirror sites usually offer only a subset of available recordings after a slight delay.

As the OTR server farms are often overloaded, new requests are queued when the number of provided download slots is full. The restriction to a maximum number of simultaneous downloads guarantees a minimal download bandwidth for each user. The download duration itself depends on the total capacity of the server farm and the number of users sharing this capacity. However, users who might encounter slow downloads may abort their downloading attempt if their patience is exceeded. When requesting a file, the user may begin immediately his download if the server has available download slots, else he will be put on a waiting queue. OTR also offers prioritizing premium users who pay for advancing faster in the waiting queue, but we assume later in our model that all users are treated equal. Users who experience too long waiting times might abort their download attempt before being served.



### 3.1.3 Related Work

As we want to compare the reliability and efficiency of a P2P-based and a client/server-based OTR system, we review literature regarding the performance of P2P CDNs with a special focus on flash crowds and pollution. Next we review related work on P2P file diffusion and epidemic models which is the approach we follow to analyze the P2P-based system. Finally, an overview on existing work on the analysis of queueing systems with user impatience is given, since we develop a processor-sharing model with impatience to analyze centralized OTR.

#### Performance of P2P CDNs

Most studies on the performance of P2P systems as content distribution network rely on measurements or simulations of existing P2P networks. For example, Saroiu et al. [70] conducted measurement studies of content delivery systems that were accessed by the University of Washington. The authors distinguished traffic from P2P, WWW, and the Akamai content distribution network and they found that the majority of volume was transported over P2P. A comprehensive survey of different P2P-based content distribution technologies is given in [89]. In [105] a simulation study of P2P file dissemination using multicast agents is performed and the propagation under different conditions is studied. Hoßfeld et al. [10] provide a simulation study of the well-known eDonkey network and investigate the file diffusion properties under constant and flash crowd arrivals. However, most work on P2P file diffusion as those mentioned above usually do not assume any fake files from pollution or poisoning.

Han et al. [98] study the distribution of content over P2P and consider rewarding strategies as incentives to improve the diffusion. They show that the network structure in terms of hierarchy and clustering improve the diffusion over flat structures and that compensating referrers improves the speed of diffusion and an optimal referral payment can be derived. The user behavior and an analysis of the rationale in file sharing is studied in [90] using game theory. The focus lies on free riding in the network and the authors offer suggestions on how to improve

the willingness of peers to share. Qiu et al. [103] model a BitTorrent network using a fluid model and investigate the performance in steady state. They study the effectiveness of the incentive mechanism in BitTorrent and prove the existence of a Nash equilibrium. Rubenstein and Sahu [119] provide a mathematical model of unstructured P2P and show that P2P networks show good scalability and are well suited to cope with flash crowd arrivals. Another fluid-diffusive P2P model from statistical physics is presented by Carofiglio et al. [109]. Both, the user and the content dynamics are included, but this is only done on file level and without pollution. These studies show that by providing incentives to the peers for sharing a file, the diffusion properties are improved. We include appropriate parameters in our model which capture this effect, while also considering pollution.

Christin et al. [110] measured content availability of popular P2P file sharing networks and used this measurement data for simulating different pollution and poisoning strategies. They showed that only a small number of fake peers can seriously impact the user's perception of content availability. In this chapter we present a diffusion model for modeling eDonkey-like P2P networks based on an epidemic SIR [156] model. This model includes pollution and a peer patience threshold at which the peer aborts its download attempt.

#### **P2P File Diffusion and Epidemic Models**

Epidemic methods have been considered as simple and effective protocol for disseminating data in communication networks. The main feature of these *gossip*-based protocols is that they do not require any specific topology and that any node has the same probability to contact another node. This approach has been used to devise gossip-based protocols operating in mobile ad-hoc networks [67] or in unstructured P2P networks [94, 100, 101]. The effect of the network topology on the spread of the epidemics is studied in [113].

In epidemic computing, nodes contact each other with a certain rate and depending on the rate of cure to infection a disease may become an epidemic. Epidemic models are also well suited to model the diffusion behavior of specific

information in a network, see [68], and has often been applied to forecast the spreading of worms and viruses in the Internet [65]. In a very similar manner, epidemic models can be used to model file diffusion in P2P file sharing networks. The papers found on P2P file diffusion either consider measurement studies, e.g. [99, 104], or by means of simulation [10, 106]. A theoretical model of a BitTorrent P2P network can be found in [103]. The authors use a fluid model and study the performance of the network and investigate the effects of the incentive mechanism. This work is extended to considering different classes of access links in [102] and the authors show that bandwidth heterogeneity can have a positive effect on content propagation. While in [103] and [102] the steady-state network performance is investigated, we emphasize the time-dynamics of the system which requires us to consider non-stationary process, e.g. caused by flash crowd arrivals of file requests. Measurement studies on pollution and poisoning can be found in [110, 117]. Both papers show that there can be a substantial influence from introducing even a small number of fake peers to the network.

Chen et al. [93] and Thommes and Coates [121] use a model based on the classic SIR approach, which is also the fundamental idea of our work. However, as we will see later from comparison with simulations, the steady state assumptions made in many papers, e.g. in [103], are not appropriate due to the highly non-stationary behavior of the system. The transitions are made between the states after a fixed amount of data has been downloaded. Using simply a transition rate does not properly reflect the system dynamics. The focus in this work is on the time-dynamic transition phase during the diffusion process. This facilitates the consideration of flash crowd arrivals of file requests. When considering illegally shared content this often corresponds to the release date of a song or a movie as at that time the number of requests will be highest. For legally distributed software (e.g. distributions of the Linux operating system) P2P file sharing is also much more effective for content distribution than client-server as it relieves the download servers from overloading when new software releases are available [99]. Furthermore, we investigate the influence of fake peers that share corrupt or fake content on the diffusion process.

### Queueing Systems with Impatience

In general, with a slight abuse of the Kendall notation for queueing systems, the model we use to investigate the high-performance OTR server cluster can be expressed as  $M(t)/GI/1^{n^*}$ -PS with user impatience  $\Theta$ , an unlimited waiting queue, and a server capacity  $C$  which is shared among  $n^*$  users at maximum. Thus, the service rate is influenced by the download rate of a user as well as the user impatience  $\Theta$  and depends on the number of currently served users.

Admission control to the system can be taken into account by restricting the size of the waiting queue. However, in this paper we use the number of download slots  $n^*$  to guarantee the bandwidth per user and only investigate the impact of the user's impatience on the system's performance. While *reneging* is considered with an i.i.d. random variable  $\Theta$ , *balking*, i.e., taking back the download request if the waiting queue is too long, is neglected in our model. We focus on the effect of wasted capacity due to users' impatience regardless of whether they are being served or not, and the impact of variability of the file size distribution, which is expressed by the service rate. Our findings show that the ratio of successful downloads increases with the variability of service time.

Basically, there are several approaches on how to analytically evaluate such a system depending on the number of currently available download slots  $n$ . If  $n \leq \lfloor \frac{C}{R} \rfloor$ , the user's access bandwidth  $R$  limits the download rate. For  $n^* \leq \lfloor \frac{C}{R} \rfloor$ , this effectively results in a  $M(t)/GI/n^*$ -FCFS system with independent service rates, since  $\Theta$  is an i.i.d. random variable and the service rate is constant. The service rate depends only on the file size and the users' access bandwidth. An analytical evaluation is provided in [76]. For  $n > \lfloor \frac{C}{R} \rfloor$ , the download rate and therefore the service rates depend on the current state of the system. On the other hand, if the downlink of a user is not the limiting factor, i.e., a user can always utilize the offered bandwidth of the server ( $C < R$ ), the system approaches a real processor sharing system with increasing  $n$ , which is investigated in [54, 129]. In the past, a lot of research has been dedicated to the analysis of queueing models with impatience. Barrer [50] was among the first to analyze an  $M/M/1$  system

with deterministic impatience thresholds. In the following, more sophisticated FCFS models with Markovian arrival and service processes were investigated in [155], [52], and [62].

As we will see later, the general service time has a great impact on the performance, however, it is well known that for such systems only approximative evaluations can be performed for metrics of interest [155]. For the exact analysis of steady-state sojourn times, we focus therefore only on simple models which are easily analytically tractable. Nevertheless, our measurements of video contents in Section 3.2 show that these assumptions are valid. As we also consider time-dependent flash crowds arrivals a transient analysis as described later in Section 3.3.2 is additionally required.

## 3.2 Measurement of Video Contents

The performance evaluation of a video content delivery systems requires several input parameters. Among others, these are the arrival process, the user behavior, and the video duration as well as the corresponding data size. In the case of a centralized system additionally the number of servers, the server discipline, and the queue length are of interest. We focus in this chapter on videos offered by OTR or YouTube. However, as OTR and YouTube are proprietary, each system is regarded as a black box and measurements are taken from the user's edge. Thus, only the service time can be obtained, which is in our case characterized by the offered video file sizes. To investigate the impact of the remaining parameters in a OTR system, parameter sensitivity studies are performed, e.g. by varying the popularity of files and the accompanying arrival rate of user requests.

OTR and YouTube are both web-based server-oriented systems. OTR records TV shows at the main server or mirrors, and HTTP or FTP over TCP is used for file transfer. The achieved download speed heavily depends on the selected mirror. For many mirrors, the user's DSL access speed is the limiting factor. However, a user often has to wait for an available download slot until he is served. OTR supports different video resolutions from low quality ( $160 \times 120$ ) to high

quality ( $720 \times 576$ ) and post-processing of the videos allows e.g. to remove commercials. On the other hand, YouTube is designed for VoD sharing user-created contents and video streams are downloaded over HTTP. Due to advertisements on the web portal, several TCP connections to different IP addresses are established. During the course of the measurements, only low resolutions ( $320 \times 240$ ) are supported. Table 3.1 summarizes the statistics of the measurement studies and considers the file size, the duration of videos, as well as the codec efficiency as ratio between file size and video duration. The standard deviation is abbreviated as ‘std’, the coefficient of variance as ‘cov’, and the skewness as ‘skew’.

Table 3.1: Basic statistics for OTR and YouTube video contents

		mean	std.	coef.	skew.	kurtosis	median	min	max
OTR 11563 samples	duration [min]	47.21	29.27	0.62	1.14	4.42	45	1	195
	size [MB]	343.19	186.71	0.54	1.12	4.31	305.87	0.06	1236.87
	efficiency [kbps]	1155.01	662.93	0.57	7.33	86.05	1038.42	0.71	16310
YouTube 21014 samples	duration [s]	339.11	419.16	1.24	7.91	90.64	252	5	10233
	size [MB]	12.38	14.88	1.20	7.09	69.25	9.41	0.07	274.59
	efficiency [kbps]	302.11	52.43	0.17	-1.61	16.81	318.54	1.12	1040.52

### 3.2.1 Network-based Personal Video Recorder – OTR

The measurements which were made in April 2007 consist of 11,563 file samples from 19 different TV channels. According to the information provided by OTR, standard video files are encoded at a resolution of  $512 \times 384$  pixels at a video bitrate of about 750 kbps and an audio bitrate of 128 kbps. The measured data contains only standard quality video files and consist of approximately 80 % encoded in the DivX format and 20 % in Windows Media Video (WMV) format.

Figure 3.2(a) shows the probability distribution of the TV show durations in minutes. The majority of the files (95%) are discretized in units of 5 min. We can distinguish 4 different categories of TV shows. Most files are short features (e.g. animation series) of about 30 min and shorter files may be for instance news shows. Another peak can be found between 45-60 min which is the usual duration of TV dramas or other periodical shows. Movies usually have the duration

between 90-120 min and very few larger recordings of special events exist, like the broadcasts of live sports events.

However, we are more interested in the file size distribution in order to approximate the download time than the duration of the shows themselves. Figure 3.2(b) shows that the actual file size distribution  $f_s$  has a mean of 368.31 MB and standard deviation of 196.82 MB. It can be well fitted by an Erlang- $k$  distribution with  $k = 3.34$  phases and an average volume of  $E[V] = 107.67$  MB per phase, i.e., it is the sum of  $\lfloor k \rfloor$  independent identically distributed random variables each having an exponential distribution with mean  $E[V]$  and an exponential distribution with mean  $(k - \lfloor k \rfloor)E[V]$ . Due to the real-valued  $k$  a Gamma distribution is used for numerical computation. The mean squared error between the fitted Erlang- $k$  distribution and the measured values is only  $E^2 = 0.0008$ . Using an exponential distribution yields to a larger mean squared error of  $E^2 = 0.0205$ .

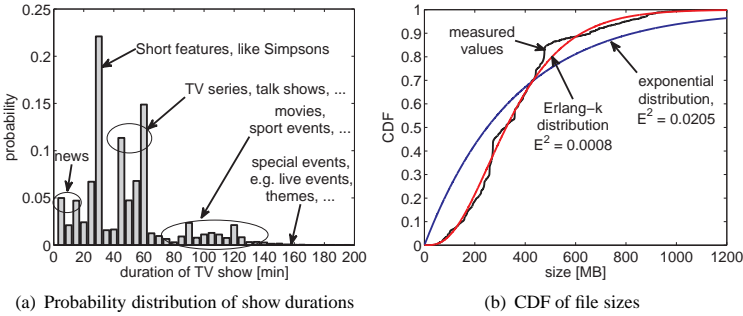


Figure 3.2: Measured TV show durations, file sizes, and codec efficiency of OTR

The codec efficiency is defined as ratio of the file size over the duration of the TV show in kbps. The probability density function has a distinct peak at about 1 Mbps and is comparable to other standard quality video formats, such as VCD or SVCD. The measured values could be well fitted with a log-logistic distribution superimposed with a Dirac function at the peak value, cf. [43].

### 3.2.2 Server-based Video-on-Demand – YouTube

The measurements were conducted in December 2007. In total, we downloaded 21,014 randomly selected video streams from the YouTube website and analyzed their file sizes and durations. For the data transport, an HTTP connection to the server is established and the content is delivered via TCP. Currently, YouTube uses the H.263 video codec and the MP3 audio codec, packed into the flash video container (file extension *.flv*). The video bitrate of a random stream is about 300 kbps, while the audio bitrate is typically about 60 kbps.

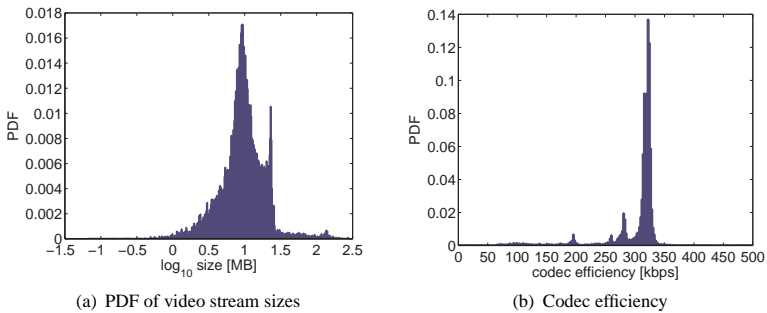


Figure 3.3: Measurements of YouTube video streams

Figure 3.3(a) shows the PDF of the sizes of downloaded video streams from YouTube. Note that the x-axis is logarithmically scaled, as user-created content is usually restricted to 10 min. With a special user account, however, it is also possible to upload larger video files. In our measurements, we observed video durations of up to 170 min and video sizes of up to 275 MB. Again, the file size distribution is leptokurtic highlighted by the strong peak of the PDF at 22.85 MB and shows a large kurtosis value of 69.25. This peak corresponds to the maximum allowed duration of 10 min for user-created contents.

For YouTube videos, the codec efficiency shows a very strong peak at roughly 315 kbps and is nearly constant, see Figure 3.3(b). Accordingly, the PDF of the



video stream durations looks quite similar to that of the stream size and is omitted here. Further results can be found in the technical report [43]. The negative skewness of the codec efficiency shows that the mass of the distribution is concentrated on the right of the figure and only a few videos require less bandwidth.

In [147], YouTube videos are evaluated by distinguishing the categories and popularity of video clips, as well as user access patterns like views and ratings. Small-world characteristics for video groups are identified and caching or P2P strategies for utilizing these clustering effects are proposed. Furthermore, [150] monitors YouTube usage in a local campus network in order to understand how it is used by clients. The video files and the transfer of the videos is characterized. They also provide statistics on the most popular videos at the YouTube site and get similar results as obtained in our measurement studies.

## 3.3 High-Performance Server Clusters

Currently, the majority of OTR subscribers are using server-based platform for downloading the recorded video contents. The user either downloads directly from the OTR server farm or from user-created mirror sites which we do not distinguish here. Since the service provider aims at satisfying its customers with a good QoE, such a centralized IPTV system has to be dimensioned properly to provide a high-performance server cluster.

In this section, we analytically investigate the performance of an OTR server in different scenarios and enhance basic queueing models by considering user impatience. Since the file sizes are very large, the download duration may take longer than the user is willing to wait. For this reason, we include the user impatience in our model, where a waiting or downloading user may leave the system before completing the download. This is taken into account with two impatience thresholds during downloading and during waiting. Queueing models with user impatience can be found in [62, 129], however, those models cannot be applied to our system. Further discussions of the existing literature were given in Section 3.1.3.

The OTR server model is described in detail in Section 3.3.1. To describe

the time-dynamic behavior of the system, e.g. for investigating flash crowds and time-depending popularity of files, we develop a deterministic fluid model in Section 3.3.2. To understand the key influence factors of the system and to dimension the system properly, we additionally model the performance of the OTR video delivery service by means of a Markov model which is the main focus of the investigation of the OTR server. The Markov model allows to derive the stationary sojourn time, which corresponds to the time until a typical user completes the download of a file. An exact analysis of the stationary sojourn times distribution follows in Section 3.3.3 which requires the derivation of the remaining sojourn time and the stationary distribution of the number of users in the system. The key influence factors investigated are presented in Section 3.3.4 and comprise the impatience during downloading and waiting, the number of available download slots and the variation of the file size distribution.

#### 3.3.1 Model of Centralized OTR System

In the following we will describe the model of an OTR server which is responsible of managing the demands of a maximum finite number of  $\underline{N}$  customers. We assume first that user requests arrive at the server according to a Poisson process with parameter  $\lambda > 0$ . When a request arrives and the system has free download slots, the client immediately proceeds with the download. Then, the user becomes a *downloading client* and we also say that the customer is *served*.

We may further assume that the server has a total fixed upload bandwidth of capacity  $C$ . This bandwidth is equally shared among a maximum of  $n^*$  simultaneously downloading clients. If there were more than  $n^*$  simultaneous clients, the exceeding customers would wait for a downloading slot to become free. We refer to those clients as *waiting clients*. The total number of downloading and waiting clients at the server is thus in this particular setting finite (with a maximum number equal to  $\underline{N}$ ). When downloading a file, the access bandwidth  $R_d$  of the client may be the bottleneck. We assume this bandwidth  $R_d$  is the same for all customers. Clearly, the condition  $R_d < C$  must hold.

The average rate at which clients complete their downloads also depends on the file size. We assume here that the file size is randomly distributed following an exponential distribution reflecting the measurement results described in Section 3.2. There, Figure 3.2(b) shows that the actual file size distribution has a mean of 368.31 MB and standard deviation of 196.82 MB. It can be well fitted by an Erlang-distribution with only small residual mean squared error  $E^2 = 0.0008$ . We will assume an exponential file size distribution for the sake of analytical tractability in spite of its slightly higher residual error, cf. Figure 3.2(b), however, the equations developed here could be extended to the Erlang case, too.

While in the system, a client might become *impatient* and decides to leave the system after a random amount of time. We assume that the average impatience duration depends on the speed of the download. That is, when there are less than  $n^*$  customers in the system, the impatience duration is distributed according to an exponential random amount of time with average  $\theta_1^{-1}$ . In the other case, i.e., there are more than  $n^*$  customers in the system, the impatience duration for customers being served remains the same, but the average impatience time for waiting customers is  $\theta_2^{-1} < \theta_1^{-1}$ . If we consider impatience being independent from being currently served or waiting in the queue, we simply use the random variable  $\Theta$  with average  $\theta^{-1}$ .

### 3.3.2 Time-Dynamic Evaluation with Fluid Model

User requests are assumed to arrive at the server according to a non-stationary Poisson process with rate  $\lambda(t)$ . The time-dependent arrival rate of user requests is a realistic scenario when looking at individual files, since the popularity of a TV show highly depends on the time it was recorded. Once a show becomes outdated, the interest for this file decreases. This phenomenon is referred to as flash crowd arrivals [10]. However, since a server may offer several different files, the overall rate may remain nearly constant. The superposition of time-dependent arrival processes with different starting points can be modeled as stationary Poisson process for a sufficiently large number of offered files per server.

When a request arrives and there are free download slots, the client may proceed with the download. We assume that the server system has a total fixed capacity  $C$  which is shared among all simultaneously downloading clients  $\mathcal{D}(t)$  at time  $t$ . The maximum number of users served in parallel is restricted to  $n^*$ . Thus, the time-dependent download rate  $\mu(t)$  is

$$\mu(t) = \frac{1}{\mathbb{E}[f_s]} \min \left\{ \frac{C}{\min \{\mathcal{D}(t), n^*\}}, R_d \right\} \quad (3.1)$$

for an average file size  $\mathbb{E}[f_s]$  and the maximum download rate is limited by the maximum physical rate  $R_d$  of each client.

In order to consider flash crowd arrivals, we use a fluid analysis technique. The state space of transitions is shown in Figure 3.4 and the differential equation system is given in Eqn. (3.5). The partial derivative of the function  $f(t, y)$  with respect to variable  $t$  is denoted as  $\partial_t f(t, y) = \frac{df(t, y)}{d(t)}$ .

$$\partial_t \mathcal{W}(t) = \begin{cases} 0 & \text{if } \mathcal{D}(t) < n^* \\ \lambda(t) - \mathcal{D}(t) \mu(t) - \nu \mathcal{W}(t) & \text{otherwise} \end{cases} \quad (3.2)$$

$$\partial_t \mathcal{D}(t) = \begin{cases} \lambda(t) - \mathcal{D}(t) \mu(t) & \text{if } \mathcal{D}(t) < n^* \\ 0 & \text{otherwise} \end{cases} \quad (3.3)$$

$$\partial_t \mathcal{A}(t) = \mathcal{D}(t) p(t) \mu(t) + \theta_2 \mathcal{W}(t) \quad (3.4)$$

$$\partial_t \mathcal{F}(t) = \mathcal{D}(t) (1 - p(t)) \mu(t). \quad (3.5)$$

Arrivals enter the waiting population  $\mathcal{W}$  with rate  $\lambda$  or directly the download-ing population  $\mathcal{D}$ , if the number of slots  $n^*$  is not full. If the slots are full, waiting users simply proceed to the downloading state with rate  $\tilde{\mu} = \mu \mathcal{D}$ , which does not depend on  $\mathcal{W}$ . After entering state  $\mathcal{D}$ , the client remains in this state until he either fully downloads the file and enters the finished state  $\mathcal{F}$  or he aborts the download when the download duration exceeds his patience threshold  $\Theta_1$ .

The latter is expressed by entering abort state  $\mathcal{A}$ . In both cases the transitions are performed at rate  $\mu$  multiplied with a probability  $p$  (when the download fails) or  $1 - p$  in the case of success. An abort occurs when the patience of the downloading user is exceeded either during downloading or waiting. The patience in this model is characterized by the exponential random variables  $\Theta_1$  with rate  $\theta_1$  during downloading and  $\Theta_2$  with rate  $\theta_2$  during waiting, while the downloading time is exponentially distributed as well with rate  $\psi = C(t)/E[f_s]$ . The variable  $C(t)$  denotes the time-dependent capacity per user, i.e.,  $C(t) = C/\mathcal{D}(t)$ , and  $E[f_s]$  is the mean file size. Thus, the probability that the patience is exceeded during downloading at time  $t$  can be expressed as

$$p(t) = \frac{\theta_1}{\theta_1 + \psi} = \frac{\mathcal{D}(t) E[f_s]}{\mathcal{D}(t) E[f_s] + CE[\Theta_1]}. \quad (3.6)$$

Note that in the case of a single downloading state  $\mathcal{D}$ , exponential file sizes  $f_s$  and thus exponentially distributed rates  $\mu$  are assumed. If we consider Erlang- $k$  distributed file sizes as obtained in our measurements, the state  $\mathcal{D}$  must be expanded to several intermediate states  $D_0, D_1, \dots, D_k$ .

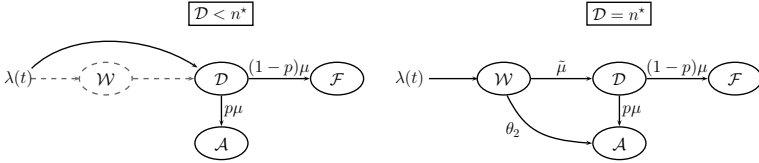


Figure 3.4: Fluid model state transitions

### 3.3.3 Derivation of the Stationary Sojourn Time Distribution

In the following, we use a Markov Model to investigate the stationary sojourn time distribution. Our objective is to derive the time needed for an arbitrary customer to successfully complete the download of a file. We call this the *sojourn time* of a customer. The stationary sojourn time distribution allows us to understand the key influence factors of the system and to dimension the system properly. In order to simplify the resulting equations, the capacities in our model are normalized by  $C$ , thus without loss of generality the normalized server capacity is 1 and the access bandwidth of a client is  $R_d/C$ , where  $R_d$  is the maximum physical download bandwidth of the customer and  $C$  the real capacity at the server. The file size is randomly distributed following an exponential distribution of parameter  $\mu$  reflecting the measurement results described and normalized by the system capacity, i.e.  $\mu = C/E[f_s]$ .

The changeover point  $N^*$  reflects whether the user's access bandwidth or the server's capacity is the limiting factor. Let first  $N^*$  be such that

$$\frac{C}{N^* - 1} > R \quad \text{and} \quad \frac{C}{N^*} \leq R. \quad (3.7)$$

We assume that  $N^* < n^*$ , otherwise the resulting model is trivial.

We are interested in computing the exact sojourn time a customer spends in the system in order to completely download the entire file. The method we will apply actually consists of solving a system of differential equations and is inspired by the work of Sericola et al. in [120] or Masuyama and Takine in [84]. However, our system is more complex since it integrates impatience and different service behaviors according to the actual number of customers in the system. We first derive equations for the remaining sojourn time distribution of an observed customer, then we establish the stationary distribution of the number of customers in the system and finally obtain the stationary distribution of the sojourn time of an arbitrary customer.

### Remaining Sojourn Time

When the system counts less than  $N^*$  customers, any new customer is served at an average speed of  $(R_d/C\mu)^{-1}$ . Obviously, in this case the bandwidth is not shared. However, as soon as the system counts more than  $N^*$  clients, say  $n_d$  clients, the bandwidth of our observed customer shrinks to  $(\mu/n_d)^{-1}$ , on average. It is important to know exactly when this happens.

It is clear that as soon as our observed customer is in service, i.e. downloading the file, he will continue until either the file is completely downloaded or the customer becomes impatient and leaves the system before completion. Nevertheless, we still need to take the arriving customers following his arrival into account, even if they do not directly interfere with our observed customer's sojourn time, i.e. if they are waiting customers. Indeed, these waiting customers will eventually become downloading customers. Accordingly, the service rate will remain at a level  $\mu/n^*$ , even when a customer in service leaves the system.

Imagine now our customer entering the system counting already more than  $n^*$  clients. Our observed customer becomes, thus, a waiting customer. It is then important to know exactly how many clients were waiting prior to his arrival in order to exactly determine when his service will start. In the same manner we also need to record how many clients arrive after his arrival, in order to determine the subsequent speed of service.

Owing to the necessity to keep track of the actual number of the other clients in the system and, therefore, to differentiate between the system behavior, we define the following three different conditional random variables.

For  $K \in \{0, 1, \dots, n^* - 1\}$ , we define the random variable  $W(K, 1)$  as the remaining time a customer needs in order to completely download the file he requested, given that there are  $K$  customers in service. For  $K \in \{n^*, \dots, \underline{N} - 1\}$  and  $N \in \{n^* + 1, \dots, \underline{N}\}$ , we define the two random variables  $W(K, 1)$  and  $W(K, 0, N)$  according to whether our customer is in service or not. The random variable  $W(K, 1)$  is the remaining sojourn time of the observed customer when the system counts  $K$  customers. The random variable  $W(K, 0, N)$  is the

remaining sojourn time of the observed customer when there are  $K$  customers in the system and our observed client is waiting at position  $N$  to be served, i.e.  $N - n^*$  customers need to leave the system before our customer starts downloading the file. Note that these  $N - n^*$  clients have to be clients in service or waiting customers located in front of our observed customer in the queue.

We denote by  $\mathcal{E}(x)$  an exponentially distributed random variable with mean  $1/x$  and formulate Theorem 3.1, considering all possible cases that can occur.

**Theorem 3.1.** *For  $K \in \{0, \dots, N^* - 1\}$ , the remaining sojourn time of a customer  $W(K, 1)$  in a system that counts  $K$  customers is such that:*

$$W(K, 1) = \begin{cases} \mathcal{E}(\Lambda(K)) \\ \quad w.p. \mu R_d / C (\Lambda(K))^{-1} \\ \mathcal{E}(\Lambda(K)) + W(K - 1, 1) \\ \quad w.p. K(\mu R_d / C + \theta_1) (\Lambda(K))^{-1} \\ \mathcal{E}(\Lambda(K)) + W(K + 1, 1) \\ \quad w.p. \lambda (\Lambda(K))^{-1} \end{cases} \quad (3.8)$$

where  $\Lambda(K)$  is the exponentially distributed rate at which the next event occurs that changes the system state:

$$\Lambda(K) = (K + 1) (\mu R_d / C + \theta_1) + \lambda. \quad (3.9)$$

When  $K$  belongs to  $\{N^*, \dots, n^* - 1\}$ , we have:

$$W(K, 1) = \begin{cases} \mathcal{E}(\Lambda(K)) \\ \quad w.p. \mu ((K + 1) \Lambda(K))^{-1} \\ \mathcal{E}(\Lambda(K)) + W(K - 1, 1) \\ \quad w.p. (K / (K + 1) \mu + K \theta_1) (\Lambda(K))^{-1} \\ \mathcal{E}(\Lambda(K)) + W(K + 1, 1) \\ \quad w.p. \lambda (\Lambda(K))^{-1} \end{cases} \quad (3.10)$$



where

$$\Lambda(K) = \mu + (K + 1) \theta_1 + \lambda. \quad (3.11)$$

When  $K \in \{n^*, \dots, \underline{N} - 2\}$ , the remaining sojourn time of the observed customer, assuming he is already in service, is:

$$W(K, 1) = \begin{cases} \mathcal{E}(\Lambda(K)) \\ \text{w.p. } \mu(n^* \Lambda(K))^{-1} \\ \mathcal{E}(\Lambda(K)) + W(K - 1, 1) \\ \text{w.p. } \frac{(n^* - 1)/n^* \mu + (n^* - 1)\theta_1 + (K + 1 - n^*)\theta_2}{\Lambda(K)} \\ \mathcal{E}(\Lambda(K)) + W(K + 1, 1) \\ \text{w.p. } \lambda(\Lambda(K))^{-1} \end{cases} \quad (3.12)$$

where

$$\Lambda(K) = \mu + n^* \theta_1 + (K + 1 - n^*) \theta_2 + \lambda. \quad (3.13)$$

When the observed customer is not in service and assuming he is at position  $n^* + 1$ , we have:

$$W(K, 0, n^* + 1) = \begin{cases} \mathcal{E}(\Lambda(K)) + W(K - 1, 1) \\ \text{w.p. } (\mu + n^* \theta_1)(\Lambda(K))^{-1} \\ \mathcal{E}(\Lambda(K)) + W(K - 1, 0, n^* + 1) \\ \text{w.p. } (K - n^*)\theta_2(\Lambda(K))^{-1} \\ \mathcal{E}(\Lambda(K)) + W(K + 1, 0, n^* + 1) \\ \text{w.p. } \lambda(\Lambda(K))^{-1}. \end{cases} \quad (3.14)$$

In case the observed customer is at position  $N$  with  $N \in \{n^* + 2, \dots, K + 1\}$ , we have:

$$W(K, 0, N) = \begin{cases} \mathcal{E}(\Lambda(K)) + W(K - 1, 0, N - 1) \\ \text{w.p. } \frac{\mu + n^* \theta_1 + (N - n^* + 1) \theta_2}{\Lambda(K)} \\ \mathcal{E}(\Lambda(K)) + W(K - 1, 0, N) \\ \text{w.p. } (K + 1 - N) \theta_2 (\Lambda(K))^{-1} \\ \mathcal{E}(\Lambda(K)) + W(K + 1, 0, N) \\ \text{w.p. } \lambda (\Lambda(K))^{-1}. \end{cases} \quad (3.15)$$

In both cases described by Eqn. (3.14) and Eqn. (3.15), the term  $\Lambda(K)$  is used as defined in Eqn. (3.13). When  $K$  equals  $\underline{N} - 1$ , the remaining sojourn time of the observed customer, already in service, is:

$$W(\underline{N} - 1, 1) = \begin{cases} \mathcal{E}(\Lambda(\underline{N} - 1)) \\ \text{w.p. } \mu (n^* \Lambda(\underline{N} - 1))^{-1} \\ \mathcal{E}(\Lambda(\underline{N} - 1)) + W(\underline{N} - 2, 1) \\ \text{w.p. } \frac{(n^* - 1)/n^* \mu + (n^* - 1) \theta_1 + (\underline{N} - n^*) \theta_2}{\Lambda(\underline{N} - 1)} \end{cases} \quad (3.16)$$

where

$$\Lambda(\underline{N} - 1) = \mu + n^* \theta_1 + (\underline{N} - n^*) \theta_2. \quad (3.17)$$

When the observed customer is not in service, then assuming he is at position  $n^* + 1$ , we have:

$$W(\underline{N} - 1, 0, n^* + 1) = \begin{cases} \mathcal{E}(\Lambda(\underline{N} - 1)) + W(\underline{N} - 2, 1) \\ \text{w.p. } (\mu + n^* \theta_1) (\Lambda(\underline{N} - 1))^{-1} \\ \mathcal{E}(\Lambda(\underline{N} - 1)) + W(\underline{N} - 2, 0, n^* + 1) \\ \text{w.p. } (\underline{N} - (n^* + 1)) \theta_2 (\Lambda(\underline{N} - 1))^{-1}. \end{cases} \quad (3.18)$$

In case the observed customer is at position  $N$  with  $N \in \{n^* + 2, \dots, \underline{N}\}$ , we have:

$$W(\underline{N}-1, 0, N) = \begin{cases} \mathcal{E}(\Lambda(\underline{N}-1)) + W(\underline{N}-2, 0, N-1) \\ \quad \text{w.p. } (\mu + n^*\theta_1 + (N - (n^* + 1))\theta_2) (\Lambda(\underline{N}-1))^{-1} \\ \mathcal{E}(\Lambda(\underline{N}-1)) + W(\underline{N}-2, 0, N) \\ \quad \text{w.p. } (\underline{N} - N)\theta_2(\Lambda(\underline{N}-1))^{-1}. \end{cases} \quad (3.19)$$

where the definition of  $\Lambda(K)$  found in Eqn. (3.17) is used in Eqn. (3.18) and Eqn. (3.19).

**Proof.** We only establish a formal proof of Eqn. (3.14), since the proof for all other equations can be obtained following a similar argument.

We compute the remaining sojourn time of an observed customer, given that the observed customer is in a system counting  $K$  other clients with  $K \geq n^*$ . Moreover, we assume that our tagged customer's service has not yet started. However, as soon as one of the  $n^*$  clients already in service leaves the system, our observed customer will begin with his download. We, thus, compute the remaining sojourn time of our observed customer given that he is at position  $n^* + 1$ . In this case, because of the memoryless property of the exponential distribution, the next event (arrival or departure) takes place after an exponentially distributed time with parameter  $\Lambda(K)$ . We have, as stated in Eqn. (3.13):

$$\Lambda(K) = \mu + n^*\theta_1 + (K + 1 - n^*)\theta_2 + \lambda. \quad (3.20)$$

Indeed, we may observe one of the  $n^*$  clients in service, either finishing their download (at a rate  $\mu/n^*$ ), or becoming impatient (at a rate  $\theta_1$ ). The remaining customers including our observed customer, thus, those  $K + 1 - n^*$  customers that have not yet started downloading their file, may become impatient at a rate  $\theta_2$ . Of course, we may still observe the arrival of a new customer at a rate  $\lambda$ .

If the departure of a served customer occurs, then our observed customer will become served. This happens with the probability  $(\mu + n^*\theta_1) (\Lambda(K))^{-1}$  and

corresponds to the first case in Eqn. (3.14). The second case corresponds to where a waiting customer becomes impatient, leaving our observed customer still waiting for service, but the system counts one customer less. This happens with probability  $(K - n^*) \theta_2 (\Lambda(K))^{-1}$ . The last case corresponds to the arrival of a new user. ■

Let us remark the following point. Since we are interested in computing the sojourn time of a customer, defined as the total time needed to download his desired file, we only consider successfully completed downloads and the event that our observed customer leaves the system due to impatience is not taken into account in any of the equations in Theorem 3.1.

Let  $W$  be the remaining sojourn time of a typical customer. We define the following conditional probabilities. For  $K \in \{0, \dots, n^* - 1\}$

$$\begin{aligned} R(y | K, 1) &= P [W > y | X = K, S = 1] \\ &= P [W(K, 1) > y], \end{aligned} \tag{3.21}$$

thus, the complementary remaining sojourn time distribution of a customer in service ( $S = 1$ ) in a system with  $K$  users ( $X = K$ ). For  $K \in \{n^*, \dots, \underline{N} - 1\}$

$$\begin{aligned} R(y | K, 1) &= P [W > y | X = K, S = 1] \\ &= P [W(K, 1) > y] \end{aligned} \tag{3.22}$$

$$\begin{aligned} R(y | K, 0, N) &= P [W > y | X = K, S = 0, P = N] \\ &= P [W(K, 0, N) > y], \end{aligned} \tag{3.23}$$

where  $N \in \{n^* + 1, \dots, K + 1\}$  and  $P$  is the position of the observed user in the queue of waiting users, since  $S = 0$  indicates that our observed customer is not yet in service.

We now establish the system of differential equations in the next theorem, which is given without proof.

**Theorem 3.2.** *The conditional complementary probability distributions  $R(y | K, 1)$  and  $R(y | K, 0, N)$  respect the following differential equations.*

If  $0 \leq K < N^*$ :

$$\begin{aligned} \partial_y R(y | K, 1) &= -\Lambda(K)R(y | K, 1) + K(R/C\mu + \theta_1)R(y | K - 1, 1) \\ &\quad + \lambda R(y | K + 1, 1). \end{aligned} \tag{3.24}$$

If  $N^* \leq K < n^*$ :

$$\begin{aligned} \partial_y R(y | K, 1) &= -\Lambda(K)R(y | K, 1) \\ &\quad + \left( \frac{K}{K+1}\mu + K\theta_1 \right) R(y | K - 1, 1) + \lambda R(y | K + 1, 1). \end{aligned} \tag{3.25}$$

If  $n^* \leq K < \underline{N} - 1$ :

$$\begin{aligned} \partial_y R(y | K, 1) &= -\Lambda(K)R(y | K, 1) + \lambda R(y | K + 1, 1) \\ &\quad + \left( \frac{n^* - 1}{n^*}\mu + (n^* - 1)\theta_1 + (K + 1 - n^*)\theta_2 \right) R(y | K - 1, 1). \end{aligned} \tag{3.26}$$

$$\begin{aligned} \partial_y R(y | K, 0, n^* + 1) &= -\Lambda(K)R(y | K, 0, n^* + 1) \\ &\quad + (\mu + n^*\theta_1)R(y | K - 1, 1) + (K - n^*)\theta_2 R(y | K - 1, 0, n^* + 1) \\ &\quad + \lambda R(y | K + 1, 0, n^* + 1). \end{aligned} \tag{3.27}$$

Moreover, if  $n^* + 1 < N \leq K + 1$ :

$$\begin{aligned} \partial_y R(y | K, 0, N) &= -\Lambda(K)R(y | K, 0, N) + \lambda R(y | K + 1, 0, N) \\ &+ (\mu + n^* \theta_1 + (N - 1 - n^*) \theta_2) R(y | K - 1, 0, N - 1) \\ &+ (K + 1 - N) \theta_2 R(y | K - 1, 0, N). \end{aligned} \quad (3.28)$$

Finally, we have

$$\begin{aligned} \partial_y R(y | \underline{N} - 1, 1) &= -\Lambda(\underline{N} - 1) R(y | \underline{N} - 1, 1) \\ &+ \left( \frac{n^* - 1}{n^*} \mu + (n^* - 1) \theta_1 + (\underline{N} - n^*) \theta_2 \right) R(y | \underline{N} - 2, 1), \end{aligned} \quad (3.29)$$

$$\begin{aligned} \partial_y R(y | \underline{N} - 1, 0, n^* + 1) &= -\Lambda(\underline{N} - 1) R(y | \underline{N} - 1, 0, n^* + 1) \\ &+ (\underline{N} - 1 - n^*) \theta_2 R(y | \underline{N} - 2, 0, n^* + 1) \\ &+ (\mu + n^* \theta_1) R(y | \underline{N} - 2, 1) \end{aligned} \quad (3.30)$$

and

$$\begin{aligned} \partial_y R(y | \underline{N} - 1, 0, N) &= -\Lambda(\underline{N} - 1) R(y | \underline{N} - 1, 0, N) \\ &+ (\underline{N} - N) \theta_2 R(y | \underline{N} - 2, 0, N) \\ &+ (\mu + n^* \theta_1 + (N - 1 - n^*) \theta_2) R(y | \underline{N} - 2, 0, N - 1) \end{aligned} \quad (3.31)$$

when  $n^* + 1 < N \leq \underline{N}$ .

For  $n^* \leq i < \underline{N}$ , we define the vectors  $\mathbf{R}(y, i)$  of size  $(i - n^* + 2) \times 1$  as follows:

$$\mathbf{R}(y, i) = \begin{pmatrix} R(y | i, 0, n^* + 1) \\ R(y | i, 0, n^* + 2) \\ \dots \\ R(y | i, 0, i + 1) \\ R(y | i, 1) \end{pmatrix}. \quad (3.32)$$

Accordingly, we define the vector  $\mathbf{R}(y)$  as composed as follows:

$$\mathbf{R}(y) = \begin{pmatrix} R(y | 0, 1) \\ R(y | 1, 1) \\ R(y | n^* - 1, 1) \\ \mathbf{R}(y, n^*) \\ \mathbf{R}(y, n^* + 1) \\ \dots \\ \mathbf{R}(y, \underline{N} - 1) \end{pmatrix}. \quad (3.33)$$

which has the dimension  $\frac{1}{2}((n^*)^2 - (2\underline{N} + 1)n^* + 3\underline{N} + \underline{N}^2)$ .

The system of differential equations in Theorem 3.2 can now be written as

$$\partial_y \mathbf{R}(y) = A\mathbf{R}(y) \quad \text{and} \quad \mathbf{R}(0) = \mathbf{1}, \quad (3.34)$$

where  $\mathbf{1}$  is a vector of appropriate size consisting of 1. The matrix  $A$  is defined as composed of the following blocks:

$$A = \begin{pmatrix} C_0 & A_0 & 0 & \dots & 0 \\ B_1 & C_1 & A_1 & \dots & 0 \\ 0 & B_2 & C_2 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & C_{\underline{N}-1} \end{pmatrix}, \quad (3.35)$$

where for  $0 \leq i < n^*$

$$C_i = -\Lambda(i) \quad (3.36)$$

$$A_i = \lambda \quad (3.37)$$

$$B_i = \begin{cases} i(R_d/C\mu + \theta_1) & \text{if } 0 < i < N^* \\ i/(i+1)\mu + i\theta_1 & \text{if } N^* \leq i < n^* \end{cases} \quad (3.38)$$

When  $i$  belongs to  $\{n^*, \dots, \underline{N} - 1\}$ , we have  $C_i = -\Lambda(i)I$  with  $I$  the identity matrix of appropriate size  $(i - n^* + 2) \times (i - n^* + 2)$ . We have

$$B_{n^*} = \begin{pmatrix} \mu + n^*\theta_1 \\ \frac{n^*-1}{n^*}\mu + (n^* - 1)\theta_1 + \theta_2 \end{pmatrix} \quad (3.39)$$

$$B_{n^*+1} = \begin{pmatrix} \theta_2 & \mu + n^*\theta_1 \\ \mu + n^*\theta_1 + \theta_2 & 0 \\ 0 & \frac{n^*-1}{n^*}\mu + (n^* - 1)\theta_1 + 2\theta_2 \end{pmatrix} \quad (3.40)$$

For  $2 \leq i \leq \underline{N} - 1 - n^*$  the matrix  $B_{n^*+i}$  is of size  $(i+2) \times (i+1)$  with

$$\begin{aligned} B_{n^*+i}(j, j) &= (i - (j - 1))\theta_2 && \text{for } 1 \leq j \leq i \\ B_{n^*+i}(j + 1, j) &= \mu + n^*\theta_1 + j\theta_2 && \text{for } 1 \leq j \leq i \\ B_{n^*+i}(1, i + 1) &= \mu + n^*\theta_1 \\ B_{n^*+i}(i + 2, i + 1) &= \frac{n^*-1}{n^*}\mu + (n^* - 1)\theta_1 + (i + 1)\theta_2, \end{aligned} \quad (3.41)$$

and other elements being equal to 0. For  $0 \leq i \leq (\underline{N} - 1) - n^*$ , the matrix  $A_i$  is a matrix of size  $(i+2) \times (i+3)$ , whose elements are

$$\begin{aligned} A_{n^*+i}(j, j) &= \lambda && \text{for } 1 \leq j \leq i + 1 \\ A_{n^*+i}(i + 2, i + 3) &= \lambda \end{aligned} \quad (3.42)$$

and others elements being equal to 0.



The system of differential equations with initial conditions in Eqn. (3.34) can be easily solved and we get:

$$R(y) = \exp(Ay). \quad (3.43)$$

It has to be noted that  $\exp(Ay)$  is the matrix exponential of the quadratic matrix  $Ay$  of size  $\underline{N} \times \underline{N}$ , i.e.  $\exp(Ay) = \sum_{i=0}^{\infty} \frac{1}{i!} (Ay)^i$ . This series always converges, so the exponential of  $Ay$  is well-defined. Algorithms for the fast computation of power series solutions of systems of differential equations can be found in [146].

### Stationary Distribution of the Number of Users in the System

Our objective is to compute the stationary distribution of the time a customer needs in order to completely download a file. When a customer enters the system, he may find the system already occupied with  $K < \underline{N}$  customers. This section aims at computing the stationary distribution of the number of customers in the system at the arrival instant of the observed customer. Due to the PASTA property, this stationary distribution is simply equal to the stationary distribution of the number of customers in the system at any time.

Let  $\{X(t); t \in \mathbb{R}^+\}$  be the Markov process counting the number of customers in the system. As previously mentioned, the corresponding stationary random variable is given by  $X$ . We denote by the vector  $\pi$  the corresponding stationary distribution, that is

$$\pi(K) = P[X = K], \quad (3.44)$$

with  $K \in \{0, \dots, \underline{N}\}$ .

We define  $Q$  as the generator associated to the process  $\{X(t); t \in \mathbb{R}^+\}$ . The elements of  $Q$  are:

$$\begin{aligned}
 Q(i, i+1) &= \lambda && \text{for } 0 \leq i \leq \underline{N} - 1 \\
 Q(i, i-1) &= i R_d / C\mu + i \theta_1 && \text{for } 1 \leq i \leq N^* \\
 &= \mu + i \theta_1 && \text{for } N^* + 1 \leq i \leq n^* \\
 &= \mu + n^* \theta_1 + (i - n^*) \theta_2 && \text{for } n^* + 1 \leq i \leq \underline{N}
 \end{aligned} \tag{3.45}$$

The diagonal elements of  $Q$  are such that  $Q\mathbf{1} = \mathbf{0}$ , with  $\mathbf{0}$  being a vector consisting of entries 0 and of appropriate size. Other elements of  $Q$  are zeros. Clearly, the process  $\{X(t); t \in \mathbb{R}^+\}$  is a birth-and-death process. Accordingly, let  $\rho_i$  be

$$\begin{aligned}
 \rho_i &= \lambda / ((i+1) R_d / C\mu + (i+1) \theta_1) && \text{for } 0 \leq i < N^* \\
 &= \lambda / (\mu + (i+1) \theta_1) && \text{for } N^* \leq i < n^* \\
 &= \lambda / (\mu + n^* \theta_1 + (i+1 - n^*) \theta_2) && \text{for } n^* \leq i < \underline{N} - 1.
 \end{aligned} \tag{3.46}$$

We obtain the stationary probabilities for  $K \in \{1, \dots, \underline{N}\}$

$$\pi(K) = \pi(0) \prod_{i=0}^{K-1} \rho_i \quad \text{with} \quad \pi(0) = \left( 1 + \sum_{K=1}^{\underline{N}} \prod_{i=0}^{K-1} \rho_i \right)^{-1}. \tag{3.47}$$

### Stationary Total Sojourn Time

When a customer enters the system, he may either be immediately served or is placed last in the queue depending on the current number of customers present in the system. The complementary total sojourn time distribution of a customer respects the following equation:

$$P[W > y] = \sum_{K=0}^{n^*-1} \pi(K) R(y | K, 1) + \sum_{K=n^*}^{\underline{N}-1} \pi(K) R(y | K, 0, K+1). \tag{3.48}$$

For  $i \in \{0, \dots, \underline{N} - 1 - n^*\}$ , we define the vectors  $\underline{\pi}(i)$  as:

$$\underline{\pi}(i) = \mathbf{e}(i + 2, i + 1) \pi(n^* + i), \quad (3.49)$$

where  $\mathbf{e}(i, j)$  is a row vector of size  $i$  full of 0's except element  $j$  which is equal to 1. Accordingly, we define  $\mathbf{\Pi}$  as composed of

$$\mathbf{\Pi} = \left( \pi(0) \quad \pi(1) \quad \dots \quad \pi(n^* - 1) \quad \underline{\pi}(n^*) \quad \dots \quad \underline{\pi}(\underline{N} - 1) \right). \quad (3.50)$$

Then, using Eqn. (3.43), we obtain the complementary distribution of the user's sojourn time as

$$P[W > y] = \mathbf{\Pi} \exp(Ay) \mathbf{1}. \quad (3.51)$$

For numerical results, we use an approximation using Krylov subspace projection techniques to obtain the sojourn time distribution  $W = \mathbf{\Pi} \exp(Ay)$ . It is available as a software package that provides matrix exponential routines for small dense or very large sparse matrices [60]. It does not compute the matrix exponential in isolation but instead, it computes directly the action of the exponential operator on the operand vector. This way of doing so allows for addressing large sparse problems and improves computational speed significantly.

### 3.3.4 Understanding the Key Influence Factors

In this section we will provide some numerical results and briefly discuss the influence of some of the important parameters on the system behavior. Let us consider an OTR mirror server site with a total capacity of  $C = 20$  Mbps. Note that while the analytical Markov model used rates normalized by the server's capacity, we give absolute values here, as we are interested in the actual downloading durations and they are more meaningful for verifying the plausibility of the results. The average file size  $E[f_s]$  is 359.87 MByte and the maximum download rate  $R_d$  of all users is 4 Mbps as specified by the ITU G.992.2 standard for ADSL Lite. Since we assume this is a mirror site operated by a private person,

it is reasonable to assume that the maintainer only limits access to a relatively small number of concurrently served downloads, e.g.  $n^* = 10$  and  $\underline{N} = 20$ , due to the following consideration. Many existing mirror servers indicate the current queue length and the expected waiting time until a download slot will become free. In the above scenario, a user at the first position of the waiting queue would have to wait 20 min and at the last (i.e. 10th) position would require a waiting time of 200 min. A requesting user who would be facing to wait so long before service would obviously select a different mirror site beforehand. As further parameters, if not indicated otherwise, we assume a request arrival rate of  $\lambda = 10^{-2}$  requests/s as well as the impatience thresholds of  $\theta_1^{-1} = 2$  h and  $\theta_2^{-1} = 1$  h for downloading and waiting users, respectively. Furthermore, we verified the accuracy of our numerical implementation by simulations.

### **Influence of Impatience During Downloading**

Let us first consider the impatience  $\Theta_1$  and its influence on the sojourn time of a successful customer as shown in Figure 3.5. On the left, in Figure 3.5(a), the CDF of the sojourn time as computed from Equation (3.51) is shown with different impatience thresholds for downloading users  $\theta_1^{-1}$ . Darker lines represent smaller values of  $\theta_1^{-1}$ . Obviously, the sojourn time increases when the patience threshold increases. This can be explained by the fact that when  $\theta_1^{-1}$  is small, only small files are actually downloaded and users downloading larger files will have a large tendency to abort their attempts, so on average the sojourn time in the system will be small. This is also suggested when we look at the steady state distribution of the number of users in the system, see Fig. 3.5(b). For all considered values, in particular when  $\theta_1^{-1}$  is small, the probability of finding an empty system upon arrival, i.e.  $\pi(0)$ , is greater than zero. A larger  $\theta_1^{-1}$  shifts the weight of the distribution toward a larger number of customers. In the case of  $\theta_1^{-1} = 60$  min we can see that  $\pi(\underline{N}) > 0$ , leading to blocking of potentially new arrivals.

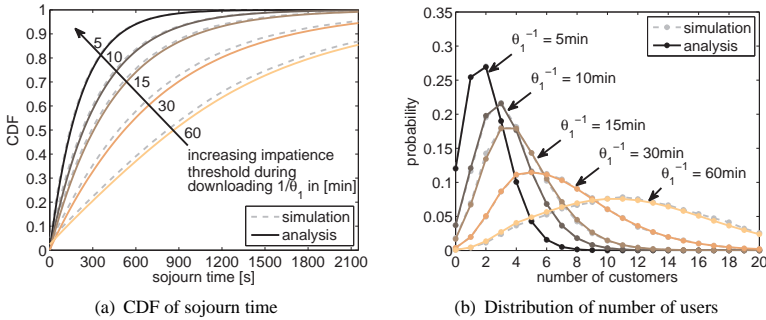


Figure 3.5: Influence of the impatience time  $\Theta_1$  during downloading

Figure 3.5 compares additionally simulation results, indicated by dashed lines, with the numerical results obtained from the analysis, indicated by solid lines. Since we provide here an exact analysis of the sojourn time and the number of customers in the system, we simulated only 1,000 download requests. Otherwise, the curves lie on top of each other and cannot be distinguished. The confidence intervals out of ten simulation runs are omitted, since they are too small to be visible in the figure. Therefore, we skip the simulation results in the following. The comparison of the analysis and simulation was presented here in order to validate the accuracy of the numerical solution as well as the correct implementation of the analytical model.

### Influence of Impatience during Waiting

We now investigate the influence of the impatience of waiting users  $\Theta_2$  on the sojourn time. The numerical results are shown in Fig. 3.6. Note that in contrast to Figure 3.5(a) the left figure (Figure 3.6(a)) now shows the complementary CDF of the sojourn time. It is remarkable that the probabilities  $P[W > 0]$  may be less than 1, if the waiting impatience time  $\theta_2^{-1}$  is large. Again, we can interpret this

result better by looking at the corresponding distributions of customers as shown in Fig. 3.6(b). When  $\theta_2^{-1} = 5$  min the system is already serving  $n^*$  customers and there are on average 3 waiting users in the queue, indicated by the highest probabilities  $\pi(i)$  for  $i \in \{12, 13, 14\}$ . However, since the impatience time is small, queued users quickly leave the system again. On the other hand for larger impatience values of  $\theta_2^{-1}$ , especially when  $\theta_2^{-1} > 15$  min, users wait longer in the queue and the probability is very large to find the system fully occupied. Note that  $P[W = 0] = P[X = \underline{N}]$ , i.e., the probability that the sojourn time is zero is equal to the probability that there are  $\underline{N}$  customers in the system.

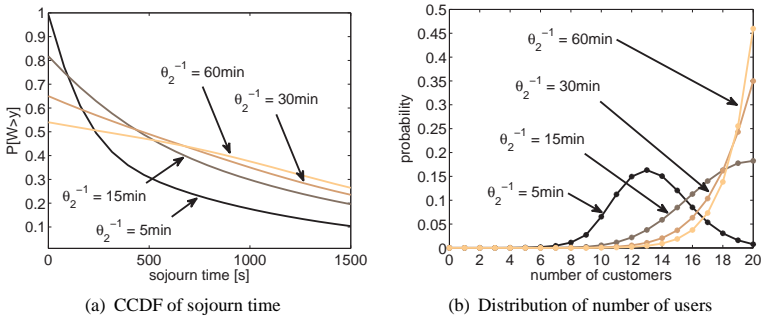


Figure 3.6: Influence of the impatience time  $\Theta_2$  during waiting

### Influence of the Number of Available Download Slots

Finally, we investigate how the number of available download slots affects the sojourn time. We now look at a slightly different scenario, with  $\lambda = 1/80 \text{ s}^{-1}$ ,  $\underline{N} = 42$  and for values of  $n^* = 5, 10, 20, 40$  to emphasize exemplarily the effect. Although the curves for the CDF of the sojourn time do not intersect at the same point, cf. Figure 3.7(a), the intersection point  $y'$  for any two curves lies in a small range between 1600 s and 1700 s. With larger  $n^*$  the download

bandwidth decreases and together with a higher patience while downloading than while waiting, the probability for sojourn times  $y > y'$  beyond this intersection point increase with  $n^*$ . The CDF of the sojourn time displays a similar behavior with blocking for large  $n^*$  as already observed for  $\theta_2$ . Note that a value of  $n^* = 40$  and  $\underline{N} = 42$  means that the maximum waiting queue length is 2.

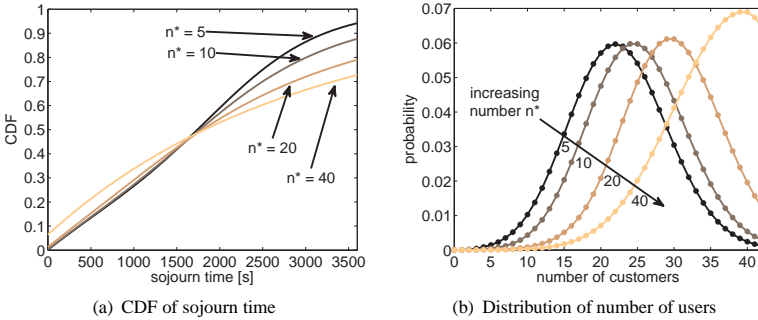


Figure 3.7: Influence of the maximum number of download slots  $n^*$

### Influence of Variation of File Size

The next investigation aims at the optimal dimensioning of the number of download slots  $n^*$  for different file size distributions in flash crowd scenarios. In order to get numerical results for this particular study, we use a discrete event simulation to investigate the OTR server cluster for various distributions. For the file size, we consider deterministic, exponential, Erlang, and lognormal distributions. The parameters of the distributions are chosen such that the average file size corresponds to the measurement values for OTR videos in Section 3.2. For the Erlang and the lognormal distribution, we fitted the CDF of the measurement values to obtain the corresponding parameters for both distributions. Clearly, the deterministic distribution has a coefficient of variation of 0, while

the exponential distribution yields to a value of 1. For the flash crowd scenario, we use  $\lambda(t) = \lambda_0 e^{-\alpha t}$  with  $\alpha = 10^{-3}$  and  $\lambda_0 = 1$ . This means the popularity of the video shows an exponential decay and the number of interested users is limited to  $\int_0^\infty \lambda_0 e^{-\alpha \tau} d\tau = \lambda_0/\alpha$ . In the scenario with the parameters above, we consider 1,000 download requests.

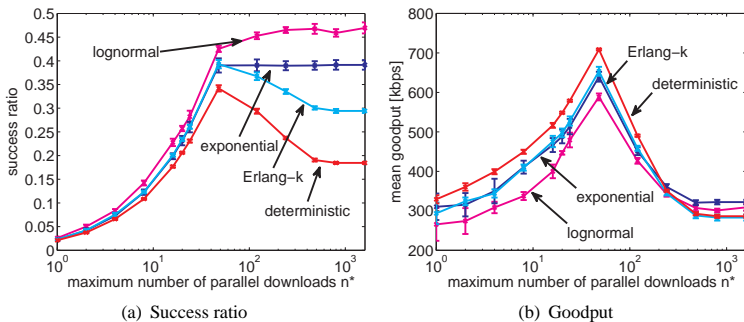


Figure 3.8: Influence of variation of file size

While Figure 3.8(a) shows the success ratio depending on the maximum number  $n^*$  of simultaneously served users, Figure 3.8(b) depicts the average goodput in kbps. Both figures illustrate the influence of the skewness on the system behavior. It is remarkable that for deterministic and Erlang-distributed file sizes a maximum success ratio exists, whereas for exponential and lognormal the success ratio remains nearly constant when  $n^* > \lfloor C/R_d \rfloor$ . However, this is caused by the fact that with a higher skewness, smaller files are downloaded more often. In all four cases the goodput is highest at  $\hat{n} = \lfloor C/R_d \rfloor$ , as can be seen from Figure 3.8(b). The goodput is defined as the ratio of the file size and the download time for successful downloads. For larger  $n^*$  the system capacity is wasted due to longer download times caused by the processor sharing discipline and the aborts due to the user's impatience.



As a result of this investigation, we see that  $n^* = \hat{n}$  leads to optimal performance in a homogeneous scenario, where all users have the same access bandwidth. In this case, the available resources are efficiently utilized and the aborts of impatient users are minimized. In a heterogeneous system, however, we should follow a dynamic approach to achieve both goals, similar to the proposed cooperation strategy for the utilization of scarce resources of a P2P CDN in a heterogeneous cellular network in Section 2.3.5.

### 3.4 Pollution of P2P Content Distribution Service

Recordings of OTR can alternatively be downloaded via eDonkey or BitTorrent P2P file sharing networks. The efficient and cost-effective way for distributing contents makes P2P interesting for services like OTR with a huge amount of video contents. However, the file is not offered at a single, trusted server location, but by multiple sharing peers. Thus, peers may offer a corrupted version of a file or parts of it. This is referred to as pollution. In the original version of eDonkey, error detection is done after all blocks of a chunk have been received and the complete chunk is discarded in case of an error. As a result, the download of the entire file is prolonged. However, in more recent versions of eDonkey clients, e.g. eMule, the Intelligent Corruption Handling (ICH) mechanism is implemented that performs the error detection on smaller data units than chunks and which we define in the following as segments. Instead of discarding the complete chunk when at least one corrupted block is received, only all blocks of the damaged segment need to be re-requested. The actual size of a segment depends on the ICH mechanism. In our proposed model, such mechanisms can be easily taken into account, since we model the actual transmission of data on block level.

In this section, we want to model these fake peers and evaluate their impact on the performance of an eDonkey-like P2P file sharing system and the QoE of their users. Since the file dissemination is disturbed, the download process may be prolonged and the user may abort download requests. Therefore, an appropriate model is required which takes into account the user behavior with respect to

pollution and impatience. In addition, we want to investigate as well flash crowd effects. Therefore, we derive an epidemic model for the file diffusion in eDonkey like P2P networks. We start with a simple SIR (Susceptible-Infected-Recovered) model from biology and refine this to describe file sharing properly. After that, we introduce pollution in our model which is finally described by a differential equations system (DES).

The numerical results of the DES can be interpreted for two different scenarios. Firstly, the service provider wants to disseminate contents via P2P. Malicious peers may disturb the service by pollution and degrade the QoE of the other users. Secondly, the service provider wants to disturb illegal dissemination of copyright-protected content via other file sharing platforms. This can be realized by polluting such P2P systems. In that case, the service provider has to know how many fake peers are required to heavily disturb the system, such that the peers get impatient and stop downloading via P2P. In other words, the service provider has to dimension the required resource for pollution. The results in this section can be applied for both scenarios.

#### 3.4.1 Epidemic Model of File Diffusion

In the following, let us consider a basic epidemic model for P2P file sharing. In general, the epidemic model is used to describe the progress of an epidemic in a population. It categorizes the population in groups depending on their state. A commonly used approach is the SIR model [156]. SIR is an abbreviation for the states that are taken during the course of the spread of the disease. At first, there are *susceptibles*, which are users that can be possibly infected with a certain rate. When they are contacted with the disease, they move to the state of *infectives* and can pass the disease on to other members of the susceptible population. Finally, there is the *removed* population, consisting of users who have either fatally suffered from the disease or have recovered and become immune to it. In either case, they cannot get infected by the disease again. In the basic SIR model the total population  $N$  remains constant.

### Analogy of P2P to Biological SIR Model

We start describing the basic underlying biological model and show the commonalities with P2P file sharing. Although there are various analogies between both models, we will see that simply applying an SIR model is insufficient due to the complexity of the P2P file sharing applications. However, the principle time-dynamic modeling technique from biology will be maintained and we are able to consider cases that are not in steady state.

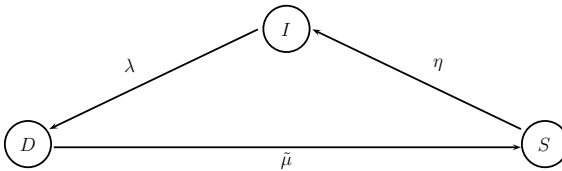


Figure 3.9: *Simple IDS state space*

Let us now define a model similar to SIR in the context of file sharing. We denote the number of susceptibles as *idle peers*  $I(t)$  at time  $t$ . From this set, the file requests are generated with a rate of  $\lambda(t)$ , which can be a time dependent function or a constant reflecting the popularity of the file, see [10]. Once the peer starts to download the file, he is attributed to the set of *downloading peers*  $D(t)$ .

The download rate  $\tilde{\mu}(t)$  depends on the number of peers sharing the file and the other downloading peers, which all compete for the download bandwidth. Once downloading of the complete file with size  $f_s$  is finished, the peer joins the *sharing peers*  $S(t)$ , that offer the file to the other users. The peer shares the file only for a limited time after which he returns with rate  $\eta$  to the idle peers, see Figure 3.9. Note that this is a rather simplified view for a generic file sharing application, as the detailed mechanism in eDonkey involves downloading and sharing chunks of the file. As we will see later, the sharing of smaller data units also has an impact on the accuracy of the model.

Thus, the dynamic system of the sharing process can be expressed by the equation system given in Eqns. (3.52)-(3.54). In analogy to the SIR model, we will refer to it as the IDS model.

$$\partial_t I(t) = -\lambda(t)I(t) + \eta S(t) \quad (3.52)$$

$$\partial_t D(t) = \lambda(t)I(t) - \tilde{\mu}(t)D(t) \quad (3.53)$$

$$\partial_t S(t) = \tilde{\mu}(t)D(t) - \eta S(t). \quad (3.54)$$

The initial values are  $I(0) = I_0$ ,  $S(0) = S_0$ , and  $D(0) = N - I_0 - S_0$ .

In Eqns. (3.52)-(3.54) we can at first assume a constant request arrival rate  $\lambda$  which is adapted to match a Poisson arrival process and the main problem lies in the determination of the download rate  $\tilde{\mu}(t)$ . Let us define the upload and download rates as  $R_u$  and  $R_d$ , respectively. For the sake of simplicity, we assume homogeneous users with ADSL connections, resulting in rates of  $R_u = 128$  kbps and  $R_d = 768$  kbps. Since eDonkey employs a fair share mechanism for the upload rates, there are on average  $S(t)/D(t)$  peers sharing to a single downloading peer and we multiply this value with  $R_u$  which gives us the bandwidth on the uplink. However, since the downloading bandwidth could be the limiting factor, the resulting effective transition rate  $\mu(t)$  consists of the minimum of both terms divided by the file size  $f_s$ , see Eqn. (3.55).

$$\tilde{\mu}(t) = \frac{1}{f_s} \min \left\{ \frac{S(t) R_u}{D(t)}, R_d \right\}. \quad (3.55)$$

The dynamics of the populations of  $D$  and  $S$  are shown in Figure 3.10 and compared to the mean population sizes, i.e. mean number of peers, obtained from 5,000 simulation runs. We selected  $S_0 = 5,000$ ,  $I_0 = 100$  and a constant  $\lambda$  of 1300 requests per hour. For the sake of simplification we consider at this point  $\eta = 0$ , i.e., all peers remain sharing peers after a completed download and do not leave the system.

When comparing the simulation with the analytical model, we can see that the same general shape matches for  $t > 2000$  s, whereas a problem arises with respect to the accuracy of the model for smaller values of time  $t$ . This can be explained as follows. The transition from  $D$  to  $S$  is performed only after the complete file with fixed size  $f_s$  has been downloaded. The current model using the states  $I$ ,  $D$ , and  $S$ , however, is memoryless and does not take into account the number of bits that have already been downloaded. The transitions between these states are given here as rates indicating the “average” number of transitions per time unit. In reality, the average download rate changes during the downloading process of an individual peer and it is insufficient to consider it a priori as constant for the complete file. While this assumption is generally applied in epidemic modeling of diseases, we wish to provide an enhanced mathematical model by considering a finer granularity. In the following we will, therefore, minimize the error by splitting the macro state  $D$  into  $M$  smaller states corresponding to the number of bits downloaded. We expect that when  $M$  approaches infinity that the error will be reduced to zero.

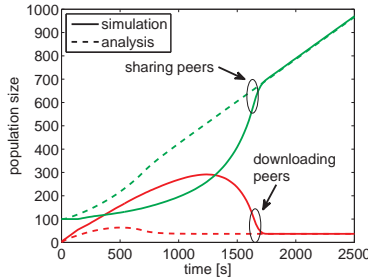


Figure 3.10: Comparison of simulation results with basic IDS model

### Detailed File Sharing Model

For the sake of simplicity, we consider in the following the last chunk of a file which is the most interesting one, as its completion results in the completion of the entire file. The user can then decide whether the whole file is shared or not, i.e., whether the peer becomes a leecher or a seeder. In the following the terms file and last downloaded chunk have the same meaning.

Let us split the file with size  $f_s$  into  $M$  logical units which we will consider individually. Our model thus increases by the states  $D_0, \dots, D_M$ . We can interpret the states  $D_i$  as the state where  $i$  logical units have been successfully downloaded, i.e.,  $D_0$  means that the download is initiated and  $D_M$  indicates a complete download. After reception of each block, the queue mechanism of eDonkey determines the sharing peers from which the next block is downloaded. This involves an update of the download rate  $\mu(t)$  after each logical unit. If we choose the logical unit as blocks, our model is exact and the obtained numerical error is acceptably small as will be shown in more detail later, cf. Figure 3.12(a). The transitions from the states  $D_i$  use a rate  $\mu(t)$  similar to the one described in Eqn. (3.55).

$$\mu(t) = \frac{M}{f_s} \min \left\{ \frac{S(t) R_u}{\sum_{i=0}^{M-1} D_i(t)}, R_d \right\}. \quad (3.56)$$

A further enhancement of the simple model is the introduction of  $p_s$  as the probability of sharing a file. The updated state space with transitions is illustrated in Figure 3.11. After the  $M$ -th logical unit has been downloaded, the peer enters the sharing peers with probability  $p_s$  and returns to the idle state with  $1 - p_s$ . This corresponds to the user leaving the system after downloading (leecher) or downloading it another time again at a later time.

The new equation system is summarized below. The original model given in Section 3.4.1 corresponds to a value of  $M = 1$ . Obviously, the larger  $M$  is, the more accurate is the model, but the computational requirements for solving the equation system increases as well. Finding a good value of  $M$  involves a tradeoff

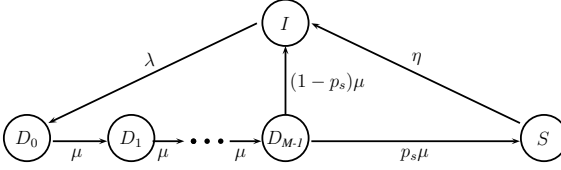


Figure 3.11: Detailed IDS state space

between accuracy and computation speed.

$$\partial_t I(t) = (1 - p_s)\mu(t)D_{M-1}(t) - \lambda(t)I(t) + \eta S(t) \quad (3.57)$$

$$\partial_t D_0(t) = \lambda(t)I(t) - \mu(t)D_0(t) \quad (3.58)$$

$$\partial_t D_i(t) = \mu(t)(D_{i-1}(t) - D_i(t)) \quad \forall 1 \leq i < M \quad (3.59)$$

$$\partial_t S(t) = p_s\mu(t)D_{M-1}(t) - \eta S(t). \quad (3.60)$$

Again, we include the condition – as in the original SIR model – to keep the total population at the index server constant at

$$N = I + \sum_{i=1}^M D_i + S. \quad (3.61)$$

Since the equation system is a closed system, initial values obeying this constraint lead to a constant population. Hence, we assume that  $N = I_0 + S_0$  and  $D_i = 0$  for all  $i$ . The considered values for  $M$  are  $M \in \{1, 18, 53\}$  corresponding to the size of a chunk, a download unit of 540 kB, and a block, respectively. The extended model is compared to the average over  $N = 5,000$  simulation runs in Figure 3.12(a). We can recognize that using a large value of  $M$  greatly improves the accuracy of the model. Throughout the rest of this thesis, we use  $M = 53$  for the numerical results. This means that the size of the logical units in our model is given in blocks.

Note that the task of comparing results averaged from simulation runs to the mathematical model is not fully appropriate. The time is discretized into steps of length  $\delta$  and at each time point  $t_i = i\delta$  the average population size  $\bar{X}(t_i)$  is calculated over the  $N$  simulation runs, i.e.  $\bar{X}(t_i) = \frac{1}{N} \sum_{j=1}^N X_j(t_i)$  with  $X_j(t_i)$  being the population size of simulation run  $j$  at time  $t_i$ . The DES now describes the average behavior of a single evolution over time, depending on the initial values and boundary values. Each individual simulation run matches exactly the shape of the analytical model, however, depending on the random variables can be different in scale, see Figure 3.12(b). When we average over the series of simulation runs, this leads to the different decreasing slope between time 1500 s and 2000 s in Figure 3.12(a).

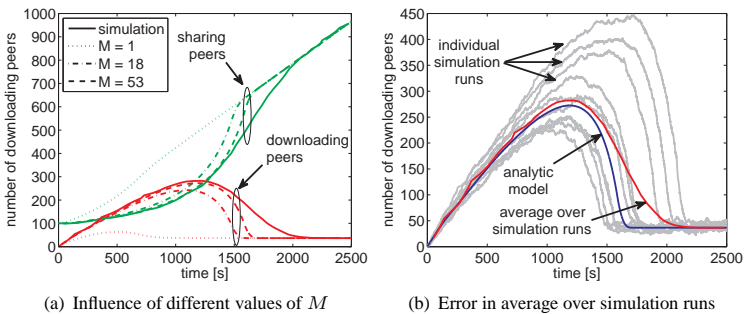


Figure 3.12: *Extended IDS model*



### 3.4.2 Analytic Modeling of Pollution in the P2P Network

So far the model assumed that all peers share correct versions of the file and none is corrupted. Now we will investigate the influence caused by these fake peers, whose cardinality we will denote as  $K$  in the following. The number of fake peers in the system can also be time-dependent, i.e.  $K(t)$ , e.g. in order to relate the degree of pollution to the popularity of the file. We modify the detailed file sharing model to include fake peers and download aborts due to impatience. In addition, we relax the condition of a constant population size and finally end with a flow model of pollution in a P2P file sharing network.

In the P2P model we assume that the file sharing process of a file with size  $f_s$  operates similar to the eDonkey network, see Section 2.1. The sharing itself is performed in units of 9.5 MB, so-called chunks, and the data of each chunk is transferred in blocks of 180 kB. After each chunk is downloaded, it is checked using MD5 hashes and in case an error is detected e.g. due to transmission errors, the chunk is discarded and downloaded again. After all chunks of a file have been successfully downloaded, it is up to the peer if the file is kept as a seeder for other peers to download or if it is removed from share (leecher or free rider). Since the model does not distinguish between specific chunks, it is sufficient to just consider an arbitrary chunk instead of the complete file in the following. Therefore, we consider a file that consists of a single chunk with  $M = 53$  blocks.

#### Description of the Flow Model

The flow model is characterized by a differential equation system describing the transitions between each of the states a peer traverses. Initially, there are only  $S_0$  peers in the system sharing a correct version of the file and  $K$  fake peers. Requests for downloading the file arrive with rate  $\lambda$ . A peer downloads  $M$  units of the file where it has the possibility of reaching a correct version of the data block with probability  $p_b$ . Since we assume an equal probability for reaching a sharing or fake peer,  $p_b$  can be given as in Eqn. (3.62) at time  $t$ .

$$p_b(t) = \frac{S(t)}{S(t) + K(t)}. \quad (3.62)$$

The population of peers with successful downloads of  $i$  units is defined as  $D_i$ . After having successfully downloaded  $M$  data units, an error check is performed and the chunk is discarded in case of an error. If the download of the entire chunk was successful, the peer either shares the file and enters population  $S$  with the *sharing probability*  $p_s$  or enters  $L$  of non-sharing peers with the complementary probability  $1 - p_s$ . On the other hand, if the download attempt of the chunk failed because of downloading at least one block from a fake peer, the peer aborts with probability  $p_a$  and retries the download attempt with  $1 - p_a$ . This means the number of download attempts is geometrically distributed, in case the user downloads from fake peers. The download of  $i$  data units of which at least one is corrupt is represented by state  $F_i$ . The P2P file sharing model with pollution and impatience is depicted in Figure 3.13 showing all populations and their transitions.

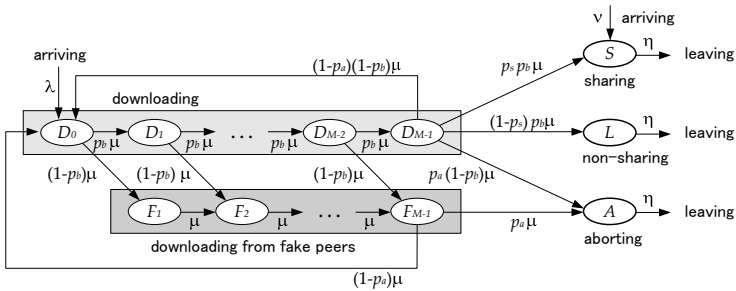


Figure 3.13: Flow diagram of P2P file sharing model

The differential equation system describing the dynamic behavior of each population is given in the following Eqns. (3.63)-(3.69). For the sake of readability, we neglect to note the time-dependency of variables and use  $f$  instead of  $f(t)$ .

$$\partial_t D_0 = \lambda + \mu(1 - p_a) [F_{M-1} + (1 - p_b) D_{M-1}] - \mu D_0 \quad (3.63)$$

$$\partial_t D_i = \mu p_b D_{i-1} - \mu D_i \quad \text{for } i = 1, \dots, M - 1 \quad (3.64)$$

$$\partial_t F_1 = \mu(1 - p_b) D_0 - \mu F_1 \quad (3.65)$$

$$\partial_t F_i = \mu(1 - p_b) D_{i-1} + \mu F_{i-1} - \mu F_i \quad \text{for } i = 2, \dots, M - 1 \quad (3.66)$$

$$\partial_t S = \nu + \mu p_s p_b D_{M-1} - \eta S \quad (3.67)$$

$$\partial_t L = \mu(1 - p_s) p_b D_{M-1} - \eta L \quad (3.68)$$

$$\partial_t A = \mu p_a [F_{M-1} + (1 - p_b) D_{M-1}] - \eta A. \quad (3.69)$$

The other variables that have not yet been discussed are the file request rate  $\lambda$  and the rates for leaving the system  $\eta$ . Furthermore,  $\nu$  is the rate of arrivals of peers that share the file which they obtained from another source than from this network. For peers in the network, we will assume flash crowd arrivals as  $\partial_t \lambda(t) = -\alpha \lambda(t)$  with initial value of  $\lambda(0) = \lambda_0$ . Hence, the flash crowd scenario corresponds to an exponentially decreasing arrival rate with parameter  $\alpha$ .

$$\lambda(t) = \lambda_0 e^{-\alpha t}. \quad (3.70)$$

For the sake of simplicity we assume that a peer decides to leave only if he either has successfully completed the download ( $S$  and  $L$ ) or when he aborts the download attempt ( $A$ ). In  $F_{M-1}$ , the peer may enter the population  $A$  with *abort probability*  $p_a$  or else reattempts. The most crucial variable in the model is the download rate per data unit  $\mu(t)$ . We use the same approximation as in [24] which assumes that if there are enough sharers, the download bandwidth  $R_d$  of a peer will be the limitation, otherwise all requesting peers fairly share the upload

bandwidth  $R_u$  of all sharing peers, see Eqn. (3.71).

$$\mu(t) = \frac{M}{f_s} \min \left\{ \frac{R_u (S(t) + K(t))}{\sum_{i=0}^{M-1} D_i(t) + \sum_{i=1}^{M-1} F_i(t)}, R_d \right\}. \quad (3.71)$$

Note that all variables in the equation system are in fact functions of time resulting in a highly non-stationary behavior. Finally, it should be remarked that the continuous transition rates lead to a slight inaccuracy from non-integer population sizes which do not appear in reality, but reflect the average values.

### Evaluation of the Download Duration

From the solution of the dynamic system in Eqns. (3.63)-(3.69), we can indirectly derive the transmission durations until reaching an absorbing population  $S$ ,  $L$ , or  $A$ . The states  $S$  and  $D_i$  allow from Eqn. (3.71) the computation of the download rates per data unit  $\mu(t)$ . For the computation of the download duration  $\delta(t)$ , let us consider the start of the download attempt of a chunk at time  $t_0$  and a series of time instants  $t_1, \dots, t_M$ . Each  $t_i$  indicates the time at which the downloading of one data unit is completed. Since the transmission rates are with respect to the transmission of a block, the  $t_i$  values can be computed by numerical solution of Eqn. (3.72), beginning with a given  $t_0$ .

$$\int_{t_{i-1}}^{t_i} \mu(t) dt = 1 \quad 1 \leq i \leq M. \quad (3.72)$$

Once the whole chunk is downloaded, we also define this time instant as  $T_j$ ,  $j > 1$  indicating with  $j$  the number of attempts a download attempt was made starting at  $T_0$ . Thus,  $t_0$  is always set to the starting time of a new chunk download and is considered only within the context of a chunk. The relationship between  $\mu(t)$ ,  $t_i$ , and  $T_j$  is illustrated in Figure 3.14.

At time instants  $T_j$  we compute the probability that the chunk was correctly received by considering the possibilities of encountering a fake source at all  $t_i$ .

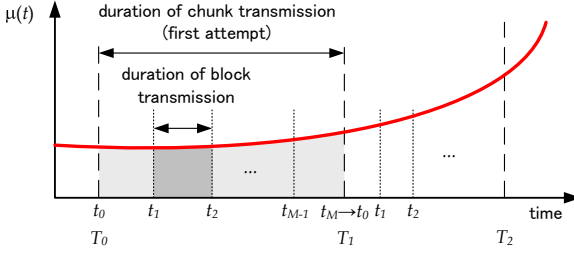


Figure 3.14: Computation of block and chunk transmission durations from  $\mu(t)$

The probability for a correct block  $p_b(t_i)$  at the start of each block download interval  $[t_i, t_{i+1}]$  and the probability  $p_c(t_0)$  of the chunk being correctly received is the product over each of the correct block probabilities beginning at  $t_0$ .

$$p_c(t_0) = \prod_{i=0}^{M-1} p_b(t_i). \quad (3.73)$$

If the chunk was not successfully downloaded, the peer chooses to retry its attempt with probability  $1 - p_a$ . The average successful download duration  $\delta(t)$  is then computed considering  $p_c(t)$  and  $p_a$ . If we define the random variable of trials  $X_s(T_0)$  needed for successfully completing the download which started at  $T_0$  after the  $j$ -th download attempt, we obtain the probabilities in Eqn. (3.74).

$$\begin{aligned} P[X_s(T_0) = 1] &= p_c(T_0) \\ P[X_s(T_0) = j] &= (1 - p_a)^{j-1} p_c(T_{j-1}) \prod_{k=0}^{j-2} (1 - p_c(T_k)), \quad j \geq 2. \end{aligned} \quad (3.74)$$

The average time until successfully completing the chunk download which the peer started at time  $T_0$  follows then as shown in Eqn. (3.75). The probabilities for

$X_s(T_0)$  must be normalized by all possible realizations in order to only take the successful download completions into account.

$$\delta(T_0) = \sum_{j=1}^{\infty} (T_j - T_0) \frac{P[X_s(T_0) = j]}{\sum_{k=1}^{\infty} P[X_s(T_0) = k]} . \quad (3.75)$$

### 3.4.3 Solving the Differential Equation System

The proposed model for pollution of a P2P CDN is described as first order ordinary differential equations (ODE). In order to solve ODE

$$\partial_t y(t) = f(t, y(t)), \quad y(t_0) = y_0 , \quad (3.76)$$

we use the Dormand-Prince method known from numerical analysis [51]. The method is a member of the Runge-Kutta family of ODE solvers. More specifically, it uses six function evaluations to calculate fourth- and fifth-order accurate solutions based on Taylor series expansion. The difference between these solutions is then taken to be the error  $\epsilon$  of the (fourth-order) solution. This error estimate is very convenient for adaptive stepsize  $h(\epsilon)$ . The Dormand-Prince method only needs the  $s$ -th order solution at the immediately preceding point to compute the next value

$$y_{n+1} = y_n + \sum_{i=1}^s b_i k_i, \text{ where } k_i = hf \left( t_n + c_i h, y_n + \sum_{j=1}^{i-1} a_{ij} k_j \right) \quad (3.77)$$

with step size  $h$ . The coefficients are given in the so-called Butcher table, see Table 3.2, and we use the Butcher table as in the related Matlab's `ode45` implementation.

Table 3.2: Butcher table containing coefficients for Runge-Kutta solver

$$\begin{array}{c|cccc}
 0 & & & & \\
 c_2 & a_{21} & & & \\
 c_3 & a_{31} & a_{32} & & \\
 \hline
 c_s & a_{s1} & a_{s2} & \cdots & a_{s-1,s} \\
 \hline
 & b_1 & b_2 & \cdots & b_{s-1} & b_s
 \end{array} \tag{3.78}$$

### Numerical Accuracy

For validation of its numerical accuracy, we compare the analytical flow model with simulation results. A user in the P2P system (i) may be malicious offering corrupted content, (ii) may behave selfish or altruistic, or (iii) be impatient and aborts a download. The parameters related to the user behavior are the number of fake peers  $K$ , the sharing probability  $p_s$ , and the abort probability  $p_a$ . We investigate different parameter settings taking such user behavior into account.

Figure 3.15(a) shows the final average population sizes of sharing peers and aborting peers over the number of fake peers  $K$ , when the whole population is in the absorbing states,  $S$ ,  $L$ ,  $A$ . The values are obtained from 20 simulation runs and error bars represent the 99% confidence intervals. The analysis matches the simulations well with only slight differences due to the underlying Markovian assumption at state transitions. The accuracy can be increased by inserting additional intermediate states at the cost of a higher computational complexity for solving the equations. Figure 3.15(a) shows that a small number of  $K = 10$  fake peers is almost sufficient to prevent any peer from completing the download.

Since we consider a non-stationary system, the download duration varies over time according to the current system state. Figure 3.15(b) shows the average duration of a peer as function of the starting time of the download for  $K = 4$ . The analytical result is computed directly from Eqn. (3.75) and compared to values obtained from 20 simulation experiments. In both scenarios with different abort and sharing probabilities, the curves show a good match. The flash crowd arrival

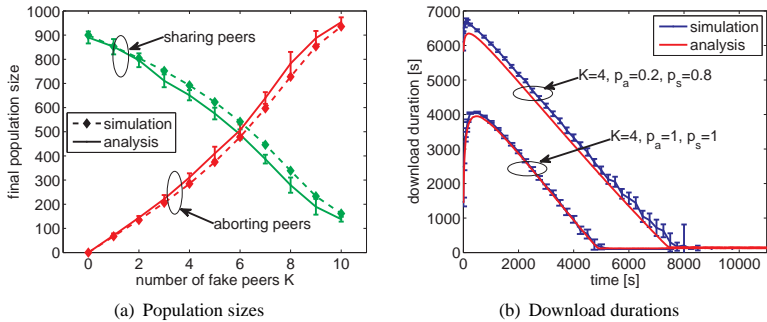


Figure 3.15: Comparison of simulation results with analytic flow model

causes in both cases a strong increase with a linear decrease and in the case of no retries and altruistic users ( $p_a = 1, p_s = 1$ ), the duration is significantly smaller since peers only attempt to download the file once. On the other hand, when  $p_a < 1$  the number of trials has an average greater than one resulting in longer download durations, see Eqn. (3.74).

### Influence of Pollution, Selfishness, and Supporting Servers

The following three major influence factors are considered, that are (a) pollution in terms of number of fake peers  $K$ , (b) user behavior in terms of willingness to share a file  $p_s$ , and (c) the number of supporting servers in terms of initial seeding peers  $S_0$ . We choose an arbitrary (but fixed) observation time instant at which we obtain the number of aborted downloads. In the following examples, we chose the time instant to be one day after the whole process starts.

For illustrating the abortion of downloads clearly, we consider here the case that each user will retry the download exactly once more after having downloaded some corrupted content. If the second download attempt fails, the user will give up. This can be easily realized by enhancing the state space of the pol-



lution model, cf. Figure 3.13, with downloading states  $D_{M+1}, \dots, D_{2M-1}$  and  $F_M, \dots, F_{2M-1}$  for the second download attempt. Then, the following transition rates  $i \rightarrow j$  from state  $i$  to  $j$  have to be modified and accordingly applied in the DES given in Eqns. (3.63)-(3.69).

Transition rates for downloading from regular peers for two download attempts:

$$D_{M-1} \rightarrow A : 0 \quad (3.79)$$

$$D_{M-1} \rightarrow F_M : (1 - p_b)\mu \quad (3.80)$$

$$D_{M+i} \rightarrow D_{M+i+1} : p_b\mu \quad i \in \{0, 1, \dots, M-2\} \quad (3.81)$$

$$D_{M+i} \rightarrow F_{M+i+1} : (1 - p_b)\mu \quad i \in \{0, 1, \dots, M-2\} \quad (3.82)$$

$$D_{2M-1} \rightarrow A : (1 - p_b)\mu \quad (3.83)$$

$$D_{2M-1} \rightarrow S : p_s p_b \mu \quad (3.84)$$

$$D_{2M-1} \rightarrow L : (1 - p_s) p_b \mu . \quad (3.85)$$

Transition rates for downloading from fake peers for two download attempts:

$$F_{M-1} \rightarrow D_0 : 0 \quad (3.86)$$

$$F_{M-1} \rightarrow F_M : \mu \quad (3.87)$$

$$F_M \rightarrow D_{M+1} : p_b \mu \quad (3.88)$$

$$F_M \rightarrow F_{M+1} : (1 - p_b)\mu \quad (3.89)$$

$$F_{M+i} \rightarrow F_{M+i+1} : \mu \quad i \in \{0, 1, \dots, M-2\} \quad (3.90)$$

$$F_{2M-1} \rightarrow A : \mu . \quad (3.91)$$

Thus, a user may only abort after the second attempt, i.e. after downloading  $2M$  blocks, cf. Eqns. (3.83) and (3.91). Download requests for the examples occur with rate  $\lambda$  to investigate the download of  $I(0) = I_0$  peers in total. The DES is enhanced accordingly,

$$\partial_t I(t) = -\lambda(t)I(t) \quad \text{and} \quad \partial_t D_0(t) = \lambda(t)I(t) + \mu(t)D_0(t). \quad (3.92)$$

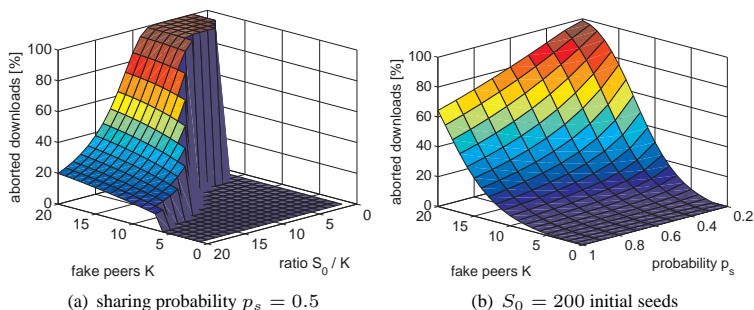


Figure 3.16: Aborted downloads regarding sharing probability  $p_s$ , number of fake peers  $K$ , and number of initial seeds  $S_0$

We can recognize in Figure 3.16 that already a small number of fake peers is sufficient to severely disrupt the diffusion process of a file. Figure 3.16(a) shows the ratio of aborted downloads over the number  $K$  of fake peers and the ratio  $S_0/K$  of initial sharing peers  $S_0$  over fake peers  $K$ . The sharing probability was set constant to  $p_s = 0.5$ . Increasing the number of fake peers leads to an increase in aborted downloads. Especially, if the number of fake sources is greater than 15 and there are less than 100 initial seeds, there will be no successful downloading attempt. It has to be noted that the flat area, where the ratio of aborted downloads is almost zero, stems from non-finished downloads. This occurs if the ratio  $S_0/K$  is too low and falls below a threshold  $\Omega$ . However, the actual threshold depends also on the absolute number of fake peers, i.e.,  $\Omega$  is a non-linear function of  $S_0$  and  $K$ . From Figure 3.16 we can conclude that the actual download time and the number of aborts show a non-linear relationship between  $S_0$  and  $K$ . Considering the scenario where the content providers uses P2P technology to distribute the files, dimensioning of supporting servers, i.e.  $S_0$ , disproportionately highly depends on pollution in terms of fake peers.

In Figure 3.16(b) we investigate the influence of the sharing probability  $p_s$  on

the ratio of aborted downloads. The number of initially sharing peers is chosen as  $S_0 = 200$ . We can recognize that the number of fake peers has more dominating effect on the aborts rather than the sharing probability.

Summarizing, the results show that a relatively small number of fake peers is sufficient to disrupt the propagation of a file in an eDonkey network. Altruism of users cannot help to overcome pollution if too much peers offer corrupted content. However, a large number of initial seeds or supporting servers, being disproportionate to the number of fake peers, attenuates the effect of pollution.

### 3.5 Comparison of Client/Server and P2P

The aim of this section is to evaluate the performance of a content distribution service with respect to reliability and efficiency. We compare a client/server system to a P2P CDN and evaluate the users' QoE in terms of downloading time, success ratio, and fairness while considering flash crowd arrivals and corrupted contents. The number of fake peers  $K$  is assumed to remain constant throughout the observation period. This allows to easily dimension the system for the scenario where the service provider wants to save copyright-protected contents.

In order to compare the performance of P2P and a C/S system, we need to match the conditions like the available capacity of the system and aborted downloads. Therefore, impatient users in both systems cancel their downloading attempt if the total sojourn time in the system exceeds an impatience time  $\Theta$ . In order to make a proper comparison, we now use a deterministic patience

Table 3.3: Default parameters for evaluation of P2P and C/S system

general parameters			P2P parameters		
file size	$f_s$	9.5 MB	initial sharing peers	$S_0$	100
upload bandwidth	$R_u$	128 kbps	seeder arrival rate	$\nu$	0
download bandwidth	$R_d$	768 kbps	departure rate	$\eta$	0
flash crowd decay	$\alpha$	$10^{-3}$	sharing probability	$p_s$	0.8

$\Theta = 50, 100, 150, 200$  minutes. In the P2P system, there are  $S_0$  initial sources for the file which have an upload capacity  $R_u$ . The C/S system is assumed to have a total constant capacity  $C = S_0 R_u$  which corresponds to the total bandwidth available in the P2P system at time  $t = 0$ . Unless stated otherwise, we will make the following assumptions as summarized in Table 3.3. We use now simulations to obtain the numerical results. Each simulation run is repeated twenty times.

### 3.5.1 Success Ratio

The performance of P2P and C/S is now compared regarding the success ratio, i.e., the ratio of successful downloads to the sum of successful and aborted downloads. The success ratio in P2P is 100% for  $\Theta > 50$  minutes and small  $K$ , see Figure 3.17(a). However, when  $K$  increases from 6 to 7, the success ratio with  $\Theta = 200$  minutes reduces to about 50% and for even larger  $K$  no peer completes the download. Figure 3.17(b) shows the equivalent results for CS as function of the number of service units  $n$ . Except when  $n^*$  is too small, the success ratio lies above that of P2P for each  $\Theta$ , especially when the optimal value  $n^* = \lfloor C/R_d \rfloor$  is chosen. We conclude that C/S has at least the success ratio of P2P, if the client

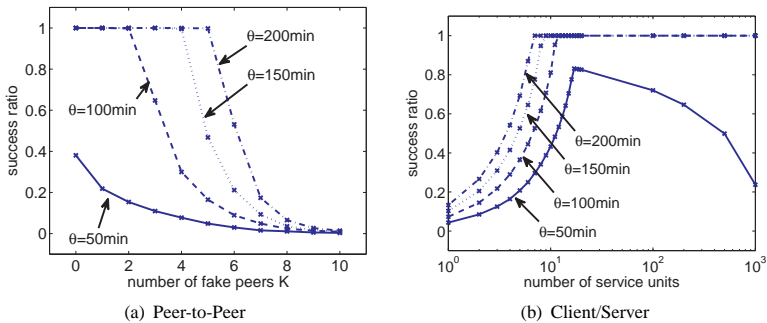


Figure 3.17: Comparison of success ratio between P2P and CS

bandwidths are known a priori for dimensioning the optimal number of service units. The P2P system strongly suffers from the presence of too many fake peers.

### 3.5.2 Download Duration

The key performance indicator from the user's viewpoint is the overall download duration, i.e., the interval from the request of a file until its successful download. In Figure 3.18(a), the time for successful downloads and the sojourn time of aborted downloads is depicted. Since the patience time is deterministic, the abort time is given as straight lines for each  $\Theta$ . The lines begin at values of  $K$  where the success ratios become less than 1. The successful download duration increases with  $K$  until impatience manifests itself in increased canceled downloads. Peers beginning their download later benefit from this effect. As a result the mean download time stays constant or even decreases again with  $K$  and the 99%-confidence intervals from the simulation runs increase due to the decreasing number of successful downloads which can be used to compute the averages.

The results in Figure 3.18(b) show that well dimensioned systems show the best download performance. However, if the optimal capacity is a priori un-

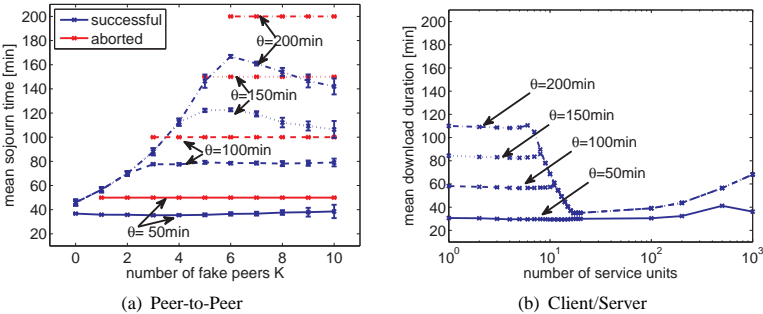


Figure 3.18: Durations of successful downloads

known, the P2P system outperforms the server as the capacity of P2P increases with the number of sharers. If the peers behave altruistic, the P2P system has its advantages and might cope with even more extreme flash crowds, which will crash a server with fixed capacity. The P2P system mainly benefits from incentives and its multiple source technique when sharing already received chunks to other peers, thus fostering the cooperation among peers [30].

### 3.5.3 Fairness Issues

We choose the fairness indicator  $J = (1 + c_\delta^2)^{-1}$  given in [53] which returns values between 0 and 1. Low values of the fairness index indicate an unfair system, while a fairness index of one describes a completely fair system, where all users experience exactly the same download time. The term  $c_\delta$  is the coefficient of variance of the download time  $\delta$  a user experiences. Independent of the number of fake peers  $K$  or the patience time  $\Theta$ , the P2P system is a more fair system with higher fairness index above 0.9, cf. Figure 3.19(a). On the other hand, CS reaches such fairness only for very large  $n^*$  in Figure 3.19(b). In that case, the average download time, however, is larger than in the P2P system (for a small number of

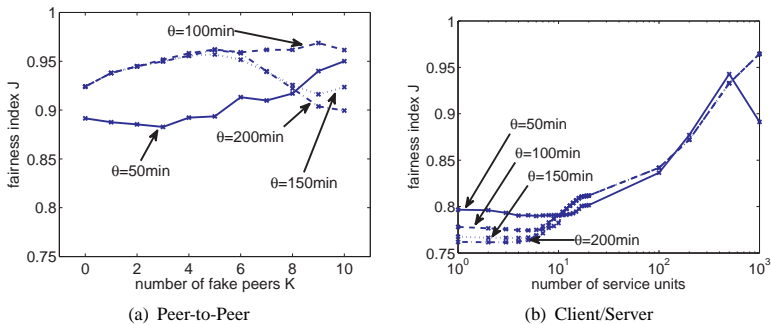


Figure 3.19: Fairness index of successful downloads w.r.t. download duration

fake peers). We can conclude that a well dimensioned CS with a priori knowledge of the clients' bandwidths outperforms P2P at the cost of fairness. Furthermore, we could see that the influence from only few fake peers is sufficient to severely cut down the performance of the P2P system.

### **3.6 Lessons Learned**

Network-based video recorder services like OTR offer their video files via content distribution networks. From the user's point of view, the perceived quality of the services mainly depends on efficiency and reliability. These characteristics can be quantified in terms of download time and success ratio of user requests. Due to the distribution of the typical large video contents, an inefficient service operation may lead to download aborts, when the user gets impatient because of too long download or waiting times. The download requests often occur as flash crowds according to the popularity of videos. Typically, a user gets interested shortly after the video content is released. In the case of OTR, this means that within the first hours or days after recording, most users will request the recorded show. Technologies like P2P help to overcome phenomena like flash crowds and improve scalability compared to server cluster which may get overloaded in such situations. Nevertheless, the P2P technology invokes additional challenges and user behavior than in traditional client/server systems. Beside the willingness to share files and churn of users, peers may be malicious and offer fake contents to disturb the data dissemination. As a consequence, the reliability may be diminished because of pollution of the P2P system and the inherently downloading of useless contents. In order for the content provider to ensure that the file is not being modified by other malicious sources, a client-server solution offers naturally greater security by having a single trusted source offering the information. However, if the served content has a high popularity, there will be a high request rate leading to a drastic increase in server load. All of this leads to a trade-off consideration between high reliability at the risk of overloaded servers and good scalability where the received data may be corrupt.

In this chapter, we aimed at comparing a client/server system to an eDonkey-based P2P system with respect to this trade-off due to the emerging user behavior. Therefore, we provided appropriate models to describe impatience and flash crowd effects. These models allow evaluating the impact of user behavior and dimensioning of system parameters. To get realistic values for the volume of video contents, a measurement study has been conducted for OTR video contents, as well as for shorter YouTube videos. For the client/server system, we proposed a fluid model to describe the time-dynamic evaluation of the system. By means of a Markov model, we derived the sojourn time of an arbitrary user, who successfully completes downloading a file from an OTR server. We investigated the effects of the impatience thresholds of waiting and downloading users, as well as the number of available downloading slots. Understanding these key influence factors allow to dimension the required resources in the system.

Next, we presented an analytical model for the file diffusion process in a P2P file sharing network similar to eDonkey. The model is based on an epidemic model with different populations reflecting the current download state of peers. The numerical results showed that a small number of fake peers can greatly inhibit the propagation of a file. This fact can be used for content providers to protect their copyrighted material from being illegally distributed in the network by introducing a sufficient number of fake peers. A higher willingness of the user to share the file after successfully downloading it can reduce the number of aborted downloads, if the initial sharing ratio among good and corrupt files is sufficiently large. As application, the P2P model allows investigating the impact of fake peers and user behavior in terms of impatience or willingness to share a file. This can be used either (a) to quantify the disturbance of the P2P system due to malicious peers when the service provider relies on P2P technology or (b) to dimension the number of fake peers to save copyright-protected contents for being distributed in illegal file sharing system.

The lessons learned in this chapter on modeling of Online TV Recording services comprise mainly the comparison of P2P and C/S for CDN with respect to reliability and efficiency while taking into account the emerging behavior of users



in both systems. While in general it is not easy to compare both types of networks due to their inherently different structures, we could qualitatively investigate both architectures under comparable situations. Basically, when it comes to the reliability, servers seems to be the better choice, as manipulated data is not being injected into the network. However, malicious users can also attack the C/S system by generating denial of service attacks which may look like massive flash crowd arrivals. From the view of the end user, the same effect may be experienced when downloading from a trusted server as with P2P networks with pollution or poisoning. Especially, when the request arrival rate is high, the waiting time until the download can be processed or its duration may become too long. The problems in C/S performance can be overcome by adding further server capacity. Now, P2P systems can be easily made inoperable when many fake sources exist. If the initial number of sources is small there is a risk of these peers leaving the system which would make the network lose content due to churn. For this reason, it is important that incentives are being provided to peers to increase the willingness to share the data. For a service provider relying on P2P, a hybrid solution may also overcome pollution by assisting the CDN with servers or caching network elements.



## 4 QoE of Edge-Based VoIP Applications

User satisfaction with application and service performance in communication networks has attracted increased attention during the recent years. The interest in how the user perceives usability, reliability, quality and price-worthiness as a means of competition is increasing. The network and service providers need to be able to observe and react upon quality problems, at best before the customer takes notice of them. The notion of QoE was introduced in several white papers [118, 158], mostly in the context of multimedia delivery such as IPTV. Besides of objective end-to-end QoS parameters, QoE focuses on subjective valuations of service delivery by the end users. The necessity of introducing QoE can be explained on the example of VoIP. A voice user is not interested in knowing performance measures like packet loss or received throughput, but mainly in the experienced speech quality and timeliness of the connection setup.

In the previous chapters, however, we have demonstrated that future Internet applications may lead to (a) new challenges, e.g. an inefficient usage of resources for P2P file sharing due to heterogeneity in B3G, and (b) newly emerging user behavior, like selfishness or pollution. In both cases, the user perceived quality is decreased as a result. The design of future Internet applications has to account for this QoE concept. Therefore, they may follow a new paradigm, in which the intelligence of the network control is gradually moved to the edge of the network. This *edge-based intelligence* is reasonable from the view point that the application knows best its service requirements. For example, a voice application knows its used voice codec and thus the corresponding required minimum throughput.

**Skype VoIP application** Currently, there exist applications and services in the Internet which implement the control of network traffic on application layer. Popular examples for such edge-based services are P2P file-sharing networks, like eDonkey or BitTorrent, or the Skype VoIP client. Both services have in common that the application itself determines the amount of consumed bandwidth. This impacts both the objective QoS of the end-to-end connection as well as the subjective QoE as perceived by the end-user.

Skype is a proprietary application which is based on P2P technology. It offers rapid access to a large base of users, seamless service operation across different types of networks (wireline and wireless) with an acceptable voice quality [19], as well as a distributed and cost-efficient operation of a new service. The good voice quality of the Skype service is achieved by appropriate voice codecs, such as iSAC and iLBC [123], as well as by adapting the traffic rate of the sender to the current conditions in the network which are described by classic end-to-end QoS parameters, like packet loss or jitter. However, the end-to-end QoE perceived by the user will be the essential criterion for the subscriber of a service. A typical QoE measure is the *Mean Opinion Score* (MOS) [79], which can be determined from *subjective* ratings by real users or predicted from *objective* measurements of properties of the delivered audio. In order to choose appropriate (counter-)measures to keep user-perceived service quality above a certain threshold, a provider needs to know how network-level QoS parameters translate into user-level QoE perception and vice versa.

To stress its edge-based intelligence, we examine if mobile Skype is feasible in current 3G networks with varying network conditions. UMTS operators promise to offer large data rates which should suffice to support VoIP calls in a mobile environment. To investigate this, the actually achieved quality of IP-based voice calls using Skype can be measured in a public UMTS network. In addition, we can emulate the UMTS environment to push the network to its limits in order to investigate how Skype reacts to certain conditions in the network. As a side-effect, we obtain a traffic profile for common QoS parameters of the proprietary Skype application. Related work on Skype is briefly reviewed in Section 4.1.4.

---

**Goal and structure of chapter** The main focus of this chapter is on how the current network conditions described as QoS parameters influence the QoE of a VoIP user and in how far an edge-based application like Skype reacts to quality degradations. As fundamental background, we elaborate on how to assess quality in Section 4.1. In particular, different quality metrics relevant for QoE and QoS evaluations are discussed. Since basic QoS problems on network level result in QoE degradations, a qualitative relationship between QoE and QoS exists. The identification of such a generic relationship between QoE and QoS is formulated and derived as IQX hypothesis. It presents an exponential dependency of QoE from QoS. With respect to the IQX hypothesis, we review related work dealing with user experience in web browsing and demonstrate that the exponential interdependency is also valid here.

To quantify the influence of QoS problems on the QoE for VoIP applications, we set up a testbed to measure the quality of VoIP traffic. In the testbed, we are able to control the network conditions and to inject for instance loss or jitter. Packet traces are captured to measure the QoS parameters. The received audio signals are compared to the originally sent audio signals to assess the QoE. The computation of QoS and QoE parameters, as well as the verification of the correct emulation of network conditions are explained in Section 4.2.

After that, the QoE of the voice codecs iLBC as used by Skype and G.711 is related to certain QoS impairment factors in Section 4.3. In particular, we quantify the impact of uncorrelated and correlated delay and jitter, packet reordering, random packet losses, and bursty losses. We test the IQX hypothesis and show that we can confirm the hypothesis when appropriate metrics are selected for describing the QoS impact on application layer.

In Section 4.4, we investigate Skype's edge-based intelligence in a 3G environment. This is done by performing measurements in both a public UMTS network and a testbed environment. The latter is used (a) to introduce network disturbances like packet loss or jitter, as well as (b) to emulate rate control mechanisms and changing system conditions of UMTS networks. Based on the obtained QoS and QoE measurements, we answer the following questions in Section 4.4.

Does Skype work properly with a rate-controlled dedicated channels in UMTS? In how far does the emulation tool influence the behavior of Skype? Which impact does the UMTS network itself have on the voice quality? During a connection, does Skype react to network changes? The gained experiences and results brings us to Section 4.5 where we give an outlook and future work in the area of QoE management and provisioning. Finally, the lessons learned in this chapter are summarized in Section 4.6.

## 4.1 Background: Assessment of Quality

Quality of Experience combines user perception, experience and expectations with non-technical and technical parameters such as application- and network-level QoS. While the ITU standards focus on service quality towards the end user [55], the IETF's understanding of QoS relates to the capabilities of the network to provide packet transfer in a better-than-best-effort way. While the ITU view on QoS is user-centric, the IETF view on QoS is network-centric. This raises the question of how network-level QoS measurements and control relate to the user perception of a service.

There is however still a lack of quantitative descriptions or exact definitions of QoE. One particular difficulty consists in matching subjective quality perception to objective, measurable QoS parameters. Subjective quality is amongst others expressed through MOS [79]. Links between MOS and QoS parameters exist predominately for packetized voice such as VoIP. Numerous studies have performed measurements to quantify the effects of individual impairments on the speech quality on a single MOS value for different codecs, for example G.729 [56], GSM-FR [132], or a comparison of some codecs [83]. Additionally, the E-model [59] and related extensions [73] assess the combined effects of different influence factors on the voice quality. In [58], the logarithmic function is selected as generic function for mapping the QoE, there denoted as user level QoS, from a single parameter because of the mathematical characteristics of the logarithmic function.

We follow a different approach in this chapter and motivate a fundamental relationship between the QoE and quality impairment factors such as packet loss and related jitter on the example of VoIP. Basic QoS problems on network level relate to (a) late delivery, (b) non-delivery, (c) out-of-order delivery, and (d) changed contents of IP packets if not captured by the link level. They affect the timely behaviour of the application and the appearance of the content, respectively. Obviously, generic QoS problems (such as loss, delay, jitter, re-ordering, throughput limitations) imply *generic QoE problems* (such as glitches, artifacts, excessive waiting times). For estimating the QoE, different approaches are explained and classified in Section 4.1.1.

Due to these generic quality problems, we research and propose a generic relationships between QoE and QoS in Section 4.1.2. The identification of such a generic relationship is formulated as IQX hypothesis and derived in Section 4.1.3. It presents a unified and practicable formula expressing an exponential interdependency of QoE from QoS. The formula's three parameters allow for simple matching and comparison of statistics and limits, respectively. It is thus applicable for online, in-service classification of QoE problems based on QoS observations, which is of interest for service providers and network operators. With respect to the IQX hypothesis, we review related work dealing with web browsing in Section 4.1.4 and demonstrate that the exponential interdependency is also valid here.

### 4.1.1 Quality Comparisons and Classification of Metrics

The derivation of QoE–QoS relationships builds upon quality comparison between (1) the so-called *reference*, by which we mean undistorted content such as voice or video, or an undistorted service such as a download activity, and (2) the outcome of the transmission in form of a potentially distorted voice or video, or a delayed download activity which is referred to as *outcome* in the following. Such a distortion may impact the quality of the content (e.g. speech quality) and/or of

the timing (e.g. fluidity of a video or download times). Then, the QoE can be seen as the remaining quality of the outcome after such a distortion. In the context of VoIP, the encoding of voice data and the delivery through the IP network may introduce distortion and degrades the QoE.

**Quality comparison** For quality comparison, there are different measurement methods and observation levels which we introduce briefly to clearly show the applied methodology in this chapter. We can distinguish between *communication situation* and *lab situation*, cf. [149]. In a communication situation, the reference is in general not available, only the outcome can be observed and analyzed. In a lab situation, both reference and outcome are available and can be compared with great effort and in great detail, which often imposes the necessity of carrying out this analysis off-line. The measurements presented in this chapter are conducted in a lab situation.

There exist two basic measurement options which are *subjective testing* and *objective testing*. Usually, subjective quality tests form the basis for perceptual objective test methods. *Subjective tests* are carried out by a test panel of (real) users. While many (possibly even diverging) views on the quality of the outcome can be taken into account leading to accurate results as well as a good understanding of the QoE and its sensitivity, this type of test can be both time-consuming and costly, since the tests have to be conducted by a large number of users for statistically relevant results. *Objective tests* are carried out by an algorithm on behalf of a real user, trying to imitate (or predict) user perception based on key properties of the reference and/or the product. Objective tests can follow psychophysical approaches and engineering approaches, a detailed description of which is found in [149]. For VoIP, the PESQ (Perceptual Evaluation of Speech Quality) standard [64] objectively evaluates and quantifies voice quality of voice-band (300 - 3400 Hz) speech codecs. It uses psycho-acoustic and cognitive model to analyze and compare the reference and the outcome. As PESQ allows for repeatable and automated measurement processes, we rely on this algorithm for quantifying QoE-QoS relationships and obtaining statistical significant results.



Depending on the object of interest, we can observe contents and related network traffic on different levels. Observation on *application level* implies examination of the payload, which makes it possible to get a detailed picture of the content and on the timing of reference and outcome. Problems with the latter may arise from the network, including its links, and the network stacks in the end systems, as well as the implementation of the application itself like pre-buffering. Additionally, measurements on *network level* may be conducted. This means investigation of the flow of packets in terms of completeness, timeliness, and pattern types like bursty losses or correlated delays. In our measurements, we observe both levels to derive QoE – QoS relationships.

**Classification of metrics** Depending on the available information for subjective or objective tests, quality metrics can be classified according to the following three categories, cf. amongst others [81, 139, 149]:

*Full Reference* (FR) metrics: Both outcome and reference are available and allow for detailed subjective and objective comparisons of voice, images, videos, download times on application level, as well as packet traces on network level, etc. Concretely, this means extraction, evaluation and comparison of QoE- and QoS-related parameters on any level in an off-line manner, which is most interesting for deriving QoE–QoS relationships. FR metrics deliver the highest accuracy, but require high computational effort.

*No Reference* (NR) metrics: Quality information has to be extracted from the outcome, as no reference is available. This is a typical online situation with sole focus on the resulting quality as perceived by the end user, e.g. evaluated through questions, or the user’s representative, e.g. an algorithm. In a networking context, NR metrics are usually lacking the possibility of discerning between quality problems stemming from the reference, e.g. quality degradations due to encoding, and additional disturbances by the network. Thus, NR metrics are not applicable for deriving QoE–QoS relationships aiming at capturing the impact of the network. NR metrics estimate the actual QoE with a low accuracy only. Common variants of NR algorithms even analyze only on network level.

*Reduced Reference (RR) metrics:* Instead of comparing directly the reference with the outcome, parameters on application and/or network level are extracted at the sending and receiving side which help predicting the QoE. As an example, on application level the RR Hybrid Image Quality Metric (HIQM) [81] computes various criterions of the reference image and sends them to the receiver. The extracted parameters are taken into account for estimating the quality of the received image without needing the reference image at the receiver. As a further example, on network level throughput variations and losses may be derived and compared to estimate the quality on receiver side as done in [75, 112]. Such parameters often have their roots in FR research as a means of summarizing and interpreting the outcomes. However, as they represent key QoE and QoS parameters in a very condensed manner, they can be applied in an online in-service scenario by transmitting them between source and sink, and subsequently comparing them in order to find out about quality problems. Because of their background, they represent promising candidates to build QoE–QoS relationships upon [19, 112, 127].

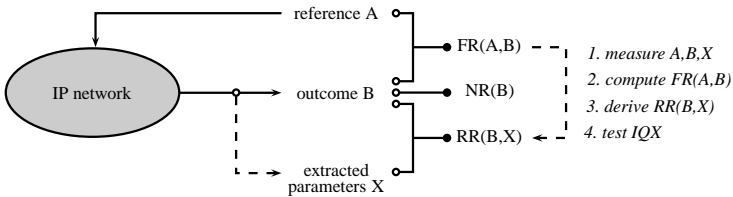


Figure 4.1: Illustration of the different quality metrics and approach applied for testing the IQX hypothesis

Figure 4.1 illustrates the different FR, NR, and RR quality metrics and their required inputs. For the FR metric, the reference  $A$  as well as the outcome  $B$  are available allowing to estimate the QoE by  $FR(A, B)$ . For the NR metric, only the outcome  $B$  is available,  $NR(B)$ . For the RR metric, in addition to the outcome  $B$  the extracted parameters  $X$  are available which (a) may be measured at the receiver side on network level and/or (b) may be extracted at the sender side

on network and/or application level. Thus, the quality is estimated as  $RR(B, X)$ .

Figure 4.1 also shows the approach we follow to quantify QoE–QoS relationships. Since we perform the measurements in a lab situation, we are able to measure the reference  $A$ , the outcome  $B$ , as well as extracted QoS parameters  $X$  on network level. We rely on FR metrics to get a high accuracy and obtain  $FR(A, B)$ . We then investigate and derive an appropriate RR metric  $RR(B, X)$  according to the proposed IQX hypothesis. Finally, the IQX hypothesis is tested for the considered scenario.

### 4.1.2 Qualitative Relationship Between QoE and QoS

We now turn our focus onto a qualitative, schematic relationship describing the impact of QoS problems onto QoE, illustrated by Figure 4.2. On the x-axis, the QoS disturbance is denoted, while the y-axis indicates a QoE value, e.g. in terms of MOS. Although this relationship is basically independent of the type of the metric discussed above, we now focus on a situation in which the network accounts for QoE reductions between reference and outcome.

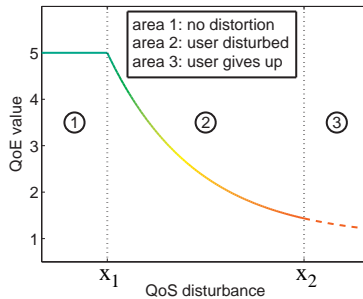


Figure 4.2: General shape of the mapping curve between QoS and QoE

The QoE of the outcome of the transmission as a function of QoS disturbance is split in several regions, (1) no distortion, (2) user is disturbed, (3) user gives

up. The actual thresholds of the different areas are referred to as  $x_1$  and  $x_2$ . Thus, threshold  $x_1$  indicates when the user gets disturbed by the QoS disturbance and experiences a lower quality. The threshold  $x_2$  indicates when the QoS disturbance is such high that the user is dissatisfied and gives up. In Chapter 3, we have considered this case for online TV recording services and QoS disturbance due to pollution.

*Area 1: Constant optimal QoE.* For a vanishing QoS disturbance, e.g. in case of a transparent network, the QoE is that of the reference, determined by hardware and software configuration as well as the chosen network technology. A slight growth of the QoS disturbance may not affect the QoE at all. For instance, small delay and delay variations may be eliminated by a jitter buffer, without the user noticing the additional delay. Typically, the user considers the quality to be good, which is illustrated by the green color in Figure 4.2. Another reason is that a user is not able to determine a better QoS. An example is given in Section 4.1.4 for the delivery of web pages. Even if web pages are delivered faster such that the web page delivery time falls below the threshold  $x_1$ , the user does not perceive a better experience.

*Area 2: Sinking QoE.* When the QoS disturbance exceeds a certain threshold  $x_1$  (e.g. when the current delay exceeds the capacity of the jitter buffer, yielding buffer underflow), the former QoE level cannot be maintained any more. As the QoS disturbance grows, the QoE and thus the user satisfaction sinks, which is illustrated by the green color switching to yellow and finally to red. In case of a high QoE, a certain additional QoS disturbance might have a considerable impact on the QoE, while for low QoE, that particular additional QoS disturbance might not be that critical any more. Consequently, as the QoE sinks, its gradient is expected to do so as well.

*Area 3: Unacceptable QoE.* As soon as the QoS disturbance reaches another threshold  $x_2$ , the outcome of the transmission might become unacceptably bad in quality, or the service might stop working because of technical constraints such as timeouts. If a user was involved, it might give up using the service at that point. This is illustrated by the dashed line.

While threshold  $x_1$  due to its technical nature represents a sharp threshold that very well may be co-located with the  $y$ -axis, threshold  $x_2$  may be user-dependent, an example of which will be discussed for the cancellation rate of web browsing users in Section 4.1.4.

### 4.1.3 The Exponential Interdependency of QoE and QoS Hypothesis

We demonstrate now a fundamental functional relationship between the QoE and QoS parameters, like packet loss or jitter. As an analytical solution of this relationship between QoE and QoS, we formulate the IQX (Interdependency between QoE and QoS is eXponential) hypothesis. At large, the QoE is a function of  $n$  influence factors  $I_j, 1 \leq j \leq n$ :

$$QoE = \Phi(I_1, I_2, \dots, I_n). \quad (4.1)$$

However, in this chapter we focus on single influence factors indicating the QoS in order to motivate the fundamental relationship between the QoE and an impairment factor corresponding to the QoS. The idea is to derive the function  $QoE = f(QoS)$  with a single impairment factor  $I = QoS$ .

In general, the subjective sensibility of the QoE is the more sensitive, the higher this experienced quality is. If the QoE is very high, a small disruption will decrease strongly the QoE, also stated in [58]. On the other hand, if the QoE is already low, a further disturbance is not perceived significantly. This relationship can be motivated when we compare with a restaurant quality of experience. If we dined in a five-star restaurant, a single spot on the clean white table cloth strongly disturbs the atmosphere. The same incident appears much less severe in a beer tavern.

On this background, we assume that the change of QoE depends on the current level of QoE – the expectation level – given the same amount of change of the QoS value. Mathematically, this relationship can be expressed in the following

way. The performance degradation of the QoE with respect to a certain QoS parameter, like packet loss, is  $\frac{\partial QoE}{\partial QoS}$ . Assuming a linear dependence on the QoE level, we arrive at the following differential equation:

$$\frac{\partial QoE}{\partial QoS} = -\tilde{\beta} \cdot (QoE - \gamma) . \quad (4.2)$$

The solution for this equation is easily found as an exponential function, which expresses the basic relation of the IQX hypothesis:

$$QoE = \alpha \cdot e^{-\beta \cdot QoS} + \gamma . \quad (4.3)$$

Note that in this context the IQX hypothesis is formulated with  $QoS$  as parameter for the current quality of service. The higher the value  $QoS$  is the lower the objective quality is. The higher the value  $QoE$  is the higher the subjective quality is. The limit  $QoS \rightarrow \infty$  goes to  $\gamma$  in this case. In Eqn. (4.3),  $QoS$  is for example the packet loss ratio and  $QoE$  is described in terms of MOS. In any other cases, the algebraic signs have to be adapted adequately in Eqn. (4.3).

### 4.1.4 Related Work

While we test the IQX hypothesis for the G.711 and Skype's iLBC voice codec in Section 4.3, web browsing is considered as a second example for testing the IQX hypothesis in the following two subsections. In that case, we use well-known results and measurements from literature. The first one uses passive measurements of HTTP traffic to find a relationship between QoE and QoS. In the second approach, users are interviewed about their perceived quality and average mean opinion scores are calculated. In both cases, the authors propose a logarithmic interdependency, but we will show that in both cases the IQX hypothesis can be applied convincingly. Finally, related work on Skype is briefly reviewed, since we use the Skype application and in particular Skype's iLBC codec for the measurements presented in this chapter.

### Cancellation Rate of Web Browsing Users

The presented measurements here are taken from [69] in which Khirman and Henriksen measure the level of user dissatisfaction with the web-content delivery quality. We use these results to check the IQX hypothesis in the context of web browsing. As QoS parameter, the delivery bandwidth is used. The QoE is expressed as cancellation rate of web requests.

Khirman and Henriksen use a passive network-attached sniffing device that collects packets traveling across a specific network link. Afterwards, they apply reverse engineering to the captured packets to get information about the states of TCP connections and to extract details of the application layer transactions. The data collector was installed in a commercial ISP network with public Internet access. 80 % of the traffic was generated by customers using dial-up modem connections up to 56 kbps; 20 % of the traffic was generated by customers using high-speed connections. More details of the measurements can be read in [69].

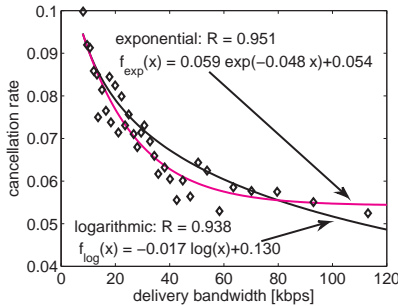


Figure 4.3: Measurement results for web browsing taken from Khirman and Henriksen [69] and comparison of logarithmic model  $f_{log}(x)$  and exponential model  $f_{exp}(x)$

Figure 4.3 shows the cancellation rate of HTTP objects depending on the delivery bandwidth of that object. It has to be noted that only objects for which

at least 8 kB was transferred are considered. This results into 373,050 object requests of which 22,903, i.e. 6.1 %, were cancelled. Note that in this figure only low range delivery bandwidth up to 120 kbps is considered due to the fact that the majority of users have dial-up connections.

Every point in this graph represents the cancellation rate for a bin of 7461 objects with a similar delivery bandwidth. The x-value a point denotes the average delivery bandwidth of these objects; the y-value represents the cancellation rate which is the ratio of canceled objects divided by the total number of requests in this bin. In order to determine if an object is canceled, the object size advertised by the server and the actual size of the delivered object are compared.

Khirman and Henriksen propose a logarithmic fitting function for the cancellation rate  $f_{log}(x) = -0.017 \log x + 0.130$  in dependence of the delivery bandwidth  $x$  in [69]. The resulting coefficient of correlation is  $R = 0.938$ . Applying the IQX hypothesis to these measurements leads to a slightly better coefficient of correlation  $R = 0.951$ . The exponential model is described by  $f_{exp}(x) = 0.059e^{-0.048x} + 0.054$ .

In Figure 4.3, we see that the exponential function decreases stronger in the beginning. At the end, however, the shape of the exponential curve is more flat than for the logarithmic function in this area and approaches slowly the asymptote  $\lim_{x \rightarrow \infty} f_{exp}(x) = \gamma = 0.054$ . Note that the logarithmic function – in contrast to the exponential – is not bounded and goes towards minus infinity,  $\lim_{x \rightarrow \infty} f_{log}(x) = -\infty$ . Hence, the IQX hypothesis might be more appropriate for modeling the cancellation rate as QoE parameter with respect to the delivery bandwidth as QoS parameter.

### **Mapping of Weighted Session Time to Perceived Web Browse Quality**

The next example for checking the IQX hypothesis is based on the ITU-T recommendation G.1030, *Estimating end-to-end performance in IP networks for data applications* [116]. It applies perceptual models to gauge user satisfaction, i.e.



QoE, with the end-to-end performance, i.e. QoS. As an example web browsing applications are depicted. As QoS parameter response and download times are used which are measured in the network or calculated from the HTTP transaction times. Regarding the QoE, experiments are conducted where the response and download times in a web session were manipulated and the users are asked to evaluate the perceived quality according to the five-point MOS scale (5: excellent; 4: good; 3: fair; 2: poor; 1: bad).

In [116], it is stated that the expected maximal session time will dominate the perceived quality and the user's rating. Therefore, the network context, i.e. fast, medium, and slow network, is taken into account and for each of these network types individual time scales are used for the maximal session time  $t_{max}$ . The considered values are 6 s, 15 s, and 60 s, respectively.

Basically, the web session consists of three steps, (a) a subject first requests and retrieves a search page which is then displayed, (b) the subject types and submits a search time on this page, and (c) then retrieves a page showing the search results. The users were asked to type in the same search query in every session. In total, 49 experiments were conducted for each of the three network contexts where the response times ( $t_1, t_3$ ) and download times ( $t_2, t_4$ ) are varied, see Figure 4.4. Here, the testing users are distinguished into two separate groups, trained experts and untrained (naïve) users.

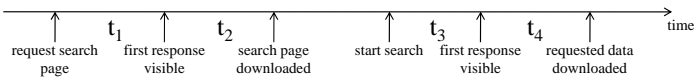


Figure 4.4: *Different time components of the web session for computing the weights of the weighted session time in G.1030. Figure is taken from [116]*

As a result of [116], it was found out that for the fast network with  $t_{max} = 6$  s and naïve users the coefficient of correlation between session time and MOS is too low,  $R = 0.72$ . Therefore, the model was extended and the weighted session

time was used as QoS parameter. The idea is to find weights  $w_i$  for the different time components  $t_i$  of the entire web session which maximize the correlation between this weighted session time  $t_w$  and the QoE. The different time components are illustrated in Figure 4.4. It is

$$t_w = w_1 t_1 + w_2 t_2 + w_3 t_3 + w_4 t_4 . \quad (4.4)$$

The sum of the weighting coefficients is normalized to 4.0 in order to be able to compare normal session times,  $t = t_1 + t_2 + t_3 + t_4$ , with weighted ones  $t_w$ .

Figure 4.5 shows the measurement results from [116] for the fast network and the entire user set, i.e. experts and naïve users. Each point in the graph represents a single experiment with the weighted session time on the x-axis and the mean opinion score on the y-axis. In [116], a logarithmic mapping function  $f_{log}(x)$  is proposed which yields to a coefficient of correlation  $R = 0.954$ . Note that the logarithmic model leads to MOS values above 5 for  $t_w < 0.62$  s and to MOS values below 1 for  $t_w > 13.48$  s. This is indicated in Figure 4.5 by the dotted line style.

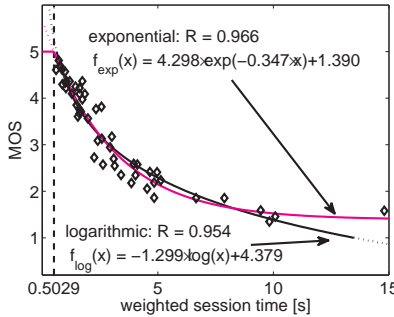


Figure 4.5: Measurement results for web browsing in a fast network taken from G.1030 [116] and comparison of logarithmic model  $f_{log}(x)$  and exponential model  $f_{exp}(x)$

Testing the IQX hypothesis results into a coefficient of correlation  $R = 0.966$  which is slightly better. An advantage of using the exponential curve as mapping between QoS and QoE is the fact that  $f_{exp}(x)$  is bounded for large weighted session times in contrast to  $f_{log}(x)$ . In particular, it is  $\gamma = 1.390$ . However, for very small session times  $t_w < 0.50$  s the exponential function also leads to MOS values above 5. This example nicely demonstrates the threshold  $x_1$  in Figure 4.2 of the principal shape of the mapping function between QoS and QoE, see Section 4.1.2. A user is completely satisfied if the session time is around half of a second. If the data is delivered even faster than that, the user is not able to perceive this better quality of service. Regarding network planning, it might be possible to save resources, as it is not necessary to provide better QoS for maintaining the same QoE. This already shows the potential impact of QoE and paradigm change in telecommunication networks accompanied by the consideration of QoE instead of QoS.

### Skype VoIP Application

The immense success and popularity of Skype made it subject to different research studies illuminating various interesting aspects. First of all, Baset et al. [123] analyzed initial versions of Skype and revealed its different mechanisms to traverse NAT routers and firewalls. They showed that Skype is based on P2P technology and relies on the concept of Superpeers which are, e.g., used to relay calls between peers which are not able to establish a direct connection. Ehlert et al. [126] derived typical signatures of such relayed Skype VoIP sessions in order to support administrators in detecting Skype traffic in their network. A similar approach was applied in [143] by performing measurements on both the client and the server side of a relayed call in order to characterize and detect relayed traffic. Guha et al. [130] studied the session times and bandwidth consumption of Superpeers in the Skype overlay. Based on measurements on a specific Superpeer they derived the complementary distribution function for relayed VoIP call durations as well as for the size of files transferred over the Skype overlay.

In [91] first concerns were expressed to use Skype in a corporate environment due to security issues. The topic of security was further discussed in [114], [124], and [108]. Skype encrypts its calls using AES with a block size of 128 bit and a key size of 256 bit. Authorization is done using RSA keys of up to 2048 bit. A closer look at the Skype binary also revealed that it tries to protect itself from being reverse engineered by refusing to start when tools like the Soft-Ice debugger are present.

In contrast to all previous work, we intend to study how Skype reacts to changes in the network and how this affects the satisfaction of the user with the service. Therefore we characterize the traffic generated by a Skype client in different environments and relate it to the quality as perceived by the end user. The work which comes closest to our studies is [125], in which the authors derive a User Satisfaction Index (USI) which translates typical network parameters as well as measured call durations into a performance measure for user satisfaction. The two main points in which we differ from this approach are that we regard a mobile UMTS environment and try to uncover how Skype performs in such situations and how it is able to maintain the measured user satisfaction even under the changing network conditions which are typical in mobile networks. In addition, we measure the QoE using the established and generally accepted MOS value, which relies on the comparison of audio files instead of trying to translate network parameters into user satisfaction.

## 4.2 Measurement Testbed and Setup

The general measurement setup applied in the experiments of this chapter is as follows. We installed the VoIP application on two end hosts *A* and *B*. The voice user *A* sends audio data to voice user *B* using UDP and IP on transport and network layer, respectively. The audio data is an English spoken text without noise of length 51 seconds, sampled at a rate of 8 kHz, encoded with 16 bits per sample which is a standard audio file for evaluating VoIP and available at [141]. The audio file was played with the Winamp audio player on machine *A*, whereas

the output of Winamp was used as input for the voice application (instead of a microphone). Packet traces were captured using the latest versions of TCPDump and Windump on each machine according to the underlying OS, respectively.

The main focus of our studies is on how the current network conditions influence the QoE of the end user and in how far an edge-based application like Skype reacts to quality degradations. This is done by performing measurements in both a real UMTS network and a testbed environment. The latter is used (a) to introduce network disturbances like packet loss or jitter, as well as (b) to emulate rate control mechanisms and changing system conditions of UMTS networks. In this context, we investigate to what extent the results depend on the way the network is emulated and if there are differences to measurements in a real UMTS networks. Therefore, we apply two different emulation approaches, one based on hardware and one based on software. For the hardware based approach we used a Cisco 3660 router running IOS 12.0, the software based approach was realized using dummynet (<http://dummynet>) and NIST Net, two freely available tools which offer different abilities to emulate typical network behavior for individual end-to-end connections. Figure 4.6 gives an overview on the conducted measurement scenarios. Details of the concrete measurement setup for the different scenarios are given in the related sections.

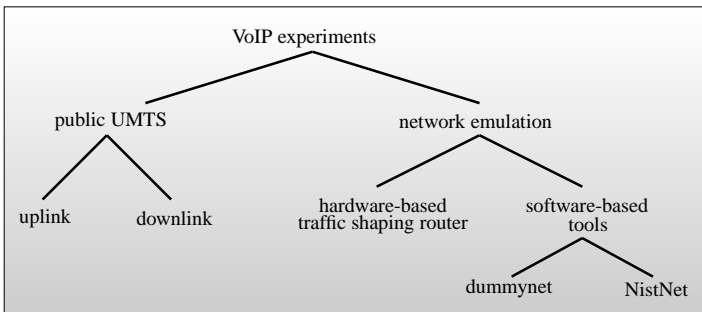


Figure 4.6: Overview on conducted measurement scenarios

During the course of the measurements, NIST Net turned out to fit well for our purposes and was therefore used in the experiments for quantifying QoE in dependence on QoS parameters and testing the IQX hypothesis in Section 4.3. NIST Net is a network emulation package running only on Linux. It allows a single Linux PC set up as a router in order to emulate a wide variety of network conditions. In particular, selected performance effects are applied to the IP packets of the out-going stream. Via command line, the network conditions of a single end-to-end path can be controlled, which is again required for the automated measurement process. The controllable network parameters of interest for our measurement scenarios are packet loss and delay. It is possible to generate random packet losses according to a given packet loss probability  $p_L$ . This means IP packets are randomly dropped with probability  $p_L$ . NIST Net additionally accepts an autocorrelation parameter  $r_L$  for the loss, however, this parameter has no effect on the out-going stream, which is demonstrated in Section 4.2.2. In order to control the delay between two nodes, the average delay  $\mu_d$ , the standard deviation  $\sigma_d$  of the delay, and the autocorrelation  $r_d$  can be passed to NIST Net, which uses a normal distribution with the related parameters to randomly generate delays. To verify our measurement setup, we checked in particular in Section 4.2.2 whether the desired network conditions are correctly emulated by NIST Net.

### 4.2.1 Computation of QoS and QoE Parameters

The end-to-end quality of the communication between two end hosts can be evaluated on different levels and from different points of view. The traditional approach captures the QoS using measurements on the network layer. The derived technical parameters precisely describe the current ability of the network to provide a service but do not necessarily reflect the quality felt by the user of the service. On that account a new paradigm emerged which intends to assess the QoE describing the satisfaction of a user with the service. In the following, we show how we measured and captured QoS and QoE in our testbed environment. The investigated performance measures comprise the QoE in terms of the MOS value

and the QoS in terms of network-based factors like throughput, packet interarrival times, or packet loss.

As results of the measurements we obtain the received audio file and packet traces at the sender machine  $A$  and the receiver machine  $B$ . For each sent and received packet on both machines, we extract a unique ID, the size of the packet, and the local timestamp when the packet is sent or received, respectively. Note that the clocks at  $A$  and  $B$  are not synchronized and might drift. However, time and frequency synchronization are not necessary for assessing the applied QoS parameters.

**Packet Loss** Let  $s$  be a stream of packets stemming from the application under investigation and  $s_{out} = \{p_{out,1}, p_{out,2}, \dots, p_{out,n}\}$  be the set of packets that are sent from  $A$  to  $B$ . The packets  $p_{out,i}$  are ordered in ascending order according to their sending timestamps  $t_{s,p_{out,i}}$ , i.e.  $i < j \Rightarrow t_{s,p_{out,i}} \leq t_{s,p_{out,j}}$ . Analogously, let  $s_{in} = \{p_{in,1}, p_{in,2}, \dots, p_{in,m}\} \subseteq s_{out}$  be the set of packets that are received by  $B$  from  $A$ . The packets  $p_{in,i}$  are ordered in ascending order according to the timestamps  $t_{r,p_{in,i}}$  when the packets are received, i.e. for  $i < j$ , we observe  $t_{r,p_{in,i}} \leq t_{r,p_{in,j}}$ . The measured packet loss ratio simply follows as

$$\tilde{p}_L = 1 - \frac{|s_{in}|}{|s_{out}|} = 1 - \frac{m}{n}. \quad (4.5)$$

On average, the measured loss ratio  $\tilde{p}_L$  should be equal to the preset packet loss probability  $p_L$  of the network emulator, i.e.  $\tilde{p}_L$  converges asymptotically to  $p_L$ .

**One-Way Delays** The one-way delay is basically defined as the time difference between the time  $t_{s,p}$  when sending the first bit of a packet  $p$  at the sender side until the time  $t_{r,p}$  receiving the last bit of the packet  $p$  at the receiver side [61]. The one-way delay  $d_p$  for a packet  $p$  follows as

$$d_p = t_{r,p} - t_{s,p} \text{ for } p \in s_{in} \subseteq s_{out}. \quad (4.6)$$

Note that in case of dropped packets the one-way delay is not defined. However, as the clocks at the sender and the receiver side do not need to be synchronized and the clocks might additionally drift, the estimation of one-way delays out of measurement data is a complex task. Binzenhöfer et. al propose in [145] a method to estimate accurate one-way delays based on packet captures at the sender and at the receiver side. The method assumes symmetric one way delays in the uplink and the downlink and readjusts the measured values in such a way, that the median one way delay is equal in both directions. Thus, the estimation method is applicable for our measurements to derive one-way delays. The proposed method overcomes unsynchronized clocks and linear clock drifts. Note that the one-way delays are only required in Section 4.2.2 to verify the emulated end-to-end one-way delays. However, we will additionally use them to show alternative metrics for jitter.

**Jitter** The term jitter is used to express delay variations within a stream of received packets. In literature, there exist different definitions of how to assess the jitter. The most common ones are (a) the standard deviation of the one-way delay  $\omega = \sigma_{OWD}$  and (b) the inter-packet delay variation  $\sigma_{IPDV}$  as defined in RFC 3393 [66]. The standard deviation of the round trip delay is also a common measure, however, it cannot be used in the context of VoIP, as the packets are neither acknowledged nor returned to the sender.

After computing the one-way delays  $d_p$  for all received packets  $p \in s_{in}$ , the standard deviation  $\omega$  of the one-way delays simply follows as

$$\omega = \text{STD} [d_p | p \in s_{in}] = \sqrt{\frac{1}{|s_{in}| - 1} \left( \sum_{p \in s_{in}} d_p^2 - \left( \sum_{p \in s_{in}} d_p \right)^2 \right)}. \quad (4.7)$$

A different common metric for expressing jitter uses the inter-packet delay variation IPDV as defined in [66]. The IPDV compares the one-way delays of a selected pair of packets within a stream. It is defined as the difference between



the one-way delays  $d_p$  and  $d_q$  of the packets  $p$  and  $q$ . It holds

$$IPDV(p, q) = d_p - d_q = (t_{r,p} - t_{s,p}) - (t_{r,q} - t_{s,q}) \quad (4.8)$$

$$= (t_{r,p} - t_{r,q}) - (t_{s,p} - t_{s,q}) \quad (4.9)$$

$$= \Delta t_{r,p,q} - \Delta t_{s,p,q} . \quad (4.10)$$

Thus, the IPDV of two packets is the difference of the inter-packet delay in the outgoing stream of packets  $s_{out}$  and the inter-packet delay in the received stream  $s_{in}$ . As measure for the jitter of a packet stream, the standard deviation of the IPDV any two consecutively received packets is computed as follows:

$$\sigma_{IPDV} = \text{STD} [IPDV(p_{in,i}, p_{in,i+1}) | 1 \leq i < m] . \quad (4.11)$$

**Packet Reordering** As a consequence of delay variations in a stream of packets, it might occur that packets are reordered. Depending on the actual implementation, an application might be able to handle jitter by using an appropriate jitter buffer, however, reordered packets might be more difficult to deal with on application layer and hence result into significant QoE degradations. This performance issue was revealed during the course of this work for one application under study. Therefore, we also investigate this phenomenon and its influence on the QoE, although in the Internet, packet reordering is indeed possible, but seldom observed.

There exist different metrics for quantifying packet reordering. In [136], a detailed introduction on the necessity of different packet reordering metrics is given and the computation of the metrics is proposed. In general, a received packet  $p \in s_{in}$  is referred to as *reordered packet* if and only if there is at least one packet  $q \in s_{in}$  which was sent after  $p$ , i.e.  $t_{s,p} < t_{s,q}$ , but arrives before the packet  $p$ , i.e.  $t_{r,q} < t_{r,p}$ . We formally define

$$p \text{ is reordered} \Leftrightarrow \exists q \in s_{in} : t_{s,p} < t_{s,q} \wedge t_{r,q} < t_{r,p} . \quad (4.12)$$

The ratio  $\rho_{s_{in}}$  of reordered packets within a stream of packets is denoted as

*Type-P-Reordered-Ratio*, or reordering ratio in short. It is calculated as

$$\rho_{s_{in}} = \frac{|\{p \in s_{in} | p \text{ is reordered}\}|}{|s_{in}| - 1}. \quad (4.13)$$

The reordering ratio is a very simple metric, as it does not take into account how “much” a single packet is reordered. This can be exemplarily illustrated. Let  $s_A$  be packet stream with 8 packets,  $s_{out,A} = \{p_1, \dots, p_8\}$ . If  $p_8$  arrives for some reasons before  $p_4$ , but all other packets are sent in correct order, the received packet stream is  $s_{in,A} = \{p_1, p_2, p_3, p_8, p_4, p_5, p_6, p_7\}$  and the resulting reordered ratio is  $\rho_{s_{in,A}} = 1/2$ , as  $p_4, \dots, p_7$  are reordered according to the definition above. However, an application might only drop packet  $p_8$  while the other packets are processed correctly, as only the packet arriving out of order cannot be processed. If the stream  $s_A$  is received as  $s_{in,B} = \{p_2, p_1, p_4, p_3, p_6, p_5, p_8, p_7\}$ , we obtain the same reordering ratio  $\rho_{s_{in,B}} = 1/2$ . Later, we will see that this metric is sufficient to describe the relationship between packet reordering and QoE, as some applications like SJPhone used for testing the IQX hypothesis seems to have problems with reordered packets.

A more complex metric to quantify packet reordering is the *mean reordering late time* of a packet stream [136]. The reordering late time is the maximum distance in time from a reordered packet to the earliest packet received that has a larger sequence number. If a packet is in-order, its reordering late time is undefined. The first packet to arrive is in-order by definition and has undefined reordering late time. This metric seems appropriate to capture the network disturbance as perceived on application layer. A formal definition is

$$\tau = \frac{1}{|Z|} \sum_{i \in Z} t_{r,i} - t_{r,j}, \quad (4.14)$$

with  $Z = \{p \in s_{in} : p \text{ is reordered}\}$ ,  $j = \min\{k | 1 \leq k < i\}$ , and  $t_{r,k}$  as measured arrival time of packet  $k$ .

**Mean Opinion Scores** For the quantification of the QoE, we use a full reference metric, i.e. we compare the sent signal with the received one offline. Our measurement testbed allows to capture the audio signals on the sender and the receiver side and allows to apply the full reference metric after a measurement run. In particular, we use the mean opinion score (MOS) [79] to express the QoE of the VoIP call. Therefore, the audio file sent is compared with the received wav-file using the Perceptual Evaluation of Speech Quality (PESQ) method described in ITU-T P.862 [64]. The resulting PESQ value can be mapped into a subjective MOS value according to ITU-T Recommendation ITU-T P.862.1 [80]. The MOS can take the following values: (1) bad; (2) poor; (3) fair; (4) good; (5) excellent.

## 4.2.2 Verification of the Emulation of Network Conditions

Although NIST Net is a common tool for emulating network conditions, we conducted several test runs to investigate whether the desired network conditions are correctly emulated or not. Summarizing, NIST Net correctly emulates (a) uncorrelated packet loss with input parameters (i) packet loss probability  $p_L$  and (ii) correlation factor  $r_L = 0$ , and (b) correlated as well as uncorrelated delays with input parameters (i) average delay  $\mu_d$ , (ii) the standard deviation of the delay  $\sigma_d$ , and (iii) the correlation factor  $r_d$ . However, correlated packet loss streams are not correctly emulated which we show later. Before that, we discuss uncorrelated packet loss and uncorrelated delay and jitter.

### Emulation of Packet Loss

For verifying the emulation of uncorrelated packet loss, we investigate the inter-packet loss distance  $K$ , that is the number  $K$  of received packets between two consecutive packet losses. For a given packet loss probability  $p_L$ , the inter-packet loss distance follows a geometric distribution and  $P[K = i] = p_L \cdot (1 - p_L)^i$  for  $i = 0, 1, 2, \dots$  in case of uncorrelated loss. Figure 4.7 compares the theoretical

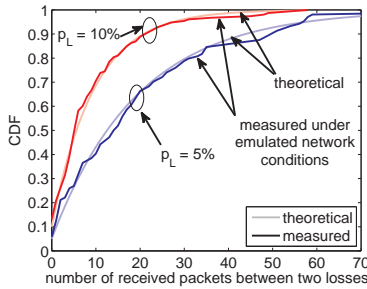


Figure 4.7: Verification of the emulation of uncorrelated packet loss

and the measured cumulative distribution functions of the inter-packet loss distance for  $p_L = 0.1$  and  $p_L = 0.05$ . For sufficiently long test runs, the measured packet loss ratio  $\tilde{p}_L$  approaches the preset dropping probability, i.e.  $\tilde{p}_L \rightarrow p_L$ .

### Emulation of Delay and Jitter

For verifying the emulation of delays, we consider the CDF of the one-way delay  $d_p$  for any packet  $p$  transmitted from sender to receiver. NIST Net offers the possibility to use different delay distributions. In our measurements, we use the normal distribution with parameter  $\mu_d$  for the average delay and  $\sigma_d$  for the standard deviation of the delay. Figure 4.8 shows the CDF of the one-way delays for  $\mu_d \in \{0 \text{ ms}, 90 \text{ ms}\}$  and  $\sigma_d \in \{1 \text{ ms}, 5 \text{ ms}, 10 \text{ ms}\}$ . Again, we can see that the theoretical and the measured curves agree. Thus, NIST Net correctly emulates delay and jitter as desired.

Note that an average delay  $\mu_d$  of 0 ms means that NIST Net does not add any additional delays before relaying a packet. As the packets are transmitted via Ethernet from sender to receiver, we obtain a minimal transmission time  $d_0$  for the one-way delay which is around  $d_0 = 0.3 \text{ ms}$ . NIST Net internally generates pseudo random numbers following a normal distribution to delay packets. As negative delays do not make sense, NIST Net sets negative values to

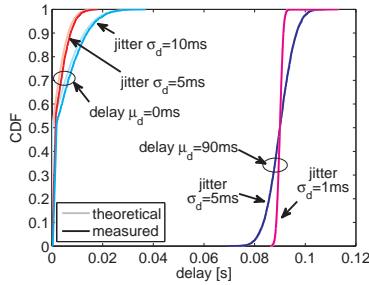


Figure 4.8: Verifying the emulation of uncorrelated jitter and delay

0ms which explains the probability of 50% for the minimal one-way delay  $d_0$ ,  $P[d_p = d_0] = 0.5$ .

A deep inspection of the source code of NIST Net revealed that NIST Net is optimized with respect to computational time at the cost of accuracy. In particular, NIST Net can only generate random delay values which in case of the normal distribution lie in the interval  $[\mu_d - 4\sigma_d; \mu_d + 4\sigma_d]$ . However, the probability for delay values to be larger than  $\mu_d + 4\sigma_d$  is negligible and it holds  $P[d_p > \mu_d + 4\sigma_d] = (1 + \operatorname{erf}(\frac{4}{\sqrt{2}}))/2 = 3.17 \cdot 10^{-5}$  for normally distributed delays with the Gauss error function  $\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt$ .

## Emulation of Autocorrelated Packet Streams

In NIST Net, the emulation of autocorrelated packet streams is basically approximated by a first-order autoregressive process AR(1), which is formally described as  $y_i = x_i \cdot (1 - r) + y_{i-1} \cdot r$ . For generating the next random value  $y_i$  the fraction  $r$  of the previous random value  $y_{i-1}$  is taken into account which leads to an autocorrelation of  $r$  at lag 1.

However, the current implementation does not correctly emulate autocorrelated packet losses which would be one possibility to produce bursty losses. We deeply investigated the source code and found out that the error stems from in-

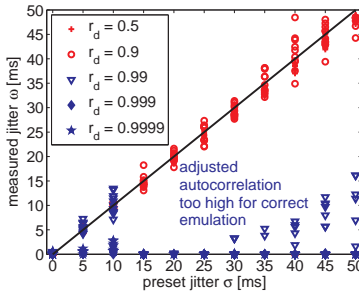


Figure 4.9: Measured standard deviation of the one-way delays  $\omega$  depending on jitter  $\sigma$  preset in network emulation package NIST Net for different autocorrelation settings  $r = r_d$

ternal conversions between 16 bit and 32 bit integer values. A formal mathematical proof that for a given packet loss probability  $p_L$  and a correlation factor  $r_L$  NIST Net generates a packet stream with a measured packet loss ratio  $\tilde{p}_L = p_L$  and  $\tilde{r}_L = 0$  (instead of  $\tilde{r}_L = r_L$ ) can be found in the technical report [153].

Next, we check the emulation of autocorrelated delay values by transmitting the audio file from sender to receiver. This results into roughly 1700 IP packets. Figure 4.9 plots the measured standard deviation  $\omega$  of one-way delays on the y-axis against the given jitter values  $\sigma_d$  passed to NIST Net on the x-axis. We varied the correlation factor  $r_d$  assigned to NIST Net from 0.5 to 0.9999. Independently of the preset correlation factor  $r_d$ , labeled with  $r$  in Figure 4.9, the measurement results should lie on the line  $\omega(\sigma_d) = \sigma_d$ . For  $r_d < 0.9$ , the generated delay values are as desired. Table 4.1 shows that the measured autocorrelation fits roughly to the preset value for  $r_d < 0.9$ . However, for very large correlation factors  $r_d \geq 0.99$ , NIST Net does not correctly emulate the given parameter settings. We therefore investigate the impact of autocorrelated delay values for correlation factors  $r_d \in \{0.5, 0.9\}$  only. In the following, we will investigate whether delay correlations in packet streams have an impact at all on the QoE.

Table 4.1: Measured autocorrelation of the one-way delays  $\omega$  depending on jitter  $\sigma$  preset in network emulation package NIST Net for different autocorrelation settings  $r_d$

preset	runs	mean	std.	min.	max.	99% conf. int.	
$r_d = 0.5$	109	0.444	0.023	0.396	0.508	0.438	0.449
$r_d = 0.9$	109	0.883	0.014	0.855	0.922	0.879	0.886
$r_d = 0.99$	109	0.442	0.399	-0.066	0.993	0.342	0.542

### 4.3 QoE of Voice Codecs iLBC and G.711

The measurements presented here were conducted during January 2007 and April 2007 at the Routerlab of the University of Würzburg. We test the IQX hypothesis for different preset QoS parameters, which are packet loss, delay and jitter. For quantifying these QoS parameters, we use the metrics as defined in Section 4.2.1. For each of the QoS parameter setting ten individual measurement runs were repeated to gain statistically significant data. In the following figures, Figure 4.10 – Figure 4.14, a single dot represents a single measurement run with the measured QoS value as obtained by the packet trace on the x-axis and the observed mean opinion score on the y-axis.

To demonstrate whether an exponential interdependency between the QoS and the QoE can be observed when varying a single QoS parameter, we fit the measurement data as described in Section 4.3.1. The resulting exponential model function is plotted in each corresponding figure, the obtained optimal parameters of Eqn. (4.3) are annotated, as well as the coefficients of determination  $R^2$  are given as goodness-of-fit measure.

In the following, the used hardware and software of the measurement testbed are explained. Detailed information on the hardware of the used machines is given in Table 4.2. An overview of the actual versions of the software and the used operating systems can be found in Table 4.3.

**Hardware Configuration** The measurement testbed is set up in a local area network without any connection to the Internet to avoid any noise traffic or cross traffic. The testbed comprises two client machines *A* and *B* for the voice communication, and a dedicated machine *D* for emulating the network conditions. The LAN is realized with Ethernet and the voice client machines are connected via crossover-cables to the emulation machine. *D* has an additional network interface that is used to control the measurements remotely. The voice clients *A* and *B* are located in different subnetworks and both use the network emulation machine *D* as routing gateway, hence, the complete traffic between *A* and *B* can be influenced by *D*, e.g. by introducing additional delays or dropping IP packets.

Table 4.2: Overview of the hardware configuration

Name	sender <i>A</i>	emulator <i>D</i>	receiver <i>B</i>
Role	Client	Router	Client
CPU	2 x Intel Pentium III		
	1.3 GHz	500 MHz	1.3 GHz
RAM	512 MB		
HDD	80 GB	16 GB	40 GB
NIC	3COM, 100 Mbps		
	1 x	3 x	1 x

**OS and Software** For our experiments, we use the *SJPhone VoIP application* (<http://www.sjllabs.com>) for several reasons. First, SJPhone implements different voice codecs, among others, the iLBC and the G.711 voice codecs, in which we are interested in this study for quantifying QoE and testing the IQX hypothesis. The SJPhone software allows to explicitly use a specified codec via the GUI or by adjusting a parameter file (in the Linux version). Second, SJPhone is open-source software that enables direct voice calls between any two hosts. Thus, the end hosts do not need to register at any SIP server in the Internet. The call initiator has to know the IP address of the machine to be called and then the



call is directly established via SIP or H.323. The used session protocol suite can also be configured via the parameter file. In our measurements, we use direct SIP calls. Third, SJPhone can be controlled from the command line and configured via parameter files without using the GUI. This was a mandatory requirement to automate the measurement process. As a consequence, the measurements could be repeated many times to get statistically significant data while reducing the human efforts for conducting the measurements.

On the voice client machines, Knoppix Linux is used as operating system. During the course of this work it has been found out that conducting the measurement process with SJPhone running on Windows makes the voice client machines crash for some reasons. Additional software tools which are used in the context of this work are *aumix*, *play*, *sound-recorder*, and *tcpdump*. They are already included in the used Knoppix 5.1.1 distribution. At the sender side, *play* makes the audio file be played locally and *aumix* allows to redirect the sound output as input for SJPhone. On the receiver side, *sound-recorder* is used to capture the received audio signals and record them into a file which is later on compared with the sent audio file to obtain the QoE. *Tcpdump* is used to capture packet traces on OSI layer 2 at the sending and the receiving voice client machines in order to get statistics on QoS parameters. The network emulation machine *D* runs SuSe Linux and hosts *NIST Net*.

Table 4.3: Overview of the used software versions

Name	Version
NIST Net	2.0.12c
SJPhone	v.1.60.299, 09.24.05
Aumix	2.8
Play	(sox) 2.0-debian
Sound-recorder	0.06 (Oct 28 2005)
Tcpdump	3.9.5

### 4.3.1 Approach to Test the IQX Hypothesis

For the investigation of the interdependency between QoS parameters and the QoE for voice calls, we emulate various network conditions, like packet loss or jitter. The testbed setup allows (a) to capture packet traces at the end hosts, which is required to compute QoS parameters and (b) to capture the sent and the received audio signals required to obtain MOS as QoE parameter. The main goal is then to quantify the relationship between QoS and QoE. In particular, we investigate whether this relationship can be expressed by a simple exponential function with appropriate parameters.

The model function  $f(x) = \alpha \cdot e^{-\beta x} + \gamma$  as derived in Eqn. (4.3) mathematically expresses the mapping from the value  $x$  of the considered QoS parameter to the QoE measure, i.e. MOS. The parameters  $\alpha, \beta, \gamma$  of the model function are retrieved by means of non-linear regression. We used the optimization toolbox of Matlab to find an optimal fitting function for the given measurement points. Optimal in this case means to find the unknown parameters  $\alpha, \beta, \gamma$  in Eqn. (4.3) such that the mean squared error  $E^2$  is minimized. The mean squared error is defined as the average of the squared residuals  $r_i^2 = (f(x_i) - y_i)^2$  for all  $n$  measurements  $(x_i, y_i)$  with a measured QoS value  $x_i$  and a measured MOS  $y_i$ :

$$E^2 = \frac{1}{n} \sum_{i=1}^n r_i^2 = \frac{1}{n} \sum_{i=1}^n (f(x_i) - y_i)^2 . \quad (4.15)$$

The goodness-of-fit for the model function  $f(x)$  can be measured with different metrics, like the coefficient of correlation  $R$  between the model function and the measured data, or the coefficient of determination  $R^2$ . The latter can be computed as follows:

$$R^2 = 1 - \frac{\sum_{i=1}^n (y_i - f(x_i))^2}{\sum_{i=1}^n (y_i - \bar{y})^2} \quad (4.16)$$

with  $\bar{y} = \frac{1}{n} \sum_{i=1}^n y_i$ . A value close to one means a perfect match between the

model function and the measured data. Other common metrics are functions of the residuals which show a perfect match between model and measurements if the value is close to zero. Examples are the mean squared error  $E^2$  or the normalized mean squared error  $NMSE = E^2 / \text{VAR}[y_i]$  which is normalized by the variance of the measured MOS values. We use the coefficient of determination  $R^2$  to test the IQX hypothesis and to show the goodness-of-fit of the proposed exponential model function for the obtained measurement results.

### 4.3.2 Voice Quality Affected by Loss

We start to investigate the influence of packet loss on the user perceived quality. Figure 4.10(a) and Figure 4.10(b) show the measurement results for the iLBC and the G.711 codec, respectively. In these experiments, the packet loss  $p_L$  was varied from 0% up to 40% in steps of 1%. Furthermore, we performed the measurements without any additional delay ( $\mu_d = 0$  ms) and with an additional delay of  $\mu_d = 90$  ms emulated by NIST Net.

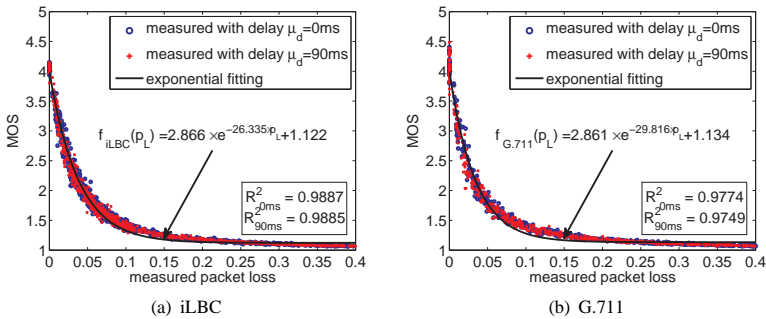


Figure 4.10: Measurement results and obtained mapping function  $f_{iLBC}(p_L)$  between packet loss ratio  $p_L$  and MOS for the iLBC codec

The first observation is that there is a clear exponential relationship between the packet loss ratio and the MOS for iLBC as well as G.711. The results show that the IQX hypothesis holds for this scenario. Thus, the QoE degradation is very strong when the packet loss ratio increases slightly. For iLBC, the MOS is 4 without any loss, 3 for 1.6 % packet loss, and 2 for 4.5 % packet loss. For G.711, the MOS is also 4 without any loss, 3 for 1.4 % packet loss, and 2 for 4 % packet loss. The second observation is that the additional delay of 90 ms has no influence on this relationship – which is expected, as only large delays above 200 ms have an additional impact on the QoE according to ITU-T G.114, cf. [78].

### 4.3.3 Jitter and Reordering

Next, the influence of jitter on the QoE is investigated. In the experiments, we vary the jitter  $\sigma_d$  from 0 ms to 30 ms in steps of 1 ms, and afterwards in steps of 5 ms up to 80 ms. Again, we executed the measurements without any additional delay  $\mu_d = 0$  ms and with an additional delay of  $\mu_d = 90$  ms. In this case, different results for both average delay values are expected as the variability of the delay values generated by NIST Net follows a normal distribution with parameters  $\mu_d$  and  $\sigma_d$ . We will see that as a consequence of the jitter, packet reordering occurs, which decreases the user perceived quality. Describing this influence on the application with an appropriate packet reordering metric allows to verify again the IQX hypothesis for both codecs. However, their performance differs significantly, and we therefore start to provide the results for iLBC before the G.711 results are depicted.

#### iLBC

We first investigate the jitter value  $\sigma_d$  as QoS parameter to test the IQX hypothesis. Figure 4.11(a) reveals that the measurement values scatter much more around the exponential fitting function than for the packet loss curves in the previous section. Obviously, for a certain jitter setting, the absence of extra delay ( $\mu_d = 0$  ms) leads to higher MOS values than a scenario with an average delay of 90 ms.

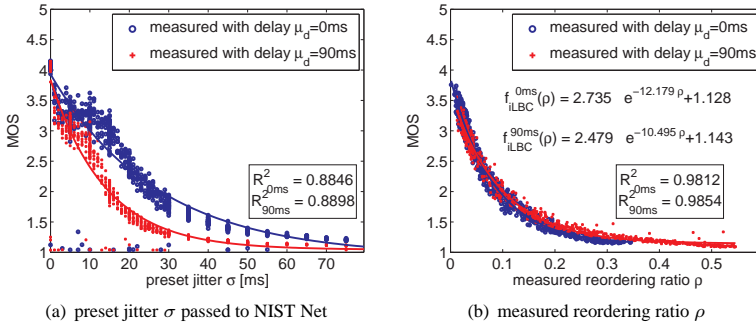


Figure 4.11: Measurement results and obtained mapping function  $f_{iLBC}(\rho)$  between jitter / reordering and MOS for the iLBC codec

Typically, real-time applications like VoIP or video streaming are able to handle jitter up to a certain level by using a jitter buffer. This explains why for small jitter values below 10 ms the curves are more flat and the QoE degradation is not so strong with increasing jitter, especially for  $\mu_d = 0$  ms. After that the MOS again show exponential decays. As the fitting is done for all variations of  $\sigma_d$ , the obtained mapping function from QoS to QoE shows a worse coefficient of determination.

However, in the experiments described above, the delay values are randomly generated and uncorrelated. Hence, packets might overtake each other and packet reordering occurs. Therefore, we use now as metric the packet reordering ratio  $\rho$  to quantify the QoS. To highlight this clearly, we use the MOSs and packet traces from the measurements as in Figure 4.11(b), but as QoS metric we calculate  $\rho$  instead using  $\sigma_d$ .

As a result of Figure 4.11(b), we clearly observe an exponential relationship between the QoE and the QoS. We obtain as large goodness-of-fit values as for packet loss and hence confirm again the IQX hypothesis. The main result of this section is that the important challenge consists in finding the appropriate QoS

metric for describing the effect of the QoS influence on the QoE. In this particular case, this means that SJPhone gets into trouble when packets are reordered. Obviously, on application layer, packet reordering has a similar impact as packet loss. If packets are reordered, they are not processed any more by SJPhone. In particular, it is possible to convert the packet reordering ratio  $\rho$  to a packet loss ratio  $p_L$  such that the same MOS values are obtained, formally  $f(p_L) = f(g(\rho))$ . From the results in Figure 4.10(a) and Figure 4.11(b), we compute the conversion function  $g$  for iLBC and  $\mu_d = 90$  ms:

$$p_L = g(\rho) = \max\{0.3837 \cdot \rho - 0.0054, 0\}. \quad (4.17)$$

This means that the packet loss is a linear function of the reordering ratio.

Table 4.4: Mean squared errors  $E^2$  of the IQX hypothesis for different QoS metrics applied to describe the impact of jitter; metric names are chosen according to Section 4.2.1 and RFC 4737 [136]

Parameters	iLBC with delay		G.711 with delay	
	0 ms	90 ms	0 ms	90 ms
mean reordering ratio $\rho$	0.097	0.067	0.063	0.036
mean reordering extent	0.089	0.072	0.040	0.035
mean n-reordering	0.086	0.061	0.041	0.030
mean reordered late time $\tau$	0.108	0.091	0.056	0.036
mean n-reordering late time	0.087	0.066	0.040	0.032
inter-packet delay variation $\sigma_{IPDV}$	0.158	0.110	0.258	0.243
std. dev. of one-way delays $\omega$	0.158	0.112	0.259	0.241
preset jitter $\sigma_d$ passed to NIST Net	0.191	0.151	0.255	0.244

Table 4.4 shows the mean squared errors  $E^2$  of the exponential mapping function between QoS and QoE when applying different QoS metrics to describe the impact of jitter. The QoS metrics are defined as in Section 4.2.1. We additionally give the results for some more common metrics, as defined in [136], without explicitly showing the fittings. Table 4.4 includes the results for iLBC and G.711 while the delay is either 0 ms or 90 ms. From the table, we conclude that in all

scenarios the packet reordering metrics reveal the relationship to the QoE better than the pure jitter metrics. However, this is not a general statement, in particular, this is caused by the fact that the used application has problems with packet reordering, which affects the user-perceived quality.

### G.711

The impact of jitter on the QoE is analogously examined for the G.711 voice codec. In Figure 4.12(a), the jitter value  $\sigma_d$ , which is passed as input parameter to NIST Net, is used as QoS parameter. The same observations as for iLBC are obtained. For a certain jitter value  $\sigma_d > 0$ , a lower average delay leads to a higher MOS. If the jitter values are below 10 ms, the curves are quite flat and the QoE degradation is not so strong with increasing jitter. For larger jitter values, the QoE in terms of MOS decays. However, the decay is not so strong as for iLBC. This is caused by the fact that as soon as jitter appears, i.e. even for  $\sigma_d = 1$  ms, the MOS drops down to a value of 2, i.e. the quality is already poor. Note that for  $\sigma_d = 0$  ms the MOS is about 4, i.e. good quality.

An explanation for this can be found when investigating the sending pattern of the SJPhone application. Even though the G.711 codec is defined with a constant packet sending rate of 50/s, SJPhone uses intervals of length 32 ms to send packets. In order to achieve the desired bitrate, several packets are sent back-to-back. In detail, we observed the following pattern of time intervals in milliseconds between two consecutively sent packets: 0, 32, 32, 0, 32, 32, 0, 32. In total, this leads to an average time of 20 ms between two packets. Thus, the codec mean bitrate is realized, but the single inter-packet delay varies. An inter-packet delay of 0 ms means that two packets are sent back-to-back, i.e., the second packet is immediately sent after the first one. For the implementation of G.711 in SJPhone, this means that 37.5 % of the packets are sent together. As a consequence, even a very small jitter like  $\sigma_d = 1$  ms might lead to packet reordering and causes a strong QoE degradation. For  $\sigma_d = 1$  ms, we already obtain a packet reordering ratio of roughly 15 %.

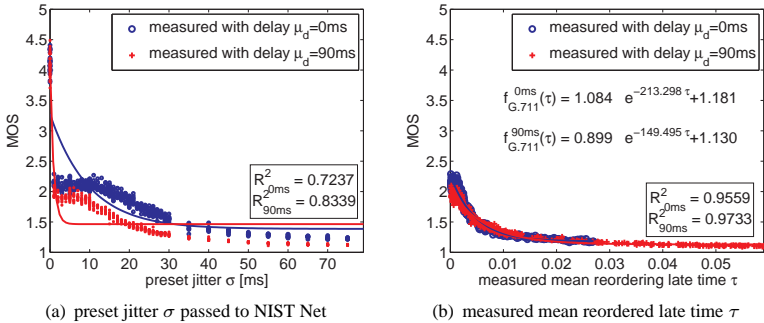


Figure 4.12: Measurement results and obtained mapping function  $f_{G.711}(\tau)$  between preset jitter  $\sigma$  / mean reordered late time  $\tau$  and MOS for the G.711 codec

In Figure 4.12(b), we use the mean reordered late time  $\tau$  to describe the impact of jitter and the resulting packet reordering as QoS parameter. Again, the IQX hypothesis can be confirmed as an exponential relationship between QoS and QoE is observed.

### 4.3.4 Autocorrelated Packet Streams

Up to now we have investigated the impact of uncorrelated packet streams. In the context of packet loss, this means that packets are dropped randomly. As a consequence of uncorrelated delays, much more packet reordering occurs than for correlated delay which might be caused e.g. by queues at router along the end-to-end path. In the previous section, we have already seen that for the actual implementation of the G.711 codec in SJPhone very small jitter values result in a high packet reordering ratio. For iLBC in contrast, this weird application phenomena was not observed. Therefore, we focus on the iLBC codec using SJPhone when investigating autocorrelated packet streams. As NIST Net does not correctly emulate autocorrelated packet loss, we generate bursty losses by dropping  $n$  subsequent



packets. In particular, we investigate the impact of  $n \in \{0, \dots, 300\}$  consecutively lost voice datagrams on the QoE. Before that, we take a closer look at correlated delay values, which are correctly generated by NIST Net.

### Autocorrelated Delay Values

In Section 4.2.2, we have already shown that NIST Net correctly emulates delay values for any correlation factor  $r_d \leq 0.9$ , that is the measured delay values show an average delay  $\tilde{\mu}_d$ , a standard deviation  $\tilde{\sigma}_d$ , and an autocorrelation  $\tilde{r}_d$  which correspond to the parameter settings preset in NIST Net. In the scenario, we consider  $\mu_d = 90$  ms and no packet loss  $p_L = 0$ , while the jitter is varied in the range  $\sigma_d \in [0 \text{ ms}; 50 \text{ ms}]$ . As correlation factor, we use either  $r_d = 0.5$  or  $r_d = 0.9$ , named as  $r$  in Figure 4.13(a) and Figure 4.13(b).

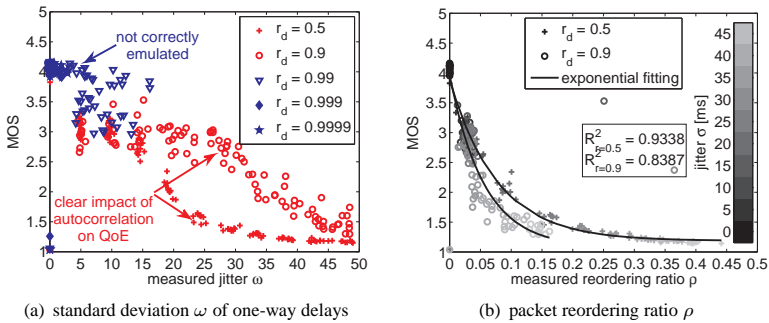


Figure 4.13: Measurement results for iLBC with autocorrelated delay values

Figure 4.13(a) shows the measured standard deviation  $\omega$  of the one-way delay vs. MOS. The different colors respective grey levels of the dots indicate the preset NIST Net setting. It shows that independently of the correlation factor, the measured delays  $\omega$  meet the preset jitter values  $\sigma_d$ . Furthermore, there is a clear difference between the curves for the different correlation factors. For  $r_d = 0.9$

the obtained MOS values are larger than for  $r_d = 0.5$ . This is expected as a larger correlation reduces the reordering of packets.

Therefore, we describe the impact of the QoS on the QoE using the packet reordering ratio  $\rho$ . Figure 4.13(b) shows the measurement results using  $\rho$  instead of  $\omega$ . For a high correlation factor  $r_d = 0.9$ , we obtain a reordering ratio  $\rho \in [0; 0.15]$  according to the preset jitter  $\sigma_d$ . This means at most 15 % of the packets are reordered even for a jitter of 50 ms. A lower correlation factor  $r_d = 0.5$  means that the one-way delays of consecutive delays do not depend so strongly on each other. As a result, a packet reordering ratio up to 45 % emerges for  $\sigma = 50$  ms. Nevertheless, for the same reordering ratio  $\rho$ , the observed MOS is higher for less-correlated delay  $r_d = 0.5$  than for strongly correlated ones,  $r_d = 0.9$ . Note that in Figure 4.13(b), the majority of measurement results for  $r_d = 0.9$  shows a reordering ratio  $\rho < 5\%$  and MOS values larger than 2.5. In contrast, for  $r_d = 0.5$ , the reordering ratio goes up to 25 % and MOS values are typically found close to 1.5, with some exceptions. As a main result, both curves can be well fitted by an exponential distribution. However, the actual curves strongly depend on the correlation factor. A more detailed analysis and the usage of different network emulator environments to investigate autocorrelated packet streams is a topic of future work.

### Bursty Losses

Finally, the impact of bursty losses on the QoE is examined. As we know that NIST Net cannot be used for the emulation of bursty losses by adjusting the correlation factor for packet loss, this investigation was performed in a different way. On a local machine, we packetized the audio signal using the iLBC codec and dropped selected voice datagrams. To be more precise, we dropped  $n$  consecutive voice datagrams starting from voice datagram  $n_0$ . After that, the remaining voice datagrams were passed to the iLBC codec to be decoded as audio signal. Accordingly, the QoE was derived in the same manner as described in Section 4.2.1.

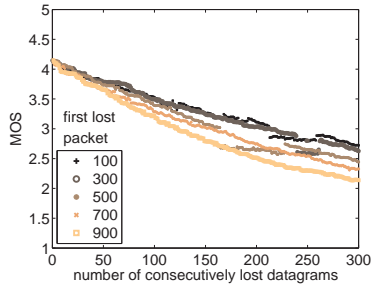


Figure 4.14: *Impact of bursty losses on the QoE for iLBC*

Figure 4.14 shows the number  $n$  of consecutively lost datagrams on the x-axis and the MOS on the y-axis. We also varied over  $n_0$  which denotes the first lost packet. Obviously, the larger  $n$ , the worse the MOS becomes. For  $n \leq 50$  there are no significant differences between the different curves for the first lost packet  $n_0$ . However, larger  $n > 50$  make the curves disperse. Note that this corresponds to a silent period of 1.5 s and it is not clear whether the PESQ and MOS computation is able to correctly map this silence period on the real user experienced degree of satisfaction. Indeed, if the silence period is too long, a user will probably abort a call. However, this is hardly considered in this computation.

One more remarkable observation is that  $n = 50$  consecutively lost packets mean a packet loss ratio of  $\bar{p}_L = 3\%$ , as the transmitted voice file has a length of 51 s consisting of 1700 iLBC voice datagrams. However, the observed MOS value of roughly 3.7 is much higher than for the same packet loss ratio with randomly dropped packets, yielding a MOS value of 2.4. But it has to be noted that in this last experiment, the voice signals were locally encoded and decoded, but not transmitted via the testbed. Therefore, we suggest to modify NIST Net or use a different network emulator which easily allows to investigate bursty loss models.

## 4.4 Skype VoIP Traffic in UMTS

After quantifying the impact of network disturbances on the QoE when using the voice codecs iLBC and G.711, we investigate now Skype VoIP traffic in a UMTS environment. We installed Skype on two end hosts *A* and *B* and used different network entities and access types to establish an end-to-end connection between the two hosts, cf. Figure 4.15. The main focus of our studies is on how the current network conditions influence the QoE of the end user and in how far the Skype application reacts to quality degradations. In this context, the network entity located in the middle of our testbed is used to emulate typical problems in wireless network environments. In particular, we investigate to what extent the results depend on the way the network is emulated and if there are differences to measurements in a real UMTS networks. Therefore, we apply two different emulation approaches, one based on hardware and one based on software as introduced in Section 4.2. Additionally, we perform measurements in the German UMTS network.

Both sender *A* and receiver *B* were running Windows XP. Packet traces were captured using the latest version of Windump on each machine. The network entity was connected to the Internet which is necessary for Skype to run on the end-hosts. If not stated otherwise, we used Skype Version 1.20.37 running on Windows XP.

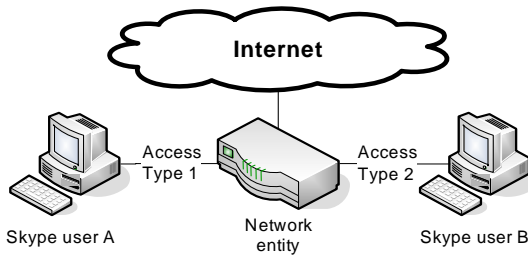
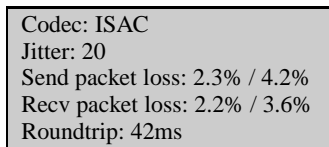


Figure 4.15: Measurement setup of the Skype experiments

Skype offers the possibility to display technical information about an ongoing call as indicated in Figure 4.16. From this we learned that it selects its audio codec according to the performance of the hardware it is running on. Since we were interested in typical UMTS measurements we used CPUs with 500 Mhz on host *A* and host *B* which roughly reflects the processing power of actual mobile UMTS devices and forces Skype to use the simple iLBC codec. To gain insight into the behavior of future generation mobile devices, we replace both machines in later measurements with state of the art hardware which allows Skype to use its adaptive multirate codec iSAC. Note, that the processing power of the user equipment is in no way the limiting factor in our measurements. The different hardware at the end hosts is merely used to force Skype to use its different codecs.

Comparing the recorded audio files at host *B* to the original audio files sent by host *A* we are also able to measure the QoE of Skype VoIP calls. To obtain a reference MOS value for our measurements we encoded the original audio file (optimal MOS of 5.0) with the iLBC codec which results in a slight degradation of the voice quality and a MOS value of 4.17. However, the Skype application is well secured against being evaluated. It refuses, e.g., to start if a debugger is installed on the system [124]. Similarly not all versions of Skype allowed us to directly record its audio output. In such cases we forwarded the audio signal to another machine using an audio cable as shown in Figure 4.17. This resulted in another slight degradation of the voice quality and a MOS value of 4.08.



Codec: ISAC  
Jitter: 20  
Send packet loss: 2.3% / 4.2%  
Recv packet loss: 2.2% / 3.6%  
Roundtrip: 42ms

Figure 4.16: *Excerpt of the technical information shown by Skype*

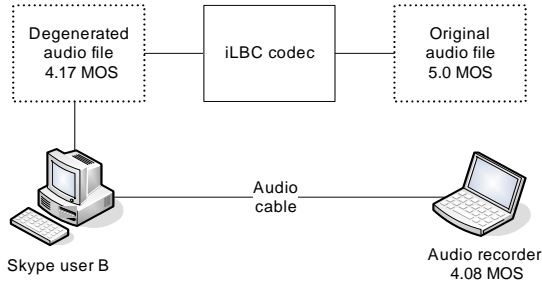


Figure 4.17: Degradation of MOS value caused by measurement methodology

#### 4.4.1 Emulated Rate-Controlled DCH in UMTS

In UMTS systems, the conditions of the wireless channel are changing over time because of radio propagation effects or fading. On rate-controlled dedicated channels (DCHs), this results in a slow adaptation of the bandwidth currently assigned to the user [26]. In order to analyze whether such DCHs suffice to carry Skype VoIP calls, we regard a simplified LAN scenario. In particular, we use a Cisco router as traffic shaping network entity and emulate the dynamically changing conditions of the DCH by restricting the bandwidth of the ongoing Skype call. Initially, we increased the bandwidth from 16 kbps to 32 kbps, 64 kbps, 128 kbps, and 384 kbps, respectively. Since during our measurements we observed that the measured MOS values are very sensitive to small changes in the range between 16 and 64 kbps, we also measured 24 kbps, 28 kbps, 40 kbps, 48 kbps, and 56 kbps.

Depending on the currently available bandwidth and processing power, Skype uses different codecs to maintain reasonable call qualities [123]. In this measurements we assumed mobile devices with up to 500 Mhz, which forced Skype (Version 1.20.37) to use iLBC [88], a simple audio codec with a fixed packet size and a fixed inter-packet transmission time. Since this particular version of Skype did not allow us to record its output directly, we forwarded the audio to a separate recorder as illustrated in Figure 4.17. In order to evaluate and compare the

perceived voice quality for the different bandwidth values, the same audio data was transmitted for each measured bandwidth.

### Characterization of Skype's iLBC Traffic

The throughput achieved by VoIP calls using Skype's iLBC codec is shown in Figure 4.18(a) in dependence of the available bandwidth. It includes the payload (67 byte) as well as the UDP and IP headers (28 bytes). Each scenario was repeated between five and ten times in order to produce credible emulation results. The figure shows the mean values of the different emulation runs as well as the corresponding minimum and maximum. Skype did neither adapt the sending rate to the available bandwidth nor to the resulting packet loss in any of the emulation runs. The sender constantly uses a bandwidth close to 26 kbps, independent of the quality of the communication channel. The communication partner receives a throughput, which corresponds to the currently available bandwidth on the link, the remaining packets are lost on the bottleneck link.

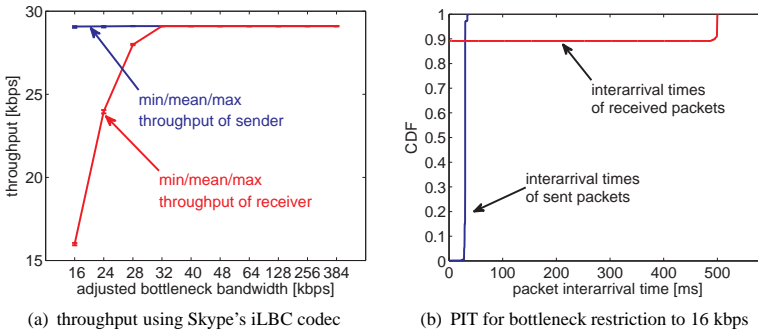


Figure 4.18: Rate-controlled DCH scenario: characterization of Skype's iLBC traffic when using a traffic shaping router for network emulation

In order to understand the details of the bottleneck in this scenario, we focus on a single emulation run. Figure 4.18(b) shows the CDF of the packet interarrival time for both the sender and the receiver of the VoIP call, using a bottleneck speed of 16 kbps. The sudden jump from 0 to 1 at the CDF of the sender illustrates an almost constant time of 30 ms between two sent packets. At first glance, the CDF of the receiver has a very unexpected shape. About 90 percent of all packets have an interarrival time of practically 0 ms, while the time between the remaining packets is about 500 ms.

This behavior is explained in the following. The buffer in the router was set to 8000 bit, while simultaneously limiting the speed of the link to 16 kbps. Skype used a total packet size of 872 bit. Thus, at most 9 packets ( $872 \cdot 9 = 7848$  bit) fit into the buffer of the router. To emulate a link speed of 16 kbps, the router fills its buffer and delays the data for exactly 500 ms. This way, a bandwidth of  $8000 \text{ bit}/500 \text{ ms} = 16 \text{ kbps}$  is achieved on a physical 100 Mbps link. This has two major implications. At first the interarrival time of the packets within a burst is  $872 \text{ bit}/100 \text{ Mbps}$ , which is in the range of  $1 \mu\text{s}$  and explains the shape of the CDF in Figure 4.18(b). Secondly, packet loss occurs in bursts during the 500 ms, in which the buffer of the router is delayed. In Section 4.4.2 and Section 4.4.4 we will see that Skype adapts its bandwidth usage to packet loss as soon as it no longer occurs in bursts but randomly.

### Relationship between End-to-End QoE and Packet Loss

To evaluate the speech quality as perceived by the end user, we have a closer look at the MOS value for the emulation runs described in the previous section, i.e. between five and ten emulation runs per scenario. Figure 4.19 illustrates the achieved MOS values (cf. left y-axis) for different link speeds between 16 kbps and 384 kbps and relates them to the observed packet loss (cf. right y-axis). The higher the packet loss, the lower is the corresponding MOS value. Nevertheless, the quality is sufficient to enable mobile Voice-over-IP via rate-controlled dedicated channels.



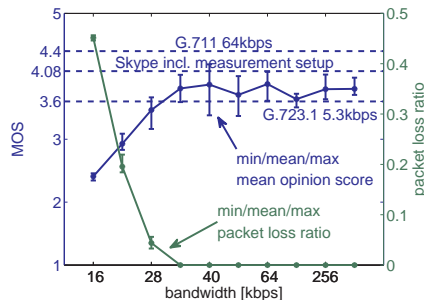


Figure 4.19: Rate-controlled DCH scenario: MOS value related to packet loss

There is no more packet loss above a link speed of 29 kbps since up from this point the throughput of the sender (109 byte/30 ms) is smaller than the available bandwidth on the link. The corresponding MOS values oscillate around a value of 3.8. Since the MOS value is a very sensitive performance measure, the fluctuations can be explained by the stochastic influences of the network, like jitter. For a better classification, the figure also shows reference MOS values for the G.711, the G.723.1, as well as for Skype's iLBC codec, when evaluated locally. Transmitting the audio packets over the network obviously results in a slight degradation of the MOS value achieved by the iLBC codec. It has to be noted that the mapping function between MOS and packet loss for the iLBC codec as found in Section 4.3.2 cannot be applied in this scenario using the traffic shaping router, since the packets are not lost randomly but in a bulk.

#### 4.4.2 Replication of Voice Data

The scope of this section is to investigate if the applied network emulator and the character of packet loss affects the behavior of the Skype application. When using a traffic shaping router to restrict the link bandwidth as in the previous scenario in Section 4.4.1, Skype did not react to the occurring packet loss and

the sender's throughput did not change. This hardware-based network emulation caused a high autocorrelation of lost packets, as the limited size of the queue deterministically determines which packets get lost. This approach should be used to simulate losses due to congestion. In this section, however, we consider a lossy link with independent and random packet losses generated by a software emulator. In such scenarios, Skype shows a different traffic profile which will be characterized in the following. This particular application behavior is generalized and evaluated analytically in order to study the impact it has on the perceived QoE.

### Burst Losses vs. Randomly Lost Packets

The dummynet software is an easy-to-use application for emulating queue and bandwidth limitations, delays, or packet losses. It works by intercepting communications of the IP layer and emulating the desired effects. It is described in detail in [57]. The dummynet software is installed on a BSD Linux machine which also acts as gateway for both machines of Skype user  $A$  and  $B$ . To emulate a lossy link between  $A$  and  $B$ , the individual packets are dropped at random with probability  $p_L$ , where  $p_L = 0$  means no loss and  $p_L = 1$  makes all packets be dropped. The packet loss value can be dynamically changed during the VoIP call. For the machines of the Skype users, we use the same hardware and software configuration as in the previous Section 4.4.1.

Figure 4.20 shows the throughput of the sender and the throughput at the receiver over time. In the considered scenario, we start with no loss. Due to the same configuration of the end users, the iLBC codec is used for encoding the audio data. Every 30 ms a packet of fixed size is sent. In this measurement scenario, however, the payload of a packet is 58 byte (instead of 67 byte in Section 4.4.1). After roughly 90 s,  $p_L$  is set to 70%. Skype user  $A$  still sends with  $m_{sent} = 22.93$  kbps including the UDP and IP header of 28 bytes, while user  $B$  only receives  $m_{rcvd} = (1 - p_L) \cdot m_{sent} = 6.88$  kbps on average. However, after another 25 s, Skype reacts to the detected packet loss and increases the bandwidth of the sender to  $m_{sent} = 8 \cdot (115 + 28) \text{ byte}/30 \text{ ms} = 38.13$  kbps by changing

the payload of a packet to 115 byte. As a result, the received goodput increases accordingly. We decreased the packet loss value over time to 50 %, 10 %, and 0 % respectively, whereas the sender's throughput only then switched back to the original 22.93 kbps when no more packet loss was detected.

This implies that Skype sends redundant information to overcome the effects of a lossy link and to maintain a certain QoE. Considering the payload of both packets, we see that the data information is nearly doubled. The simplest approach to send redundant information is to replicate the entire voice information of a single audio frame and put it into two consecutive IP packets. This observation is the motivation to evaluate in general what impact the replication of an audio frame in  $\kappa$  consecutive packets has on the end-to-end QoE (cf. Section 4.4.2).

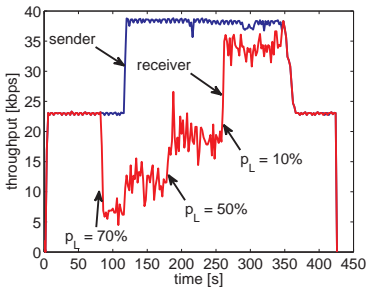


Figure 4.20: *Bandwidth adaptation for random losses*

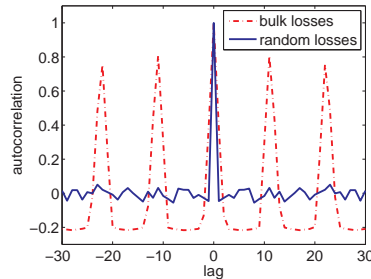


Figure 4.21: *Autocorrelation of packet loss for different lags*

The results of this experiment show that Skype reacts differently in different scenarios. As shown in Section 4.4.1, cf. Figure 4.18(a), Skype does not react to burst losses and keeps sending with a constant throughput. For randomly lost packets, however, Skype adapts its bandwidth usage at runtime, cf. Figure 4.20. This means Skype recognizes and distinguishes between the different reasons for packet loss, like congested or lossy links. This is referred to as edge-based

intelligence. The question is on what grounds Skype decides how to react? One possibility is the distance  $L$  between two consecutive packet losses.  $L$  is referred to as inter-packet-loss distance. The number of consecutive packets without any packet loss is denoted as  $K = L - 1$ . In the dummynet scenario, the random variable  $L$  follows a geometric distribution shifted by one:  $L \sim \text{GEOM}_1(q)$  with parameter  $q = \frac{\mu-1}{\mu}$  and a mean measured distance  $\mu$ . When using the traffic shaping router, losses occur in a bulk. Hence, the probability that the inter-packet-loss distance is one is very high, in our scenario  $P[L = 1] = 0.87$ .

Another possibility is to use the autocorrelation of the observed packet loss. If we have random packet losses, the autocorrelation is close to zero for an arbitrary lag  $l \neq 0$ . If the losses occur in a bulk however, a high positive correlation can be detected for appropriate lags. Hence, a regular pattern of the autocorrelation for different lags can be observed, cf. Figure 4.21.

Summarizing, Skype implements an edge-based intelligence to react to packet loss. The specific network characteristic generated by the applied measurement setup has a significant influence on the observed traffic profile. When detecting independent and random losses, redundant information is sent to maintain a certain QoE. This observation was the starting point for the investigation of dynamic changes during a VoIP call, which will be discussed in detail in Section 4.4.4.

### Analytical Evaluation of the Impact of Replication on QoE

Based on our experiences gained from Skype, we propose the replication of voice datagrams as a possible solution to overcome a QoE degradation due to packet loss. This is the simplest approach to smoothen the effect of packet loss. We assume the iLBC voice codec, i.e. every  $\Delta t = 30$  ms, a voice datagram of size  $s_{voice} = 400$  bit is sent. A *replication degree*  $\kappa$  signifies that the voice datagram is additionally sent in the following  $\kappa - 1$  packets.

As a consequence, each packet now contains  $\kappa$  voice datagrams with a total packet size of  $s_{packet} = s_{header} + \kappa \cdot s_{voice}$ . The variable  $s_{header}$  denotes the overhead for each packet caused by UDP and IP headers (8 byte + 20 byte) as well

as by the link layer (e.g. 14 byte for Ethernet). Hence, the required bandwidth is a linear function in  $\kappa$ :

$$C_{req} = \frac{s_{header} + \kappa \cdot s_{voice}}{\Delta t}. \quad (4.18)$$

The advantage gained by this bandwidth consumption is the reduction of the effective voice datagram loss probability  $1 - p_{voice}$ . For a given packet loss probability  $p_L$  and a replication degree  $\kappa$ , a voice datagram only gets lost if all  $\kappa$  consecutive packets containing this voice datagram get lost. Thus, the probability  $p_{voice}$  that a voice datagram is successfully received is

$$p_{voice} = 1 - p_L^\kappa. \quad (4.19)$$

In Section 4.3.2, we derived a relationship between the effective voice datagram loss probability and the obtained MOS value for the iLBC codec,

$$f_{iLBC}(p_L) = 2.866 \cdot e^{-26.335p_L} + 1.122. \quad (4.20)$$

This relationship is used in the following numerical example. The effect of the voice datagram replication can be seen in Figure 4.22(a) for a replication degree of  $\kappa = 1, \dots, 6$ . On the x-axis the packet loss probability  $p_L$  is denoted. The QoE on the y-axis is computed according to Eqn. (4.20) whereby the voice datagram probability in Eqn. (4.19) is used. For  $\kappa = 1$  and  $p_L = 0.1$  we obtain  $p_{voice} = 0.9$  and a MOS value as low as 1.33. Increasing the replication degree to  $\kappa = 2$  and  $\kappa = 3$  leads to  $p_{voice} = 0.99$  and  $p_{voice} = 0.999$ , respectively. The corresponding MOS values are 3.33 and 3.91, respectively. This shows that the QoE could be improved from a poor quality to a good quality. A further increase of the replication degree only yields a small gain compared to the growth of the required bandwidth  $C_{req}$ .

Besides the increased bandwidth consumption, however, the replication also causes some jitter, as the voice datagrams are not received every  $\Delta t = 30$  ms,

but may only be successfully transmitted in one of the  $\kappa - 1$  following packets. We therefore quantify the jitter by computing the probability  $\tilde{y}(i)$  that a voice datagram is successfully transmitted in the  $i$ -th try.

$$\tilde{y}(i) = p_L^{i-1} \cdot (1 - p_L) \quad (4.21)$$

The probability that a voice packet is received follows as

$$p_{voice} = \sum_{i=1}^{\kappa} \tilde{y}(i) = (1 - p_L) + p_L(1 - p_L) + \dots + p_L^{\kappa-1}(1 - p_L), \quad (4.22)$$

which agrees to Eqn. (4.19). The number  $Y$  of trials which is required to successfully transmit a voice datagram is a conditional random variable. It follows a shifted geometric distribution and is defined for  $1 \leq i \leq \kappa$ :

$$Y \sim \frac{\text{GEOM}_1(p_L)}{p_{voice}} \quad (4.23)$$

with

$$y(i) = \frac{\tilde{y}(i)}{p_{voice}} = \frac{p_L^{i-1} \cdot (1 - p_L)}{1 - p_L^{\kappa}}. \quad (4.24)$$

In the case of a constant bitrate codec, the jitter  $\sigma_{IPDV}$  using the inter-packet delay variation, cf. Section 4.2.1, simplifies to the inter-packet delay in the received stream  $\Delta t_r$ . For the sake of simplicity, we assume a deterministic inter-packet sent time  $\Delta t$  and a deterministic delay  $t_{s \rightarrow r}$  from the sender to the receiver. It holds  $\sigma_{IPDV} = \text{STD}[\Delta t_r - \Delta t] = \text{STD}[\Delta t_r]$ .

We normalize the jitter  $\sigma_{IPDV}$  by the average time  $\Delta t$  between any two sent packets,

$$j = \frac{\text{STD}[\Delta t_r]}{\Delta t}. \quad (4.25)$$

With  $\Delta t_r = Y \Delta t$  the jitter can – after some algebraic transformations – be expressed as

$$\begin{aligned} j &= \frac{\sqrt{\mathbb{E}[(Y \Delta t)^2] - \mathbb{E}[Y \Delta t]^2}}{\Delta t} = \sqrt{\mathbb{E}[Y^2] - \mathbb{E}[Y]^2} \\ &= \sqrt{\frac{p_L}{(p_L - 1)^2} - \frac{p_L^\kappa \cdot \kappa^2}{(p_L^\kappa - 1)^2}}. \end{aligned} \quad (4.26)$$

Figure 4.22(b) shows the normalized jitter  $j$  for replication degrees  $1 \leq \kappa \leq 6$  in dependence of the packet loss probability  $p_L$ . Eqn. (4.26) is an exact formula, which we also validated by implementing a simulation. The solid lines correspond to the analytical calculation of the jitter, while the dashed lines show the simulation result of a (short) single run. Both curves agree and the confidence intervals of several simulation runs are too small to be visible.

The cost of the voice datagram replication – besides the increased bandwidth consumption – is an increased jitter. However, jitter also impacts the QoE and is of course one impairment factor in Eqn. (4.1). As a result, a maximal degree  $\kappa_{max}$  of replication exists and a further increase does not improve the QoE anymore. ITU-T G.114 recommends a latency of the end-to-end delay of 150 ms, referred to as toll quality, and a maximum tolerable latency of 400 ms. According to the end-to-end delay  $t_{s \rightarrow r}$  and the inter-packet sent time  $\Delta t = 30$  ms, the following inequation must also hold

$$\kappa \cdot \Delta t + t_{s \rightarrow r} < t_{max} \quad (4.27)$$

for a maximum allowed latency  $t_{max}$ . For example with  $t_{max} = 200$  ms and  $t_{s \rightarrow r} = 10$  ms, the maximum replication degree is limited to  $\kappa_{max} \leq 6$ .

In general, the replication of voice datagrams is an effective way to reduce the impact of packet loss on the user perceived quality. The price to pay is a higher amount of consumed bandwidth. The benefit of the replication degree, however, is limited by the arising jitter and the higher latency of individual packets. In

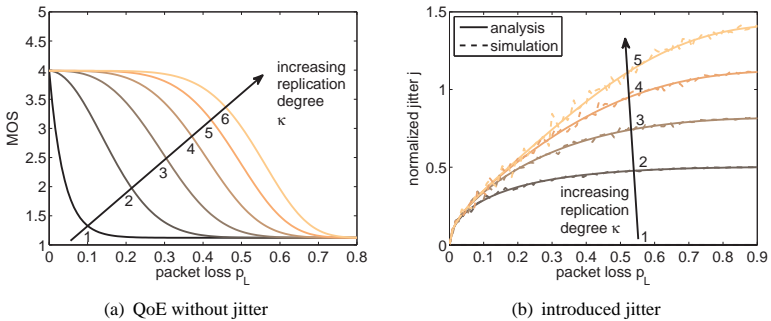


Figure 4.22: Impact of replication degree  $\kappa$

addition, in the analytical consideration we neglect the correlation between the consumed bandwidth and packet loss. If the packet loss is caused by congestion in the network, the additionally introduced bandwidth will worsen the network situation. An intelligent application should therefore try to estimate the cause of packet loss (e.g. using the autocorrelation as illustrated in the last section), or try to measure the effects of the increased bandwidth and to back off in case of unsuccessful counter measures.

### 4.4.3 Measurement in a Public UMTS Network

In this section we regard UMTS scenarios where one Skype user is connected to the Internet using a public German UMTS operator. We used a Vodafone Mobile Connect UMTS PC-card as modem for the machine. During the course of the measurements, only dedicated channels (DCH) of fixed bandwidth were used. While the uplink capacity is limited to 64 kbps, the downlink direction offers a bandwidth of 384 kbps. The second Skype user is connected via DSL and has a capacity of 128 kbps in the uplink and 1024 kbps in the downlink. To account for the essential technological difference between uplink and downlink in UMTS, we



have a separate look on both directions and regard two different scenarios. We investigate the *uplink scenario* in which the UMTS subscriber sends the audio data with 64 kbps to the DSL user. In the *downlink scenario* the DSL user sends its data over the 128 kbps link to the UMTS user. The measurements took place in July 2005 and each scenario was repeated ten times if not stated otherwise.

During all measurements the clients used a constant bitrate codec for the main audio connection, sending 108 byte of payload every 60 ms. Again, a derivate of the iLBC codec was used which was also indicated by the technical information shown during a call by the Skype application. However, in the UMTS scenario a different variation of this voice codec was used than in the LAN measurements in which a traffic shaping router or a software tool emulated typical UMTS network characteristics, see also Table 4.9 on page 191. Since this codec was used starting with the first audio packet, Skype seems to choose this codec based on local information, like access type (modem or LAN) to the Internet, or due to exchanged packets before measuring the link quality. This assumption is supported by the fact, that emulating the exact link properties of the UMTS scenarios (delay, bandwidth, etc.) with dummynet did not cause Skype to use the same codec. In the UMTS scenario, there was nearly no packet loss in any of the experiments. In total, 11 out of 15417 packets were lost. However, the MOS values are lower than before because of the network jitter. In the following we concentrate on the packet interarrival times (PIT) at the sender (PIST) and at the receiver (PIAT).

#### **Uplink: UMTS subscriber sends to wireline user**

In the uplink scenario the UMTS client uses a 64 kbps connection to send its data to the DSL user, which has a maximum download capacity of 1024 kbps. Figure 4.23(a) shows the CDF of the PIT for both the sender and the receiver. The UMTS client constantly sends a voice packet every 60 ms. However, due to the jitter in the network the PIATs at the receiver side are spread around the mean. The almost symmetric shape of the CDF reflects the fact that for every delayed packet there is obviously a packet with a correspondingly smaller PIAT.

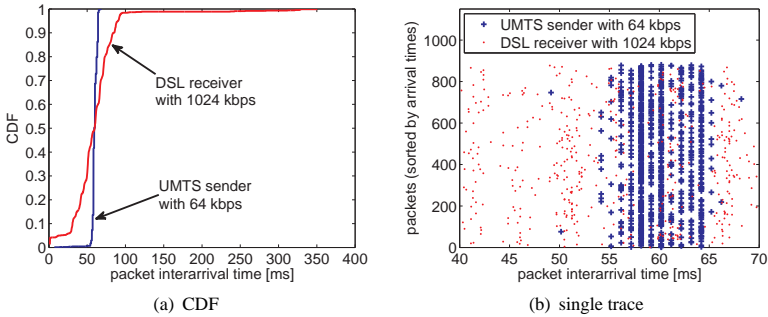


Figure 4.23: Uplink scenario: packet interarrival times

To illustrate this effect, Figure 4.23(b) plots the PIT for each packet at the sender and the receiver. There were 878 packets in this scenario. The x-axis shows the PIT zoomed from 40 ms to 70 ms, the y-axis plots the packet of the corresponding number as sorted by their arrival time. That is, the plot shows how many packets arrived with a specific PIT. As expected, the packets of the DSL user are randomly distributed due to the jitter in the network (red dots in the figure). The UMTS packets, however, are sent at a discrete resolution of 1 ms as can be seen by the blue crosses forming vertical lines in the figure. Note that this discretization already happens at the sender and is thus locally influenced, probably by the PCMCIA UMTS card. We are therefore able to exclude buffer effects and the like in the core network for the same discretization in the downlink scenario.

Table 4.5 presents a more detailed view on some key performance measures for the uplink scenario using an observation window of 300 ms. During each observation period the throughput at the sender and the goodput at the receiver are captured. In particular, Table 4.5 shows the average throughput ( $m_{sent}$ ,  $m_{rcvd}$ ) and the average deviation ( $s_{sent}$ ,  $s_{rcvd}$ ) for ten different runs of the experiment (row labeled with ' $\mu$ ') as well as the corresponding standard deviation over the ten runs for each measure (row labeled with ' $\sigma$ '). Since there is almost no packet

Table 4.5: Key performance measures for UMTS uplink scenario

	throughput		goodput		MOS
	average $m_{sent}$	deviation $s_{sent}$	average $m_{rcvd}$	deviation $s_{rcvd}$	
$\mu$	18071.58 bps	2300.95 bps	18055.23 bps	3497.57 bps	3.19
$\sigma$	8.84 bps	568.87 bps	21.20 bps	858.38 bps	0.13

Table 4.6: Received packets in the UMTS uplink scenario

type	payload	number	mean PIAT	std. PIAT
A	3 byte	3	20.02 s	10.73 ms
B	108 byte	847	61.97 ms	35.00 ms
C	112 byte	28	1.92 s	27.05 ms
B or C	108.13 byte	875	60.00 ms	2.55 ms

loss in this scenario, the throughput of the receiver corresponds to the throughput of the sender. The corresponding standard deviation (8.84 bps) of the individual runs is close to zero, as the same codec with a fixed payload size and PIST was used in each of the ten experiments. However,  $s_{sent}$  and  $s_{rcvd}$  differ by about 1200 bps in the uplink. Due to the jitter in the network the observed PIATs are almost uniformly spread around the mean PIAT, which is also reflected in the lower MOS value (3.19) as compared to the bottleneck LAN scenario (3.76) with a bandwidth restriction to 64 kbps.

To highlight these effects in more detail, Table 4.6 shows the packets received at the DSL client during a single run of the experiment. The 3 byte packets are used for quality feedback. However, this specific Skype codec uses two types B and C of packets (108 byte and 112 byte) in the main audio stream. Thereby every 32th packet is of size 112 byte, which explains why the mean PIAT of the 108 byte packets is 61.97 ms instead of 60 ms. The PIAT is exactly 60 ms when we do not differ between type B and C. The high standard deviation of the PIAT of these packets confirms our previous statements. Packet type A might be used for quality feedback, packet type B is used for pure audio data, and C for audio data

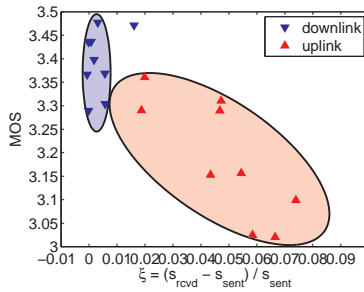


Figure 4.24: MOS scatter plot as function of the jitter for UMTS

and signaling information. The same behavior was observed in all nine remaining experiment runs.

Figure 4.24 shows the obtained MOS values for the UMTS measurements in both directions, the uplink and downlink. We used a MOS scatter plot as function of the measured throughput jitter  $\xi$  which is the normalized difference of the standard deviation of the throughput of the sender  $s_{sent}$  and the receiver  $s_{rcvd}$  for an observation window of 300 ms. The red upward-pointing triangles represent the results from the uplink scenario, the blue downward-pointing triangles the results from the downlink scenario accordingly. The higher the jitter, the lower is the MOS value. In all experiments, the uplink reveals higher jitter and hence lower quality than the downlink. In addition, the occurring jitter in the uplink shows a larger amplitude (0.01-0.08) than the downlink (0-0.006). There is only a single outlier in the downlink with a jitter of 0.0161.

#### Downlink: Wireline user sends to UMTS subscriber

In this scenario we regard the opposite direction, where the DSL user sends its voice data over a 128 kbps link to the UMTS user, who has a downlink capacity of 384 kbps. Thereby the interesting effects occur on the link from the base station to the UMTS mobile.

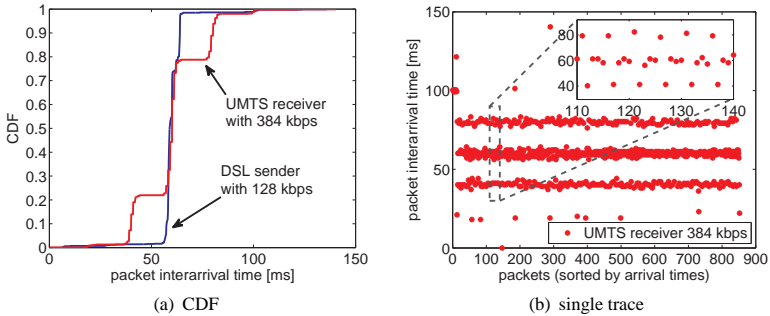


Figure 4.25: Downlink scenario: packet interarrival times

Figure 4.25(a) shows the CDF of the PIT at the DSL sender and the UMTS receiver. Like before, the packets are sent into the network exactly every 60 ms. The UMTS receiver, however, registers a different behavior of the incoming packets. The PIAT of the arriving packets is no longer uniformly spread around the mean PIAT, but mainly takes three discrete values, 40 ms, 60 ms, and 80 ms. The difference of these values, corresponds to the UMTS Transmission Time Interval (TTI) value which is typically 20 ms. As can be seen by the CDF, about 60 percent of all packets arrive with a PIAT of 60 ms at the UMTS receiver. Approximately every 5th packet misses the corresponding TTI, cf. Figure 4.25(b), and subsequently arrives with a PIAT of 80 ms. Therefore the next packet, which hits the correct TTI, has a PIAT of 40 ms instead of 60 ms. This means that 20 percent of all packets have a PIAT of 80 ms and 40 ms, respectively.

Table 4.7 gives a more detailed view of the key performance measures in the downlink scenario. The mean throughput of the receiver again corresponds to the throughput of the sender. This time,  $s_{sent}$  and  $s_{rcvd}$  do not differ as much as in the uplink scenario. Thus, the network should have less influence on the user perceived quality of the audio connection. The MOS is indeed higher in the downlink scenario (3.39) than before in the uplink scenario (3.19), cf. Figure 4.24.

Table 4.7: Key performance measures for UMTS downlink scenario

	throughput		goodput		MOS
	average $m_{sent}$	deviation $s_{sent}$	average $m_{rcvd}$	deviation $s_{rcvd}$	
$\mu$	18023.77 bps	1848.15 bps	18007.08 bps	2172.39 bps	3.39
$\sigma$	48.16 bps	282.70 bps	51.64 bps	284.97 bps	0.068

Table 4.8: Received packets in UMTS downlink scenario

type	payload	number	mean PIAT	std. PIAT
A	3 byte	6	9.46 s	4.49 s
B	21 byte	14	1.73 s	3.58 s
C	108 byte	817	61.32 ms	16.00 ms
D	112 byte	16	3.20 s	45.02 ms

Note that the standard deviations in the last row of Table 4.7 are slightly higher, since the number of quality feedback packets varied in the different runs.

Table 4.8 summarizes the four different packet sizes at the UMTS receiver in this scenario. Again for the most part 108 byte packets were used for the audio connection, while this time only every 54th packet had a payload of 112 byte. In exchange, there is a new packet type using 21 byte. This kind of packet was also used in the audio connection, replacing some of the 108 byte packets. However, they were sent very irregularly as can be seen by the high standard deviation of their PIAT. The same irregularity was obtained for the 3 byte packets, which did not have a deterministic PIT of 20 s but were sent every 10 s on average with a standard deviation of 4.49 s. What exactly triggers Skype to use this specific variation of the codec is subject to further study.

#### 4.4.4 Emulate Dynamic Changes in UMTS

So far, we investigated the influence of packet loss and bandwidth restrictions on a Skype VoIP call when using low power machines with a CPU power of 500 MHz. In that case, the iLBC codec was used which only requires low com-

putational power. We have seen that Skype is an edge-based application which reacts to the network conditions in order to maintain a certain QoE.

Edge-based applications apply traffic control on application layer and thereby shift the intelligence to the edge of the network. The goal is to maintain a certain QoE, independent of the current network conditions, by performing quality control and adaptation. When the network conditions change, the application has to react on this with appropriate mechanisms. We have already seen that the voice quality of the Skype service is maintained by using appropriate voice codecs, cf. Table 4.9.

Currently, mobile phones with a CPU of 400 MHz and smartphones with 600 MHz are available, but in near future microprocessor units for multimedia applications will be included in the mobile devices, like the OMAP [144]. In the following, we will use more powerful machines of 1.3 GHz in order to reveal all edge-based intelligence mechanisms and offered features of Skype. In that case, a better, but more complex codec is used, the iSAC codec, which is implemented by Global IP Sound [142]. It is a wideband, adaptive codec designed to deliver high quality sound in both high-bitrate and low-bitrate conditions. The adaptation of the codec is done by adjusting transmission rates to increase the listening experience for the current network situation. It requires about 6–10 MIPS.

In these measurements we used the latest available Skype version 2.0.0.81 (February, 2006) and NIST Net 2.0.12c, a Linux-based network emulation toolkit developed by the National Institute of Standards and Technology (NIST). To study the behavior of Skype under dynamic changes in the network, we emulate varying packet loss and different round trip times (RTT) during a VoIP call.

### **QoE Adaptation by Edge-Based Intelligence**

First, we investigate Skype's reaction to the current packet loss of the end-to-end connection. This QoE adaptation can be illustrated by a measurement study presented in [36]. The standard audio wav-file of length 51 s is played in a loop with a pause of 9 s in between.

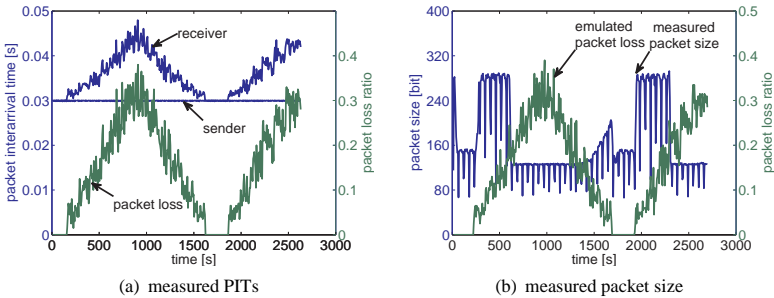


Figure 4.26: *Dynamic change of packet loss and Skype's bandwidth adaptation*

During the measurements we gradually increased the packet loss up to 30 percent, decreased it back to zero percent, and again increased it to 30 percent as shown on the right y-axis in Figure 4.26(a). The left y-axis shows that the PIST of the sender remains unaffected at 30 ms (with a measured standard deviation of 6.65 ms) during the entire process. The PIAT of the receiver, however, increases and decreases according to the current packet loss rate. Figure 4.26(b) shows how the Skype software reacts to this kind of packet loss. The measured packet loss ratio on the right y-axis denotes how many packet got lost, whereby we used the average for a window size of 6 s. On the left y-axis, the average size of the voice packets on application layer is plotted in bit. Again, we used a window size of 6 s corresponding to 200 voice packets. Initially, the Skype call is established between user *A* and *B* without any packet loss on network layer. The size of a packet varies between 90 bit and 190 bit, resulting in an average of 150 bit. The oscillations of the packet size are due to our measurement setup, as during the 9 s pause interval, Skype still sends small packets of size 50 bit.

After 5 minutes we start to increase the packet loss probability by about 5 % every two minutes, until the packet loss probability reaches 30 %. The time interval of two minutes was chosen to ensure that Skype has enough time to react to



changes. We noticed that Skype needs about one minute to adapt its voice codec to the current network conditions. As we can see in Figure 4.26(b), Skype reacts to the experienced QoE degradation in terms of packet loss by increasing the packet size. The new size ranges between 240 bit and 320 bit with an average of 280 bit. In contrast to before, the packet size is nearly doubled. This implies that Skype now sends redundant information within every voice packets in order to maintain the QoE. However, as soon as a certain threshold is exceeded (in this case about 20 % packet loss), the packet size is decreased again to a lower average value of 125 bit as compared to the original average packet size. This indicates a change in the used voice codec. As soon as the packet loss probability falls below a certain threshold, the sender rate is again adapted by increasing the packet size.

This measurement clearly shows that Skype in fact tries to keep the QoE above an acceptable threshold. This is done by adapting the amount of consumed bandwidth. If the receiver's application detects packet loss, it instructs the sender to increase the bandwidth. For a VoIP call, this is easily possible, since the connection is full duplex and the connection from user  $B$  to user  $A$  is used to send the feedback information.

### Application-Driven Re-Routing

Finally, the impact of the round trip time on the quality of a Skype call is evaluated. Therefore, we repeatedly played the audio file five times while a constant delay from machine  $A$  to machine  $B$  is set. Note, that we only disturb the direct connection between  $A$  and  $B$ . The one-way delay  $\mu_d$  from  $A$  to  $B$  and vice versa is varied from 0 s to 4 s.

Figure 4.27 shows the MOS of the audio call in dependence of the one-way delay. We plotted the minimum, the average, and the maximum MOS out of the five repetitions for each delay. There is only a small influence on the voice quality for delays smaller than or equal to 250 ms, i.e.  $\mu_d \in \{0 \text{ ms}; 5 \text{ ms}; 25 \text{ ms}; 50 \text{ ms}; 100 \text{ ms}; 250 \text{ ms}\}$ . This is consistent with ITU-T G.114 which recommends a latency of the end-to-end delay of 150 ms, referred

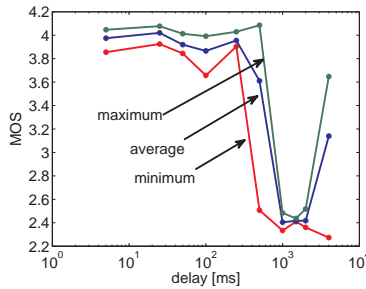


Figure 4.27: Measured MOS values while dynamically changing the e2e delay and Skype's application layer re-routing

to as toll quality, and a maximum tolerable latency of 400 ms [78].

After that a strong decay of the MOS from roughly 3.9 to 2.4 is observable for delays  $\mu_d \in \{500 \text{ ms}; 1.0 \text{ s}; 1.5 \text{ s}; 2.0 \text{ s}\}$ . Thus, the voice quality drops from good to poor. It is expected that an increase of the delay further worsens the quality. However, looking at Figure 4.27, the average MOS value increases again for a large delay of 4 s. The reason is that Skype relays the connection over a third-party machine, if the current connection becomes too bad. Thus, Skype implements *re-routing on application layer* and forms its own logical overlay. In our measurements, Skype used a different machine C in the Internet as a relay node. After 15 s, the traffic was redirected from A to C to B, instead of the direct, but disturbed connection, from A to B.

This behavior nicely demonstrates the way edge-based applications are intended to work. The current end-to-end QoS and QoE is measured and evaluated. Performance measures may be the processing power of the involved machines or the QoS of the connection, like packet loss or delay. The application reacts accordingly, e.g. by changing the voice codec, by adjusting the sender bandwidth, or by re-routing the call on application layer. Table 4.9 shows the variety of voice codecs used by Skype. The payload and the packet interarrival time is related

Table 4.9: Overview on the variety of voice codecs used by Skype

Type	Payload	Interval	Bitrate	Details
iLBC-20	38 byte	20 ms	15.2 kbps	
iLBC-30	50 byte	30 ms	13.3 kbps	
iSAC	adaptive frames		10–32 kbps	requires 6–10 MIPS
Skype-I (traffic shaping router)	67 byte	30 ms	17.9 kbps	iLBC*, Sec. 4.4.1
Skype-IIa (dummysnet)	58 byte	30 ms	15.5 kbps	iLBC*, Sec. 4.4.2
Skype-IIb (dummysnet)	115 byte	30 ms	30.7 kbps	iLBC*, packet loss detected
Skype-III (UMTS)	108 byte	60 ms	14.4 kbps	iLBC*, Sec. 4.4.3
Skype-IV (dynamic changes)	89–286 byte	18–36 ms	17.3–111.2 kbps	iSAC*, Sec. 4.4.4

to the suggested main audio datagrams. The two basic codecs which were used during the course of the measurements are the iLBC-30 and the ISAC codec. This was also displayed by Skype's technical information field. In the different scenarios, however, different derivatives of these codecs were used. Especially, the emulation of changes during a call showed the potential of Skype and the different possibilities of how to react appropriately to different network situations in order to maximize the current QoE of an user.

## 4.5 QoE Management and Provisioning

*From an operator's point of view*, it will be an increasing challenge to cope with such new edge-based applications, which are already highly popular among the users for a variety of reasons. They offer good quality, are easy to use, and provide additional functionality, for example chatting and file transfer in implemented in

Skype, but was not available in traditional telephony. Most importantly, the flat-rate cost models for ubiquitous Internet access additionally make VoIP very affordable. Moreover, operators will not be able to stop user-driven applications at the edge of the network, since the corresponding traffic cannot reliably be distinguished from regular IP traffic. However, as the traffic is transported via the Internet, there are no QoS guarantees like in regular circuit switched calls. Thus, if a network operator does not want to be reduced to a bit pipe, he needs to offer strict QoS and QoE guarantees and value-added services, like location based services in mobile environments. Therefore, *QoE management and provisioning* gets a crucial task. In this context, network virtualization may be the apparatus which allows network operators to offer and realize QoE management and provisioning.

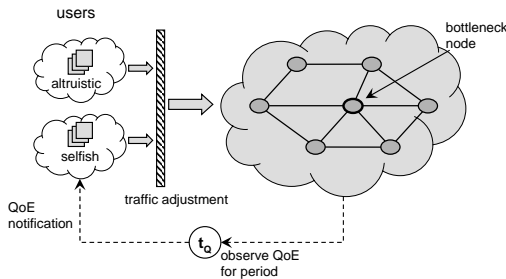


Figure 4.28: Quality assessment mechanisms for QoE of edge-based applications

From a service provider's point of view relying on edge-based intelligence, the shift of intelligence to the edge is accompanied also by the change from multi-service networks to multi-networks service. An edge-based application could use many networks with different technologies in parallel, raising the question which network has to maintain which portion of the agreed QoS. From this perspective, the QoE will be the major criterion for the subscriber of a service and the multi-network service has to maintain a certain QoE for each user. As a consequence, the edge-based application is responsible (a) to evaluate the QoE at the

end user's site and (b) to react properly on the performance degradation, i.e., that the application adapts itself to the current network situation to maintain the QoE. Figure 4.28 illustrates the QoE control scheme of such a multi-network service. Users are connected to each other via the corresponding access technologies. The QoE is assessed during a period  $t_Q$  of time. Accordingly, the altruistic users and the selfish users react on feedback obtained from measurements. It has to be noted that the selfish behavior may be implemented in the software downloaded by the user without his explicit notice. In the case of edge-based applications, QoE management is performed by the application itself, e.g. as done by Skype.

In general, QoE management and provisioning requires three basic steps, (1) understanding QoE, (2) monitoring QoE, and (3) controlling QoE. It is necessary to understand the application's requirements and the impact of disturbances on the user perceived quality. This allows for cost savings by appropriate *QoE dimensioning* and avoiding *QoE overprovisioning*, as indicated by the flat, constant shape of the mapping curve between QoE and QoS in Figure 4.2 and also by the concrete measurement results of user satisfaction related to session times for web browsing in Figure 4.5. A good understanding or even a reduced reference metric, as proposed by the IQX hypothesis, allows to easily monitor QoE at the edge or to assess QoE within the network. Then, the QoE may be controlled in such a way that the user may not get dissatisfied and even leaves the service. QoE control aims at reacting before the user reacts. Open questions in this context are (a) where to react, at the edge or within the network, (b) when to react and on which time scales, and (c) how to react and which control knobs (at the edge or within the network) to adjust.

The understanding of QoE also remains a topic of future research. For assessing QoE, a typical approach is to calculate mean opinion scores out of huge user tests. Thus, the opinions of individual users are aggregated and meant to reflect the opinion of an average user. As we have seen on the exponential interdependency of QoE and QoS parameters, the QoE might be quite sensitive in certain areas. Therefore, we propose to consider *SOS*, the standard deviation of opinion scores, in addition to MOS for properly reflecting the sensitivity of users.

## 4.6 Lessons Learned

Internet users find an ever-growing set of IP-based applications and access networks to choose from. For telephony services, the Skype VoIP application has become a strong competitor to existing telephone networks. Such multitude of offers make prices decrease and competition between service and/or network providers increase. The customer finds itself in a strong position, being able to choose between different competing providers. Given similar pricing schemes, which might be considered as a primary decision aid for many users, their subsequent choices are then likely to be influenced by expected and experienced quality, i.e. through personal ratings of the perception and price-worthiness of a service.

Consequently, the providers' interest in how users perceive usability, reliability, quality and price-worthiness has increased. A provider needs to be able to observe and react quickly upon quality problems, at best before the customer perceives them. Facing this kind of quality competition, the concept of QoE emerged, combining user perception, experience and expectations with non-technical and technical parameters such as application- and network-level QoS.

Edge-based applications like Skype impose a new control paradigm on the future Internet. Currently, they implement a rough kind of QoE management, perform QoS measurements itself and adapt the traffic process according to the perceived QoS (packet loss probability or jitter). In particular, these edge-based applications adapt the amount of consumed bandwidth to reach different goals. A selfish behaviour tries to keep the QoE of a single user above a certain threshold. Skype, for instance, repeats voice samples in view of end-to-end-perceived loss, which increases the consumed bandwidth. Altruistic behavior, on the other side, would reduce the bandwidth consumption in such a case in order to release the pressure on the network and thus to optimize the overall network performance.

The lessons learned in this chapter cover two main aspects. Firstly, we investigated on how the current network conditions described as QoS parameters influence the QoE of a VoIP user and secondly, in how far an edge-based appli-

cation like Skype reacts to quality degradations. As fundamental background, we have shown how to assess and compare quality. In particular, different quality metrics relevant for QoE and QoS evaluations were discussed and their applicability for quantifying QoE–QoS relationships evaluated. We identified a generic interdependency between QoE and QoS which is formulated as IQX hypothesis. It presents an exponential dependency of QoE from QoS. With respect to the IQX hypothesis, existing work dealing with user experience in web browsing was reconsidered and we could demonstrate that the exponential interdependency is also valid in the selected examples.

In order to quantify QoE–QoS relationships and test the IQX hypothesis for the voice codecs iLBC and G.711, we conducted measurements in a testbed which allows to control the network conditions and the corresponding QoS parameters like packet loss, one-way delay or delay jitter. The experimental setup consisted of two computers, each running the softphone SJPhone, interconnected by a third machine hosting the network emulator software NIST Net. For each preset packet loss, delay and jitter setting, the received audio file is compared to the undistorted file by software determining the mean opinion score. In case of packet loss, the exponential decay of QoE with growing QoS disturbance was clearly confirmed. While the effect of (constant) one-way delay is rather limited due to the fact that the receiver receives all packets with unchanged packet inter-arrival times, delay jitter also gives raise to exponentially-looking shapes, however with some remarkable deviations for small jitter values. A closer investigation of the traffic flow associated with SJPhone reveals the cause for this behavior, that is a pronounced sensitivity of that particular softphone to packet reordering introduced by NIST Net. Plotting the QoE against the packet reordering ratio, we again observe a clear exponential interdependency. In addition to these measurement results, we verified our testbed and in particular whether the emulated network conditions are emulated as desired. As a result, we found out that while NIST Net is capable of producing autocorrelated packet delay, it does not manage to impose autocorrelated packet loss. Thus, we cannot use the tool to emulate burst losses that can have a distinctive effect on QoE: the receiver misses a part of the

speech. Despite of these limitations, our investigations have shown that the IQX hypothesis appropriately captures the main vulnerabilities shown by the application SJPhone towards network-level disturbances, expressed in packet loss and reordering. This also shows the capability of the IQX hypothesis to identify the relevant performance metrics.

After that, the QoE of Skype calls over UMTS was measured and analyzed. Additionally, the classic QoS parameters like throughput or jitter were measured to derive a traffic profile for the proprietary Skype application. Different scenarios revealed that Skype is a multi-network service with edge-based intelligence, i.e. it forms a logical overlay and controls the VoIP traffic on application layer. We emulated the rate control mechanisms in UMTS by restricting the link bandwidth with a traffic-shaping router. In this case, Skype does not react to packet loss as caused by congestion in the network, but it constantly sends audio data. The occurring packet loss degrades the QoE, while Skype still works properly with rate-controlled DCHs. When using a software tool for emulating a lossy link, Skype generates a different traffic profile. It sends redundant information in succeeding packets, as soon as independent and random losses are detected. Hence, the edge-based intelligence tries to overcome packet loss by adapting the bandwidth in order to maximize the current QoE. The general benefit of the replication of voice datagrams was analytically investigated. The cost of the replication – besides the increased bandwidth consumption – is an increased jitter which also impacts the QoE. As a result, a maximal degree of replication can be derived up to which an increase of the QoE can be achieved. The measurements in a public UMTS network showed that the capacity offered by UMTS is sufficient to make mobile VoIP calls possible. However, due to network jitter and the use of a different codec by Skype, the MOS values are worse than those in the emulation of the bottleneck in a LAN environment. The used UMTS card sends and receives packets at discrete time instants in multiples of 1 ms. The packet interarrival times on the downlink are multiples of 20 ms, which corresponds to a common transport time interval (TTI) in UMTS. Finally, we investigated dynamic changes in the UMTS network. One possibility to maintain the voice quality of the Skype



service is the selection of appropriate voice codecs. The power of the processing unit determines whether a constant bitrate iLBC derivative or the more complex, adaptive iSAC codec is used. Another possibility is the adaptation of the bandwidth and the replication of information to overcome packet loss, even during a call. However, if the direct end-to-end connection between two users is too poor, Skype initiates re-routing on application layer by relaying the traffic over a third-party machine. This variety of mechanisms to maximize the QoE reveals the edge-based intelligence of the Skype application. Traffic engineering in future Internet is expected to follow this new paradigm.



## 5 Conclusions

In future telecommunication systems, we observe an increasing diversity of access networks. The separation of transport services and applications or services leads to multi-network services, i.e., a future service has to work transparently to the underlying network infrastructure. Multi-network services with edge-based intelligence, like P2P file sharing or the Skype VoIP service, impose new traffic control paradigms on the future Internet. Such services adapt the amount of consumed bandwidth to reach different goals. A selfish behavior tries to keep the QoE of a single user above a certain level. Skype, for instance, repeats voice samples depending on the perceived end-to-end loss. From the viewpoint of a single user, the replication of voice data overcomes the degradation caused by packet loss and enables to maintain a certain QoE. The cost for this achievement is a higher amount of consumed bandwidth. However, if the packet loss is caused by congestion in the network, this additionally required bandwidth even worsens the network situation. Altruistic behavior, on the other side, would reduce the bandwidth consumption in such a way that the pressure on the network is released and thus the overall network performance is improved.

In this monograph, we analyzed the impact of the overlay, P2P, and QoE paradigms in future Internet applications and the interactions from the observing user behavior. The shift of intelligence toward the edge is accompanied by a change in the emerging user behavior and traffic profile, as well as a change from multi-service networks to multi-networks services. In addition, edge-based intelligence may lead to a higher dynamics in the network topology, since the applications are often controlled by an overlay network, which can rapidly change in size and structure as new nodes can leave or join the overlay network in an en-

tirely distributed manner. As a result, we found that the performance evaluation of such services provides new challenges, since novel key performance factors have to be first identified, like pollution of P2P systems, and appropriate models of the emerging user behavior are required, e.g. taking into account user impatience.

However, the application of these paradigms and consideration of the user behavior does not only affect the performance of the investigated services, but additionally requires the development of new performance evaluation methods. For instance, P2P networks are typically large-scale systems with a large number of users. Investigating its performance in a mobile environment by means of simulation requires too much computation time, which prevents from a detailed performance analysis and the derivation of improved mechanisms to fulfill the user's demands. However, a good understanding of the system and the user behavior allows for abstractions in the simulation model, such that the computation time is reduced without loss of accuracy. In our particular case, an event-based approach is proposed that restricts the simulation to only those events that affect the content distribution system, like changing from one access technology to another.

As common denominator of the presented studies in this work, we focus on a user-centric view when evaluating the performance of future Internet applications. For a subscriber of a certain application or service, the perceived quality expressed as QoE will be the major criterion of the user's satisfaction with the network and service providers. The customer finds himself in a strong position, being able to choose between different competing providers. Consequently, the providers' interest in how users perceive an increase in usability, reliability, quality and (best) value for money. A provider needs to be able to observe and react quickly upon quality problems, at best before even the customer becomes aware of them. Facing this kind of competition for quality, the concept of QoE combines user perception, experience and expectations with non-technical and technical parameters such as application- and network-level QoS.

We selected three different case studies and characterized the application's performance from the end user's point of view. Those are (1) cooperation in mo-

---

bile P2P file sharing networks, (2) modeling of online TV recording services, and (3) QoE of edge-based VoIP applications. The user-centric approach facilitates the development of new mechanisms to overcome problems arising from the changing user behavior. An example is the proposed CycPriM cooperation strategy, which copes with selfish user behavior in mobile P2P file sharing system. An adequate mechanism has also been shown to be efficient in a heterogeneous B3G network with mobile users conducting vertical handovers between different wireless access technologies. In particular, the adaptation of the number of parallel upload slots of a multi-source download mechanism has shown to efficiently utilize the available resources, while a time-based cooperation strategy is introduced fostering the download from high-capacity peers in the B3G network.

The consideration of the user behavior and the user perceived quality guides to an appropriate modeling of future Internet applications. In the case of the online TV recording service, this enables the comparison between different technical realizations of the system, e.g. using server clusters or P2P technology, to properly dimension the installed network elements and to assess the costs for service providers. Technologies like P2P help to overcome phenomena like flash crowds and improve scalability compared to server clusters, which may get overloaded in such situations. Nevertheless, P2P technology invokes additional challenges and different user behavior to that seen in traditional client/server systems. Beside the willingness to share files and the churn of users, peers may be malicious and offer fake contents to disturb the data dissemination. As a consequence, reliability may be reduced because of pollution of the P2P system and the inherent downloading of useless contents. Copyright holders might exploit this e.g. to dimension the number of fake peers to save their protected contents from being illegally distributed in a file sharing system.

Finally, the understanding and the quantification of QoE with respect to QoS degradations permits designing sophisticated edge-based applications. To this end, we identified and formulated the IQX hypothesis as an exponential interdependency between QoE and QoS parameters, which we validated for different examples. Starting from a measurement of Skype, we found that the edge-based

intelligence tries to overcome packet losses by adapting the bandwidth in order to maximize the current QoE. The general benefit of the replication of voice data-grams was analytically investigated by means of the IQX hypothesis. The cost of this replication, besides the increased bandwidth consumption, is an increased jitter, which also impacts the QoE. As a result, a maximal degree of replication can be derived up to which an increase of the QoE can be achieved.

The appropriate modeling of the emerging user behavior taking into account the user's perceived quality and its interactions with the overlay and P2P paradigm will finally help to design future Internet applications.

# Nomenclature

## General

$E[X]$	mean value of random variable $X$
$\text{VAR}[X]$	variance of random variable $X$
$\text{STD}[X]$	standard deviation of random variable $X$
$C_X$	coefficient of variation of random variable $X$
$P[X \leq y]$	probability that random variable $X$ is smaller or equal to $y$
$\mathcal{E}(x)$	exponentially distributed random variable with mean $x^{-1}$
$\partial_t f(t, x)$	partial derivative of a function $f$ with respect to variable $t$

The following symbols are unique in individual chapters only.

## Chapter 2

$\Delta t$	maximum time a peer is allowed to download when applying time-based cooperation strategy
$\Delta t_{MIP}$	delay caused by Mobile IP mechanism during which no application data is exchanged
$\Delta t_{VHO}$	delay caused by VHO
$A$	rare chunk availability
$A_i$	availability of chunk $i$ in time interval from $t_0$ to $t_1$
$J$	Jain's fairness index for a given random variable
$N_{max}$	maximum number of parallel uploads of a peer
$T_{on}, T_{off}$	online time and offline time of a peer, respectively
$\gamma$	churn ratio defined as $\gamma = \frac{T_{on}}{T_{off}}$

**Chapter 3**

$\lambda$	arrival of user requests follow Poisson process with rate $\lambda$
$\lambda(t)$	arrival of user requests follow a non-stationary Poisson process with rate $\lambda(t) = \lambda_0 e^{-\alpha t}$
$\Theta$	duration of user impatience with average $\theta^{-1}$
$\Theta_1$	impatience time of a user while downloading with average $\theta_1^{-1}$
$\Theta_2$	impatience time of a user while waiting with average $\theta_2^{-1}$
$f_s$	size of video file with average $E[f_s]$
$R_u, R_d$	access bandwidth of a user in uplink / downlink direction

*Variables used for analysis of OTR server*

$C$	capacity of OTR server
$A$	number of aborting users in fluid model
$D$	number of downloading users in fluid model
$\mathcal{F}$	number of successfully finished downloads in fluid model
$W$	number of waiting users in fluid model
$\underline{N}$	maximum number of customers to be managed by OTR server
$n^*$	maximum number of users simultaneously served by OTR
$N^*$	changeover point $N^*$ reflects whether the user's access bandwidth or the server's capacity is limiting, i.e. $N^* = \lceil \frac{C}{R} \rceil$
$\pi(K)$	stationary probability that $K$ customers are in system
$R(y   K, 0, N)$	conditional probability that remaining sojourn time distribution of a customer when waiting in position $N$ is larger than $y$ in a system counting $K \in \{n^*, \dots, \underline{N} - 1\}$ customers in system, i.e. $P[W(K, 0, N) > y]$
$R(y   K, 1)$	conditional probability that remaining sojourn time distribution of a customer in service is larger than $y$ in a system counting $K \in \{n^*, \dots, \underline{N} - 1\}$ customers in system, i.e. $P[W(K, 1) > y]$
$W$	total sojourn time of a customer



$W(K, 0, N)$	remaining sojourn time of observed customer to complete file download when waiting in position $N$ while there are $K \in \{n^*, \dots, \underline{N} - 1\}$ customers in system
$W(K, 1)$	remaining sojourn time of observed customer needed to complete file download while the observed customer is downloading and there are $K \in \{n^*, \dots, \underline{N} - 1\}$ customers in system

*Variables used for analysis of P2P System*

$\eta$	departure rate of seeders, leechers, and aborting peers
$v$	arrival rate of peers sharing the file which they obtained from another source than the P2P network
$K$	number of fake peers
$A$	number of peers having aborted their download
$D$	number of downloading peers
$D_i$	number of downloading peers which have already downloaded $i$ correct blocks
$F_i$	number of downloading peers which have already downloaded $i$ blocks with at least one corrupted block
$I$	number of idle peers
$L$	number of leeching peers that do not share the file
$S$	number of sharing peers
$S_0$	number of initial sharing peers
$p_a$	probability that a peer aborts the file request after having downloaded a corrupted chunk
$p_b$	probability that a peer downloads the next block from a sharing peer, i.e., with probability $1 - p_b$ the peer downloads from a fake peer offering a corrupted chunk
$p_s$	probability that a peer shares the file after successfully finishing the downloaded of the file

**Chapter 4**

$A, B$	voice data is transmitted from machine $A$ to machine $B$
$D$	machine on which the network emulation tool is running
$\alpha, \beta, \gamma$	parameters of the exponential mapping function $f$ from QoS to QoE $f(x) = \alpha \cdot e^{-\beta x} + \gamma$
$\mu_d$	preset average one-way delay between two nodes
$\sigma_d$	preset standard deviation of the one-way delay between two nodes
$r_d$	preset autocorrelation parameter for one-way delays
$d_0$	minimal physical transmission time
$s$	stream of packets stemming from application under investigation
$s_{in}$	set of packets received by $B$ within a voice stream $s$ sent by $A$
$s_{out}$	set of packets sent from $A$ to $B$ within a voice stream $s$
$d_p$	one-way delay for a packet $p$ is $d_p = t_{r,p} - t_{s,p}$
$t_{r,p}$	time when receiving the last bit of a packet $p$
$t_{s,p}$	time when sending the first bit of a packet $p$
$\sigma_{IPDV}$	jitter expressed as inter-packet delay variation
$\omega$	jitter expressed as standard deviation of the measured one-way delays
$\rho_{s_{in}}$	Type-P-Reordered-Ratio of a stream $s_{in}$ of incoming packets, abbreviated as $\rho$ if clear
$\tau$	mean reordering late time of a packet stream
$\tilde{p}_L$	measured packet loss ratio
$p_L$	preset packet loss probability passed to network emulation tool
$r_L$	preset autocorrelation parameter for loss
$E^2$	mean squared error between model and measured values
$R$	coefficient of correlation as goodness-of-fit measure
$R^2$	coefficient of determination as goodness-of-fit measure
$\kappa$	replication degree of voice datagrams

$m_{rcvd}$	mean goodput at the receiver over captures per observation period
$m_{sent}$	mean throughput at the sender over captures per observation period
$s_{rcvd}$	standard deviation of the goodput at the receiver over captures per observation period
$s_{sent}$	standard deviation of the throughput at the sender over captures per observation period



# List of Acronyms

AMM	abstract mobility model
AR(1)	first-order autoregressive process
B3G	Beyond Third Generation Networks
C/S	client/server
CCDF	complementary cumulative distribution function
CDF	cumulative distribution function
CDN	content distribution network
CycPriM	cyclic priority masking
DCH	dedicated channel
DES	differential equation system
DHT	distributed hash table
DSL	Digital Subscriber Line
DU	download unit
FCFS	first-come-first-serve
FR	full reference
FTTH	fiber-to-the-home
GPRS	General Packet Radio Service
HTTP	Hypertext Transfer Protocol
ICH	intelligent corruption handling
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
iLBC	internet Low Bit Rate Codec

## *Nomenclature*

---

IP	Internet Protocol
IPTV	Internet Protocol-based television
IQX	Interdependency between QoE and QoS is eXponential
iSAC	internet Speech Audio Codec
ISP	Internet service provider
ITU	International Telecommunication Union
LSF	least-shared first
MMM	Manhattan mobility model
MOS	mean opinion score
MSD	multi-source download
NR	no reference
NVR	network-based video recorder
ODE	ordinary differential equation
OTR	OnlineTVRecorder
P2P	peer-to-peer
P2PTV	P2P applications designed to distribute video streams
PCMCIA	Personal Computer Memory Card International Association
PDF	probability density function
PESQ	Perceptual Evaluation of Speech Quality
PIAT	packet interarrival times at the receiver
PIST	packet interarrival times at the sender, i.e. inter-packet transmission times
PIT	packet interarrival times
PU	parallel uploads
QoE	Quality of Experience
QoS	Quality of Service
RDMM	random direction mobility model
RR	reduced reference
SIR	Susceptible-Infected-Recovered

TBC	time-based cooperation strategy
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
UMTS	Universal Mobile Telecommunications System
VBC	volume-based cooperation strategy
VHO	vertical handover
VoD	Video-on-Demand
WLAN	Wireless Local Area Network





---

## Bibliography of the Author

---

— Journals and Book Chapters —

- [1] A. Binzenhöfer and T. Hoßfeld, “Warum Panini Fußballalben auch Informatikern Spaß machen”, in *Fußball eine Wissenschaft für sich* (H.-G. Weigand, ed.), Verlag Königshausen & Neumann, 2006.
- [2] T. Hoßfeld, K. Leibnitz, and M.-A. Remiche, “Modeling of an Online TV Recording System”, *ACM SIGMETRICS Performance Evaluation Review*, Vol. 35, No. 2, 2007.
- [3] A. Binzenhöfer, T. Hoßfeld, G. Kunzmann, and K. Eger, “Efficient Simulation of Large-Scale P2P Networks”, *International Journal of Computational Science and Engineering (IJCSE): Special Issue on Parallel, Distributed and Network-Based Processing*, 2008.
- [4] T. Hoßfeld and A. Binzenhöfer, “Analysis of Skype VoIP Traffic in UMTS: End-to-End QoS and QoE Measurements”, *Computer Networks*, Vol. 52, No. 3, 2008, <http://dx.doi.org/10.1016/j.comnet.2007.10.008>.
- [5] T. Hoßfeld, M. Duelli, D. Staehle, and P. Tran-Gia, “Cooperation Strategies for P2P Content Distribution in Cellular Mobile Networks: Considering Mobility and Heterogeneity”, in *Mobile Peer-to-Peer Computing for Next Generation Distributed Environments: Advancing Conceptual and Algorithmic Applications* (B.-C. Seet, ed.), IGI Global, 2008.

- [6] T. Hoßfeld, D. Schlosser, K. Tutschku, and P. Tran-Gia, “Cooperation Strategies for P2P Content Distribution in Cellular Mobile Networks: Considering Selfishness and Heterogeneity”, in *Mobile Peer-to-Peer Computing for Next Generation Distributed Environments: Advancing Conceptual and Algorithmic Applications* (B.-C. Seet, ed.), IGI Global, 2008.
- [7] K. Tutschku, A. Berl, T. Hoßfeld, and H. de Meer, “Mobile P2P in Cellular Networks: Architecture and Performance”, in *Mobile Peer-to-Peer Computing for Next Generation Distributed Environments: Advancing Conceptual and Algorithmic Applications* (B.-C. Seet, ed.), IGI Global, 2008.
- [8] T. Hoßfeld, D. Hausheer, F. Hecht, F. Lehrieder, S. Oechsner, I. Papafili, P. Racz, S. Soursos, D. Staehle, G. D. Stamoulis, P. Tran-Gia, and B. Stiller, “An Economic Traffic Management Approach to Enable the TripleWin for Users, ISPs, and Overlay Providers”, in *FIA Prague Book*, 2009.
- [9] D. Schlosser and T. Hoßfeld, “Mastering Selfishness and Heterogeneity in Mobile P2P Content Distribution Networks with Multiple Source Download in Cellular Networks”, *Peer-to-Peer Networking and Applications, Special Issue on Mobile P2P Networking and Computing*, 2009.

— Conference Papers —

- [10] T. Hoßfeld, K. Leibnitz, R. Pries, K. Tutschku, P. Tran-Gia, and K. Pawlikowski, “Information Diffusion in eDonkey Filesharing Networks”, in *Proceedings of the Australian Telecommunication Networks and Applications Conference (ATNAC’04)*, Sydney, Australia, 2004.
- [11] T. Hoßfeld, A. Mäder, K. Tutschku, P. Tran-Gia, F.-U. Andersen, H. de Meer, and I. Dedinski, “Comparison of Crawling Strategies for an Optimized Mobile P2P Architecture”, in *Proceedings of the 19th International Teletraffic Congress (ITC’19)*, Beijing, China, 2005.

- 
- [12] T. Hoßfeld and D. Staehle, “A Hybrid Model of the UMTS Downlink Capacity with WWW Traffic on Dedicated Channels”, in *Mobile and Wireless Systems, LNCS 3427*, Dagstuhl, Germany, 2005.
- [13] T. Hoßfeld, K. Tutschku, and F.-U. Andersen, “Mapping of File-Sharing onto Mobile Environments: Enhancement by UMTS”, in *Proceedings of the 3rd Annual IEEE International Conference on Pervasive Computing and Communications Workshops: Mobile Peer-to-Peer Computing (MP2P’05)*, Kauai Island, HI, USA, 2005.
- [14] T. Hoßfeld, K. Tutschku, and F.-U. Andersen, “Mapping of File-Sharing onto Mobile Environments: Feasibility and Performance of eDonkey with GPRS”, in *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC’05)*, New Orleans, LA, USA, 2005.
- [15] T. Hoßfeld, K. Tutschku, F.-U. Andersen, H. de Meer, and J. Oberender, “Simulative Performance Evaluation of a Mobile Peer-to-Peer File-Sharing System”, in *Proceedings of the 1st Conference on Next Generation Internet Networks (NGI’05)*, Rome, Italy, 2005.
- [16] T. Hoßfeld, K. Tutschku, and D. Schlosser, “Influence of the Size of Swapping Entities in Mobile P2P File-Sharing Networks”, in *Proceedings of the GI/ITG-Workshop in conjunction with KiVS 2005: Peer-to-Peer-Systeme und -Anwendungen*, Kaiserslautern, Germany, 2005.
- [17] J. Oberender, F.-U. Andersen, H. de Meer, I. Dedinski, T. Hoßfeld, C. Kappler, A. Mäder, and K. Tutschku, “Enabling Mobile Peer-to-Peer Networking”, in *Mobile and Wireless Systems, LNCS 3427*, Dagstuhl, Germany, 2005.
- [18] F.-U. Andersen, K. Tutschku, T. Hoßfeld, and S. Oechsner, “Verlässliche Peer-to-Peer Technologie als Steuerungsverfahren für zukünftige mobile Zugangsnetze”, in *Proceedings of the 11. ITG-Mobilfunktagung, Technologien und Anwendungen*, Osnabrück, Germany, 2006.

- [19] T. Hoßfeld, A. Binzenhöfer, M. Fiedler, and K. Tutschku, “Measurement and Analysis of Skype VoIP Traffic in 3G UMTS Systems”, in *Proceedings of the 4th International Workshop on Internet Performance, Simulation, Monitoring and Measurement (IPS-MoMe’06)*, Salzburg, Austria, 2006.
- [20] T. Hoßfeld, A. Mäder, and D. Staehle, “When Do We Need Rate Control for Dedicated Channels in UMTS?”, in *Proceedings of the 2006 IEEE 63rd Vehicular Technology Conference (VTC’06)*, Melbourne, Australia, 2006.
- [21] T. Hoßfeld, A. Mäder, K. Tutschku, and F.-U. Andersen, “Time-Discrete Analysis of the Crawling Strategy in an Optimized Mobile P2P Architecture”, in *Proceedings of the EuroNGI Workshop on "Wireless and Mobility" and "New Trends in Network Architectures and Services"*, LNCS 3883, Lovenno di Menaggio, Como, Italy, 2006.
- [22] T. Hoßfeld, S. Oechsner, K. Tutschku, and F.-U. Andersen, “Evaluation of a Pastry-based P2P Overlay for Supporting Vertical Handover”, in *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC’06)*, Las Vegas, NV, USA, 2006.
- [23] T. Hoßfeld, S. Oechsner, K. Tutschku, F.-U. Andersen, and L. Caviglione, “Supporting Vertical Handover by Using a Pastry Peer-to-Peer Overlay Network”, in *Proceedings of the 4th Annual IEEE International Conference on Pervasive Computing and Communications Workshops: Mobile Peer-to-Peer Computing (MP2P’06)*, Pisa, Italy, 2006.
- [24] K. Leibnitz, T. Hoßfeld, N. Wakamiya, and M. Murata, “Modeling of Epidemic Diffusion in Peer-to-Peer File-Sharing Networks”, in *Proceedings of the 2nd International Workshop on Biologically Inspired Approaches to Advanced Information Technology (Bio-ADIT’06)*, LNCS 3853, Senri Life Science Center, Osaka, Japan, 2006.

- 
- [25] K. Leibnitz, T. Hoßfeld, N. Wakamiya, and M. Murata, “On Pollution in eDonkey-like Peer-to-Peer File-Sharing Networks”, in *Proceedings of the 13th GI/ITG Conference on Measurement, Modeling, and Evaluation of Computer and Communication Systems (MMB’06) (MMB’06)*, Nürnberg, Germany, 2006.
- [26] A. Mäder, B. Wagner, T. Hoßfeld, D. Staehle, and H. Barth, “Measurements in a Laboratory UMTS Network with time-varying Loads and different Admission Control Strategies”, in *Proceedings of the 4th International Workshop on Internet Performance, Simulation, Monitoring and Measurement (IPS-MoMe’06)*, Salzburg, Austria, 2006.
- [27] S. Oechsner, T. Hoßfeld, K. Tutschku, and F.-U. Andersen, “Supporting Vertical Handover by a Self-Organizing Multi-Dimensional P2P Overlay”, in *Proceedings of the 2006 IEEE 63rd Vehicular Technology Conference (VTC’06)*, Melbourne, Australia, 2006.
- [28] S. Oechsner, T. Hoßfeld, K. Tutschku, F.-U. Andersen, and L. Caviglione, “Using Kademia for the Configuration of B3G Radio Access Nodes”, in *Proceedings of the 4th Annual IEEE International Conference on Pervasive Computing and Communications Workshops: Mobile Peer-to-Peer Computing (MP2P’06)*, Pisa, Italy, 2006.
- [29] R. Pries, T. Hoßfeld, and P. Tran-Gia, “On the Suitability of the Short Message Service for Emergency Warning Systems”, in *Proceedings of the 2006 IEEE 63rd Vehicular Technology Conference (VTC’06)*, Melbourne, Australia, 2006.
- [30] D. Schlosser, T. Hoßfeld, and K. Tutschku, “Comparison of Robust Cooperation Strategies for P2P Content Distribution Networks with Multiple Source Download”, in *Proceedings of the 6th IEEE International Conference on Peer-to-Peer Computing (P2P’06)*, Cambridge, UK, 2006.

- [31] A. Binzenhöfer, T. Hoßfeld, G. Kunzmann, and K. Eger, “Efficient Simulation of Large-Scale P2P Networks: Compact Data Structures”, in *Proceedings of the Workshop on Modeling, Simulation and Optimization of Peer-to-Peer Environments (MSOP2P’07)*, Naples, Italy, 2007.
- [32] M. Duelli, T. Hoßfeld, and D. Staehle, “Impact of Vertical Handovers on Cooperative Content Distribution Systems”, in *Proceedings of the 7th IEEE International Conference on Peer-to-Peer Computing (P2P’07)*, Galway, Ireland, 2007.
- [33] K. Eger, T. Hoßfeld, A. Binzenhöfer, , and G. Kunzmann, “Efficient Simulation of Large-Scale P2P Networks: Packet-level vs. Flow-level Simulations”, in *Proceedings of the 2nd Workshop on the Use of P2P, GRID and Agents for the Development of Content Networks (UPGRADE-CN’07)*, Monterey Bay, California, USA, 2007.
- [34] T. Hoßfeld and K. Leibnitz, “Modeling and Evaluation of an Online TV Recording Service”, in *Proceedings of the 9th Annual Workshop on Mathematical Performance Modeling and Analysis (MAMA’07)*, in conjunction with *ACM SIGMETRICS’07 and FCRC’07*, San Diego, CA, USA, 2007.
- [35] T. Hoßfeld and K. Leibnitz, “Performance Evaluation of an IPTV Recording Service”, in *Invited Paper at the Jointly-organized Symposium of the Institute of Intelligent Information and Communications Technology (IICT) at Konan University and the ORSJ (Operations Research Society of Japan)*, Kobe, Japan, 2007.
- [36] T. Hoßfeld, P. Tran-Gia, and M. Fiedler, “Quantification of Quality of Experience for Edge-Based Applications”, in *Proceedings of the 20th International Teletraffic Congress (ITC’20)*, Ottawa, Canada, 2007.
- [37] G. Kunzmann, R. Nagel, T. Hoßfeld, A. Binzenhöfer, and K. Eger, “Efficient Simulation of Large-Scale P2P Networks: Modeling Network Transmission Times”, in *Proceedings of the Workshop on Modeling, Simulation*

---

and Optimization of Peer-to-Peer Environments (MSOP2P'07), Naples, Italy, 2007.

- [38] K. Leibnitz, T. Hoßfeld, N. Wakamiya, and M. Murata, “Peer-to-Peer vs. Client/Server: Reliability and Efficiency of a Content Distribution Service”, in *Proceedings of the 20th International Teletraffic Congress (ITC'20)*, Ottawa, Canada, 2007.
- [39] B. Staehle, T. Hoßfeld, M. Kuhnert, , and N. Vicari, “Enabling the Sleep Mode in Non-beaconed 802.15.4 Multihop Networks - A Simulative Investigation”, in *Proceedings of the 6. GI/ITG KuVS Fachgespräch Drahtlose Sensornetze*, Aachen, Germany, 2007.
- [40] J. P. Fernandez-Palacios Gimenez, M. A. Callejo Rodriguez, H. Hasan, T. Hoßfeld, D. Staehle, Z. Despotovic, W. Kellerer, K. Pussep, I. Papafili, G. D. Stamoulis, and B. Stiller, “A New Approach for Managing Traffic of Overlay Applications of the SmoothIT Project”, in *Proceedings of the 2nd International Conference on Autonomous Infrastructure, Management and Security (AIMS'08)*, Bremen, Germany, 2008.
- [41] T. Hoßfeld, D. Hock, P. Tran-Gia, K. Tutschku, and M. Fiedler, “Testing the IQX Hypothesis for Exponential Interdependency between QoS and QoE of Voice Codecs iLBC and G.711”, in *Proceedings of the 18th ITC Specialist Seminar on Quality of Experience (ITCSS'18)*, Karlskrona, Sweden, 2008.
- [42] T. Hoßfeld and K. Leibnitz, “Performance Modeling of Data Aggregation in Wireless Sensor Networks”, in *Invited Paper at the Jointly-organized Symposium of the Institute of Intelligent Information and Communications Technology (IICT) at Konan University and the ORSJ (Operations Research Society of Japan)*, Kobe, Japan, 2008.
- [43] T. Hoßfeld and K. Leibnitz, “A Qualitative Measurement Survey of Popular Internet-based IPTV Systems”, in *Proceedings of the Second Interna-*

- tional Conference on Communications and Electronics (HUT-ICCE'08)*, Hoi An, Vietnam, 2008.
- [44] T. Hoßfeld, K. Leibnitz, and M.-A. Remiche, “Exact Sojourn Time Distribution in an Online IPTV Recording Service”, in *Proceedings of the 15th International Conference on Analytical and Stochastic Modelling Techniques and Applications (ASMTA 2008) (ASMTA'08)*, Nicosia, Cyprus, 2008.
- [45] S. Oechsner, S. Soursos, I. Papafili, T. Hoßfeld, G. D. Stamoulis, B. Stiller, M. A. Callejo, and D. Staehle, “A framework of economic traffic management employing self-organization overlay mechanisms”, in *Proceedings of the 3rd International Workshop on Self-Organizing Systems (IW-SOS'08)*, Vienna, Austria, 2008.
- [46] B. Staehle, T. Hoßfeld, N. Vicari, and M. Kuhnert, “A Cross-Layer Approach for Enabling Low Duty Cycled ZigBee Mesh Sensor Networks”, in *Proceedings of the International Symposium on Wireless Pervasive Computing 2008 (ISWPC'08)*, Santorini, Greece, 2008.
- [47] T. Zinner, T. Hoßfeld, S. Oechsner, and P. Tran-Gia, “On the Trade-off between Efficiency and Congestion in Location-aware Overlay Networks - Example of a Vertical Handover Support System”, in *Proceedings of the 8th IEEE International Conference on Peer-to-Peer Computing (P2P'08)*, Aachen, Germany, 2008.
- [48] S. Oechsner, T. Hoßfeld, and P. Tran-Gia, “Performance Evaluation of a Distributed Lookup System for a Virtual Database Server”, in *Proceedings of the 20th ITC Specialist Seminar Network Virtualization - Concept and Performance Aspects (ITCSS'20)*, Hoi An, Vietnam, 2009.
- [49] D. Schlosser and T. Hoßfeld, “Service Oriented Network Framework Enabling Global QoS and Network Virtualization”, in *Proceedings of the*



---

*20th ITC Specialist Seminar Network Virtualization - Concept and Performance Aspects (ITCSS'20)*, Hoi An, Vietnam, 2009.

---

## General References

---

- [50] D. Y. Barrer, "Queueing with Impatient Customers and Ordered Service", *Operations Research Letters*, Vol. 5, 1957.
- [51] J. R. Dormand and P. J. Prince, "A Family of Embedded Runge-Kutta Formulae", *Journal of Computational and Applied Mathematics*, Vol. 6, 1980.
- [52] F. Baccelli and G. Hebuterne, "On Queues with Impatient Customers", Tech. Rep. 94, INRIA - Centre de Rocquencourt, 1981.
- [53] R. Jain, D.-M. Chiu, and W. Hawe, "A Quantitative Measure Of Fairness And Discrimination For Resource Allocation In Shared Computer Systems", Tech. Rep. DEC-TR-301, Eastern Research Lab, Digital Equipment Corporation, Hudson, MA, USA, 1984, <http://arxiv.org/abs/cs.NI/9809099>.
- [54] E. Coffman, A. Puhalskii, M. Reiman, and P. Wright, "Processor-shared buffers with reneging", *Performance Evaluation*, Vol. 19, No. 1, 1994.
- [55] International Telecommunication Union, "ITU-T Recommendation E.800: Terms and Definitions Related to Quality of Service and Network Performance Including Dependability", 1994.
- [56] M. E. Perkins, K. Evans, D. Pascal, and L. A. Thorpe, "Characterizing the Subjective Performance of the ITU-T 8 kb/s Speech Coding Algorithm-ITU-T G.729", *IEEE Communications Magazine*, Vol. 35, No. 9, 1997.

- [57] L. Rizzo, “Dummysnet: A Simple Approach to the Evaluation of Network Protocols”, *ACM Computer Communication Review*, Vol. 27, No. 1, 1997.
- [58] A. Richards and G. Rogers and M. Antoniadis and V. Witana, “Mapping User Level QoS from a Single Parameter”, in *Proceedings of the 2nd IFIP/IEEE International Conference on Management of Multimedia Networks and Services (MMNS’98)*, 1998.
- [59] International Telecommunication Union, “ITU-T Recommendation G.107: The E-model, a Computational Model for Use in Transmission Planning”, 1998.
- [60] R. B. Sidje, “EXPOKIT. A Software Package for Computing Matrix Exponentials”, *ACM Transactions on Mathematical Software*, Vol. 24, No. 1, 1998.
- [61] G. Almes, S. Kalidindi, and M. Zekauskas, RFC 2679, *A One-way Delay Metric for IPPM*. 1999, <http://www.faqs.org/rfcs/rfc2679.html>.
- [62] A. Brandt and M. Brandt, “On the M(n)/M(n)/s Queue with Impatient Calls”, *Performance Evaluation*, Vol. 35, No. 1-2, 1999.
- [63] A. Bouch, A. Kuchinsky, and N. Bhatti, “Quality is in the Eye of the Beholder: Meeting Users’ Requirements for Internet Quality of Service”, in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI’00)*, 2000.
- [64] International Telecommunication Union, “ITU-T Recommendation P.862: Perceptual Evaluation of Speech Quality (PESQ), an Objective Method for End-to-End Speech Quality Assessment of Narrowband Telephone Networks and Speech Coders”, 2001.
- [65] R. Pastor-Satorras and A. Vespignani, “Epidemic Spreading in Scale-Free Networks”, *Physical Review Letters*, Vol. 86, No. 14, 2001.

- 
- [66] C. Demichelis and P. Chimento, RFC 3393, *IP Packet Delay Variation Metric for IP Performance Metrics (IPPM)*. 2002, <http://www.faqs.org/rfcs/rfc3393.html>.
- [67] Z. J. Haas, J. Y. Halpern, and L. Li, “Gossip-based Ad Hoc Routing”, in *Proceedings of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM’02)*, New York, NY, USA, 2002.
- [68] A. Khelil, C. Becker, J. Tian, and K. Rothermel, “An Epidemic Model for Information Diffusion in MANETs”, in *Proceedings of the 5th ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM’02)*, 2002.
- [69] S. Khirman and P. Henriksen, “Relationship Between Quality-of-Service and Quality-of-Experience for Public Internet Service”, in *Proceedings of the 3rd Passive and Active Measurement Workshop (PAM’02)*, Fort Collins, CO, USA, 2002.
- [70] S. Saroiu, K. P. Gummadi, R. J. Dunn, S. D. Gribble, and H. M. Levy, “An Analysis of Internet Content Delivery Systems”, in *Proceedings of the 5th Symposium on Operating Systems Design and Implementation (USENIX-OSDI’02)*, Boston, MA, USA, 2002.
- [71] B. Cohen, “Incentives Build Robustness in BitTorrent”, Tech. Rep., bittorrent.org, 2003, <http://jmvidal.cse.sc.edu/library/cohen03a.pdf>.
- [72] G. de Veciana and X. Yang, “Fairness, Incentives and Performance in Peer-to-Peer Networks”, in *Proceedings of the 41st Annual Allerton Conference on Communication, Control and Computing*, Monticello, IL, USA, 2003.
- [73] L. Ding and R. A. Goubran, “Speech Quality Prediction in VoIP Using the Extended E-model”, in *Proceedings of the IEEE Global Telecommunications Conference (Globecom’03)*, 2003.

- [74] T. S. Eugene Ng, Y. Chu, S. G. Rao, K. Sripanidkulchai, and H. Zhang, "Measurement-based Optimization Techniques for Bandwidth-Demanding Peer-to-Peer Systems", in *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'03)*, San Francisco, CA, USA, 2003.
- [75] M. Fiedler, K. Tutschku, P. Carlsson, and A. A. Nilsson, "Identification of Performance Degradation in IP Networks Using Throughput Statistics", in *Proceedings of the 18th International Teletraffic Congress (ITC'18)*, Berlin, Germany, 2003.
- [76] N. Gans, G. Koole, and A. Mandelbaum, "Commissioned Paper: Telephone Call Centers: Tutorial, Review, and Research Prospects", *Manufacturing & Service Operations Management*, Vol. 5, No. 2, 2003.
- [77] L. Garces-Erice, E. W. Biersack, P. A. Felber, K. W. Ross, and G. Urvoyy-Keller, "Hierarchical Peer-to-Peer Systems", in *Proceedings of ACM/IFIP International Conference on Parallel and Distributed Computing (Euro-Par 2003)*, Klagenfurt, Austria, 2003.
- [78] International Telecommunication Union, "ITU-T Recommendation G.114: One-Way Transmission Time", 2003.
- [79] International Telecommunication Union, "ITU-T Recommendation P.800.1: Mean Opinion Score (MOS) Terminology", 2003.
- [80] International Telecommunication Union, "ITU-T Recommendation P.862.1: Mapping Function for Transforming P.862 Raw Result Scores to MOS-LQO", 2003.
- [81] T. M. Kusuma and H.-J. Zepernick, "A Reduced-Reference Perceptual Quality Metric for in-Service Image Quality Assessment", in *Proceedings of IEEE Symposium on Trends in Communications (SympoTIC'03)*, 2003.

- 
- [82] K. Lai, M. Feldman, I. Stoica, and J. Chuang, "Incentives for Cooperation in Peer-to-Peer Networks", in *Proceedings of the Workshop on Economics of Peer-to-Peer Systems (P2PECON'03)*, Berkeley, CA, USA, 2003.
- [83] A. P. Markopoulou, F. A. Tobagi, and M. J. Karam, "Assessing the Quality of Voice Communications over Internet Backbones", *IEEE/ACM Transactions on Networking*, Vol. 11, No. 5, 2003.
- [84] H. Masuyama and T. Takine, "Sojourn Time Distribution in a MAP/M/1 Processor-Sharing Queue", *Operations Research Letters*, Vol. 31, No. 5, 2003.
- [85] L. Penserini, L. Liu, J. Mylopoulos, M. Panti, and L. Spalazzi, "Cooperation strategies for agent-based P2P systems", *Web Intelligence and Agent Systems*, Vol. 1, No. 1, 2003.
- [86] P. Triantafillou, C. Xiruhaki, M. Koubarakis, and N. Ntarmos, "Towards High Performance Peer-to-Peer Content and Resource Sharing Systems", in *Proceedings of the First Biennial Conference on Innovative Data Systems Research (CIDR'03)*, Asilomar, CA, USA, 2003.
- [87] K. G. Anagnostakis and M. B. Greenwald, "Exchange-based Incentive Mechanisms for Peer-to-Peer File Sharing", in *Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS'04)*, Tokyo, Japan, 2004.
- [88] S. Andersen, A. Duric, H. Astrom, R. Hagen, W. Kleijn, and J. Linden, RFC 3951, *Internet Low Bit Rate Codec (iLBC)*. 2004, <http://www.faqs.org/rfcs/rfc3951.html>.
- [89] S. Androutsellis-Theotokis and D. Spinellis, "A Survey of Peer-to-Peer Content Distribution Technologies", *ACM Computing Surveys*, Vol. 36, No. 4, 2004.

- [90] J. U. Becker and M. Clement, "The Economic Rationale of Offering Media Files in Peer-to-Peer Networks", in *Proceedings of 37th Hawaii International Conference on System Sciences (HICSS-37)*, Hawaii, HI, US, 2004.
- [91] D. Bergström, "An Analysis of Skype VoIP Application for Use in a Corporate Environment." <http://www.geocities.com/bergstromdennis/>, 2004.
- [92] J. Biström and V. Partanen, "Mobile P2P - Creating a Mobile File-Sharing Environment", Tech. Rep. HUT T-111.590, Research Seminar on Digital Media, Telecommunications Software and Multimedia Laboratory, Helsinki University of Technology, Helsinki, Finland, 2004.
- [93] H. Chen, J. S. Hai Jin, and Z. Han, "Efficient Immunization Algorithm for Peer-to-Peer Networks", in *Proceedings of the 11th International Conference on High Performance Computing (HiPC'04)*, Bangalore, India, 2004.
- [94] P. Eugster, R. Guerraoui, A.-M. Kermarrec, and L. Massoulié, "Epidemic Information Dissemination in Distributed Systems", *IEEE Computer*, Vol. 37, No. 5, 2004.
- [95] P. Felber and E. W. Biersack, "Self-Scaling Networks for Content Distribution", in *Proceedings of the International Workshop on Self-\* Properties in Complex Information Systems*, Berinoro, Italy, 2004.
- [96] M. Feldman, K. Lai, I. Stoica, and J. Chuang, "Robust Incentive Techniques for Peer-to-Peer Networks", in *Proceedings of the 5th ACM conference on Electronic commerce (EC'04)*, New York, NY, USA, 2004.
- [97] F. L. Fessant, S. H. amd A. M. Kermarrec, and L. Massoulié, "Clustering in Peer-to-Peer File Sharing Workloads", in *Proceedings of the 3rd International Workshop on Peer-to-Peer Systems (IPTPS'04)*, San Diego, CA, USA, 2004.

- 
- [98] P. Han, K. Hosanagar, and Y. Tan, “Diffusion of Digital Products in Peer-to-Peer Networks”, in *Proceedings of the 25th International Conference on Information Systems (ISD’08)*, Washington, DC, USA, 2004.
- [99] M. Izal, G. Urvoy-Keller, E. W. Biersack, P. A. Felber, A. Al Hamra, and L. Garces-Erice, “Dissecting BitTorrent: Five Months in a Torrent’s Lifetime”, in *Proceedings of the 5th Passive and Active Measurement Workshop (PAM’04)*, Antibes Juan-les-Pins, France, 2004.
- [100] M. Jelasity, A. Montresor, and O. Babaoglu, “Detection and Removal of Malicious Peers in Gossip-based Protocols”, in *Proceedings of the 2nd Bertinoro Workshop on Future Directions in Distributed Computing: Survivability: Obstacles and Solutions (FuDiCo II: S.O.S.)*, Bertinoro, Italy, 2004.
- [101] C. Lindemann and O. P. Waldhorst, “Exploiting Epidemic Data Dissemination for Consistent Lookup Operations in Mobile Applications”, *SIGMOBILE Mob. Comput. Commun. Rev.*, Vol. 8, No. 3, 2004.
- [102] F. L. Piccolo, G. Neglia, and G. Bianchi, “The Effect of Heterogeneous Link Capacities in BitTorrent-Like File Sharing Systems”, in *Proceedings of the 2004 International Workshop on Hot Topics in Peer-to-Peer Systems (HOT-P2P’04)*, Volendam, The Netherlands, 2004.
- [103] D. Qiu and R. Srikant, “Modeling and Performance Analysis of BitTorrent-like Peer-to-Peer Networks”, in *Proceedings of the ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM’04)*, Portland, OR, USA, 2004.
- [104] K. Tutschku, “A Measurement-based Traffic Profile of the eDonkey File-sharing Service”, in *Proceedings of the 5th Passive and Active Measurement Workshop (PAM’04)*, Antibes Juan-les-Pins, France, 2004.

- [105] K. G. Zerfiridis and H. D. Karatza, "File Distribution Using a Peer-to-Peer Network – A Simulation Study", *Journal of Systems and Software*, Vol. 73, No. 1, 2004.
- [106] K. G. Zerfiridis and H. D. Karatza, "Optimized Dissemination of Highly Anticipated Content over an Itinerary Based P2P Network", in *Proceedings of the 37th Annual Simulation Symposium (ANSS'04)*, Arlington, VA, USA, 2004.
- [107] A. Al Hamra and P. A. Felber, "Design choices for content distribution in P2P networks", *ACM SIGCOMM Computer Communication Review*, Vol. 35, No. 5, 2005.
- [108] T. Berson, "Skype Security Evaluation", Tech. Rep. ALR-2005-031, Anagram Laboratories, 2005.
- [109] G. Carofiglio, R. Gaeta, M. Garetto, P. Giaccone, E. Leonardi, and M. Sereno, "A Statistical Physics Approach for Modelling P2P Systems", *ACM SIGMETRICS Performance Evaluation Review*, Vol. 33, No. 2, 2005.
- [110] N. Christin, A. S. Weigend, and J. Chuang, "Content Availability, Pollution and Poisoning in File Sharing Peer-to-Peer Networks", in *Proceedings of the 6th ACM Conference on Electronic Commerce (EC'05)*, Vancouver, BC, Canada, 2005.
- [111] M. Feldman and J. Chuang, "Overcoming Free-Riding Behavior in Peer-to-Peer Systems", *SIGecom Exch. Journal*, Vol. 5, No. 4, 2005.
- [112] M. Fiedler, S. Chevul, O. Radtke, K. Tutschku, , and A. Binzenhöfer, "The Network Utility Function: A Practicable Concept for Assessing Network Impact on Distributed Services", in *Proceedings of the 19th International Teletraffic Congress (ITC'19)*, Beijing, China, 2005.
- [113] A. Ganesh, L. Massoulie, and D. Towsley, "The effect of network topology on the spread of epidemics", in *Proceedings of the 24th Annual Joint*



---

*Conference of the IEEE Computer and Communications Societies (INFOCOM'05)*, Miami, FL, USA, 2005.

- [114] S. Garfinkel, "VoIP and Skype Security."  
<http://www.skypetips.internetvisitation.org/files/VoIPf>, 2005.
- [115] C. Gkantsidis and P. Rodriguez, "Network coding for large scale content distribution", in *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'05)*, Miami, FL, USA, 2005.
- [116] International Telecommunication Union, "ITU-T Recommendation G.1030: Estimating End-to-End Performance in IP Networks for Data Applications", 2005.
- [117] J. Liang, R. Kumar, Y. Xi, and K. Ross, "Pollution in P2P File Sharing Systems", in *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'05)*, Miami, FL, USA, 2005.
- [118] Nokia, "Quality of Experience (QoE) of Mobile Services: Can it be Measured and Improved?." [http://www.nokia.com/NOKIA\\_COM\\_1/Operators/Downloads/Nokia\\_Services/whitepaper\\_qoe\\_net.pdf](http://www.nokia.com/NOKIA_COM_1/Operators/Downloads/Nokia_Services/whitepaper_qoe_net.pdf), 2005.
- [119] D. Rubenstein and S. Sahu, "Can Unstructured P2P Protocols Survive Flash Crowds?", *IEEE/ACM Transactions on Networking*, Vol. 13, No. 3, 2005.
- [120] B. Sericola, F. Guillemin, and J. Boyer, "Sojourn Times in the M/PH/1 Processor Sharing Queue", *Queueing Systems*, Vol. 50, No. 1, 2005.

- [121] R. Thommes and M. Coates, “Epidemiological Models of Peer-to-Peer Viruses and Pollution”, Tech. Rep., Department of Electrical and Computer Engineering, McGill University, 2005.
- [122] S. Zöls, R. Schollmeier, W. Kellerer, and A. Tarlano, “The Hybrid Chord Protocol: A Peer-to-Peer Lookup Service for Context-Aware Mobile Applications”, in *Proceedings of the 4th International Conference on Networking (ICN’05)*, Reunion Island, France, 2005.
- [123] S. A. Baset and H. Schulzrinne, “An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol”, in *Proceedings of the 25th IEEE International Conference on Computer Communications, Joint Conference of the IEEE Computer and Communications Societies (INFOCOM’06)*, Barcelona, Spain, 2006.
- [124] P. Biondi and F. Desclaux, “Silver Needle in the Skype.” Presentation at the Black Hat Europe 2006, <http://www.blackhat.com/>, 2006.
- [125] K.-T. Chen, C.-Y. Huang, P. Huang, and C.-L. Lei, “Quantifying Skype User Satisfaction”, in *Proceedings of the ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM’06)*, Pisa, Italy, 2006.
- [126] S. Ehlert and S. Petgang, “Analysis and Signature of Skype VoIP Session Traffic”, Tech. Rep. NGNI-SKYPE-06b, Fraunhofer FOKUS, Berlin, Germany, 2006.
- [127] M. Fiedler, K. Tutschku, S. Chevul, L. Isaksson, and A. Binzenhöfer, “The Throughput Utility Function: Assessing Network Impact on Mobile Services”, in *Proceedings of the EuroNGI Workshop on "Wireless and Mobility" and "New Trends in Network Architectures and Services"*, LNCS 3883, 2006.

- 
- [128] P. Garbacki, A. Iosup, D. Epema, and M. van Steen, “2Fast: Collaborative Downloads in P2P Networks”, in *Proceedings of the Sixth IEEE International Conference on Peer-to-Peer Computing (P2P’06)*, Cambridge, UK, 2006.
- [129] H. C. Gromoll, P. Robert, B. Zwart, and R. Bakker, “The Impact of Reneging in Processor Sharing Queues”, in *Proceedings of the Joint International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS’06/Performance’06)*, 2006.
- [130] S. Guha, N. Daswani, and R. Jain, “An Experimental Study of the Skype Peer-to-Peer VoIP System”, in *Proceedings of the 5th International Workshop on Peer-to-Peer Systems (IPTPS’06)*, 2006.
- [131] D. Hales, “Emergent Group Level Selection in a Peer-to-Peer Network”, *ComPlexUs*, Vol. 3, No. 1-3, 2006.
- [132] R. Kwitt, T. Fichtel, and T. Pfeiffenberger, “Measuring Perceptual VoIP Speech Quality over UMTS”, in *Proceedings of the 4th International Workshop on Internet Performance, Simulation, Monitoring and Measurement (IPS-MoMe’06)*, Salzburg, Austria, 2006.
- [133] J. K. Lee and J. C. Hou, “Modeling Steady-State and Transient Behaviors of User Mobility: Formulation, Analysis, and Application”, in *Proceedings of the Seventh ACM International Symposium on Mobile ad hoc networking and computing (MobiHoc’06)*, New York, NY, USA, 2006.
- [134] A. Legout, G. Urvoy-Keller, and P. Michiardi, “Rarest first and choke algorithms are enough”, in *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement (IMC’06)*, Rio de Janeiro, Brazil, 2006.
- [135] W.-C. Liao, F. Papadopoulos, and K. Psounis, “A Peer-to-Peer Cooperation Enhancement Scheme and its Performance Analysis”, *Journal of Communications (JCM)*, Vol. 1, No. 7, 2006.

- [136] A. Morton, L. Ciavattoni, G. Ramachandran, S. Shalunov, and J. Perser, RFC 4737, *Packet Reordering Metrics*. 2006, <http://www.faqs.org/rfcs/rfc4737.html>.
- [137] T. Moscibroda, S. Schmid, and R. Wattenhofer, “On the Topologies Formed by Selfish Peers”, in *Proceedings of the 25th Annual ACM Symposium on Principles of Distributed Computing (PODC’06)*, Denver, CO, USA, 2006.
- [138] I. Norros, B. J. Prabhu, and H. Reittu, “Flash Crowd in a File Sharing System Based on Random Encounters”, in *Proceedings of the 2006 Workshop on Interdisciplinary Systems Approach in Performance Evaluation and Design of Computer & Communications Systems (Interperf’06)*, Pisa, Italy, 2006.
- [139] R. R. Pastrana-Vidal and J.-C. Gicquel, “Automatic Quality Assessment of Video Fluidity Impairments Using a No-Reference Metric”, in *Proceedings of 4th International Workshop on Video Processing and Quality Metrics for Consumer Electronics (VPQM’06)*, 2006.
- [140] J. Risson and T. Moors, “Cooperation strategies for agent-based P2P systems”, *Computer Networks*, Vol. 50, No. 17, 2006.
- [141] Signalogic, “Speech Codec Wav Samples.” <http://www.signalogic.com/melp/EngSamples/Orig/male.wav>, 2006.
- [142] G. I. Sound, “GIPS iSAC.” <http://www.globalipsound.com/datasheets/iSAC.pdf>, 2006.
- [143] K. Suh, D. R. Figueiredo, J. Kurose, and D. Towsley, “Characterizing and detecting relayed traffic: A case study using Skype”, in *Proceedings of the 25th IEEE International Conference on Computer Communications, Joint Conference of the IEEE Computer and Communications Societies (INFOCOM’06)*, Barcelona, Spain, 2006.

- 
- [144] Texas Instruments, “OMAP3430.” <http://focus.ti.com/>, 2006.
- [145] A. Binzenhöfer, D. Schlosser, K. Tutschku, and M. Fiedler, “An Automatic Approach to Verify End-to-End Communication Quality”, in *Proceedings of the 10th IFIP-IEEE International Symposium on Integrated Network Management (IM’07)*, 2007.
- [146] A. Bostan, F. Chyzak, F. Ollivier, B. Salvy, E. Schost, and A. Sedoglavic, “Fast Computation of Power Series Solutions of Systems of Differential Equations”, in *Proceedings of the 18th Annual ACM-SIAM symposium on Discrete algorithms (SODA’07)*, 2007.
- [147] X. Cheng, C. Dale, and J. Liu, “Understanding the Characteristics of Internet Short Video Sharing: YouTube as a Case Study”, Tech. Rep. abs/0707.3670, Cornell University, 2007, arXiv:0707.3670v1.
- [148] M. Duelli, T. Hoßfeld, and D. Staehle, “Impact of Vertical Handovers on Cooperative Content Distribution Systems”, Tech. Rep. 428, University of Würzburg, 2007.
- [149] U. Engelke and H.-J. Zepernick, “Perceptual-based Quality Metrics for Image and Video Services: A Survey”, in *Proceedings of the 3rd Conference on Next Generation Internet Networks (NGI’07)*, Trondheim, Norway, 2007.
- [150] P. Gill, M. Arlitt, Z. Li, and A. Mahanti, “YouTube Traffic Characterization: A View from the Edge”, in *Proceedings of the ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM’07)*, Kyoto, Japan, 2007.
- [151] C. M. Huang, T. H. Hsu, and M. F. Hsu, “Network-Aware P2P File Sharing over the Wireless Mobile Networks”, *IEEE Journal on Selected Areas in Communications*, Vol. 25, No. 1, 2007.

- [152] P. Michiardi and G. Urvoy-Keller, "Performance Analysis of Cooperative Content Distribution in Wireless Ad Hoc Networks", in *Proceedings of the 4th Annual Conference on Wireless on Demand Network Systems and Services (WONS'07)*, Obergurgl, Austria, 2007.
- [153] T. Hossfeld, D. Hock, P. Tran-Gia, K. Tutschku, and M. Fiedler, "Testing the IQX Hypothesis for Exponential Interdependency between QoS and QoE of Voice Codecs iLBC and G.711", Tech. Rep. 442, University of Würzburg, 2008.
- [154] A. Birolini, *Qualität und Zuverlässigkeit technischer Systeme: Theorie, Praxis, Management*. Springer-Verlag GmbH, 1991.
- [155] B. Gnedenko and D. König, *Handbuch der Bedienungstheorie II*. Berlin: Akademie-Verlag, (first edition. in russian, nauka, 1966) ed., 1984.
- [156] J. Murray, *Mathematical Biology, I: An Introduction*. Springer, 3 ed., 2002.
- [157] J. Nielsen, *Usability Engineering*. Morgan Kaufman, 1994.
- [158] D. Soldani, M. Li, and e. R. Cuny, *QoS and QoE Management in UMTS Cellular Systems*. Wiley, 2006.

ISSN 1432-8801