Julius-Maximilians-Universität Würzburg
Institut für Informatik
Lehrstuhl für Kommunikationsnetze
Prof. Dr.-Ing. Phuoc Tran-Gia

Performance Evaluation and Optimization of Content Distribution using Overlay Networks

Frank Lehrieder

Würzburger Beiträge zur
Leistungsbewertung Verteilter Systeme

Bericht 1/13

Würzburger Beiträge zur Leistungsbewertung Verteilter Systeme

Herausgeber

Prof. Dr.-Ing. Phuoc Tran-Gia
Universität Würzburg
Institut für Informatik
Lehrstuhl für Kommunikationsnetze
Am Hubland
D-97074 Würzburg

Tel.: +49-931-31-86630
Fax.: +49-931-31-86632
email: trangia@informatik.uni-wuerzburg.de

Satz

Reproduktionsfähige Vorlage vom Autor.
Gesetzt in LaTeX Computer Modern 9pt.

ISSN 1432-8801

Performance Evaluation and Optimization of Content Distribution using Overlay Networks

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

vorgelegt von

Frank Lehrieder

aus

Würzburg

Würzburg 2013

Eingereicht am: 22.10.2012

bei der Fakultät für Mathematik und Informatik

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

2. Gutachter: Prof. Dr.-Ing. Wolfgang Kellerer

3. Gutachter: Prof. Dr.-Ing. Thomas Bauschert

Tag der mündlichen Prüfung: 06.02.2013

Danksagung

Bei der Erstellung der vorliegenden Arbeit habe ich viel Unterstützung und Hilfe erfahren. All denen, die zum guten Gelingen beigetragen haben, möchte ich hier meinen Dank aussprechen.

Zu allererst danke ich meinem Doktorvater Prof. Phuoc Tran-Gia, der es mir ermöglicht hat, an seinem Lehrstuhl zu promovieren. Nicht zuletzt durch seinen persönlichen Einsatz, seine vielfältigen internationalen Kontakte und durch seine Mitarbeiterführung sorgte er für ein herausragendes Umfeld, in dem ich die Forschungsarbeiten für meine Doktorarbeit durchführen konnte. Zusätzlich bot er mir Gelegenheit, an mehreren internationalen Forschungsprojekten teilzunehmen und hochrangige Fachkonferenzen auf der ganzen Welt zu besuchen, wo ich meine Ideen mit anderen Forschern diskutieren konnte. Rückblickend war dies eine der wesentlichen Grundlagen für die Ergebnisse, die ich in meiner Forschung erzielen konnte. Darüber hinaus gab er mir die Möglichkeit, immer wieder eigene Ideen zu entwickeln und in Projekten auch selbst Verantwortung zu übernehmen. Diese außerordentlichen Rahmenbedingungen haben entscheidend zum Gelingen meiner Arbeit beigetragen und dafür bin ich Prof. Tran-Gia sehr dankbar.

Bedanken möchte ich mich auch bei Prof. Wolfgang Kellerer, der in dreierlei Hinsicht zu dieser Arbeit beigetragen hat. Die in Zusammenarbeit mit ihm durchgeführten Forschungsarbeiten stellen einen Teil der vorliegenden Doktorarbeit dar. Weiterhin übernahm er die Begutachtung der Arbeit und fungierte als Prüfer bei meiner Disputation. Für diese Unterstützung danke ich ihm herzlich. Weiterhin gebührt mein Dank Prof. Thomas Bauschert für die Erstellung eines zusätzlichen Gutachtens und Prof. Rainer Kolla, der den Vorsitz der Prüfungskommission übernahm.

Dankbar bin ich auch Prof. Michael Menth, Dr. Tobias Hoßfeld und György Dán. In zahlreichen gemeinsamen Veröffentlichungen konnte ich viel von ihnen über das wissenschaftliche Arbeiten lernen und von ihren Erfahrungen profitieren. Insbesondere durch die gute Betreuung meiner Diplomarbeit ermöglichte mir Prof. Menth einen erfolgreichen Start in meine Arbeit als wissenschaftlicher Mitarbeiter am Lehrstuhl.

Dass ich mich am Lehrstuhl immer sehr wohl gefühlt habe, habe ich vor allem meinen jetzigen und ehemaligen Kollegen zu verdanken. In vielfältiger Weise haben sie zum erfolgreichen Abschluss dieser Arbeit beigetragen, sei es in fachlichen Diskussionen, durch wertvolle Kommentare zu wissenschaftlichen Arbeiten oder auch durch technische Unterstützung. Ohne alle einzeln beim Namen zu nennen möchte ich mich herzlich bei ihnen für die angenehme Atmosphäre am Lehrstuhl und die gute Zusammenarbeit bedanken. Zusätzlich haben sich viele Freundschaften entwickelt und ich denke gerne an die zahlreichen Unternehmungen zurück, die ich mit ihnen auch außerhalb des Lehrstuhls durchgeführt habe.

Bei meiner Tätigkeit am Lehrstuhl habe ich im Rahmen von Übungen und Abschlussarbeiten eine große Zahl von Studenten betreut. Mit ihrem Engagement und ihren Ideen haben sie dafür gesorgt, dass ich an der Betreuung nicht nur viel Freude hatte sondern von ihr auch fachlich stark profitieren konnte. Für die tatkräftige Hilfe in allen administrativen Tätigkeiten, insbesondere bei der Anmeldung und Abrechnung der vielen Dienstreisen, bin ich Frau Förster und Frau Wichmann sehr dankbar.

Abschließend möchte ich mich herzlich bei meiner Familie und meiner Partnerin Sonja bedanken. Ohne den Rückhalt und die vielfältige Unterstützung wäre diese Doktorarbeit nicht möglich gewesen.

Contents

1 Introduction

Content distribution is one of the most important services of the Internet. Its applications range from pure file-sharing to video-on-demand and live streaming or Internet protocol television (IPTV). The rising popularity of these applications, and their increasing demand for transmission capacity [87], e.g., due to high-definition video content, put a high load on the network infrastructure of the Internet.

As a consequence, the problem arises how these demands can efficiently be handled. Three main stakeholders are involved in this process, each with different notions of an efficient distribution process. Internet users expect fast and reliable access to the desired content. Content providers have similar objectives since the consumers of the content, i.e., the users, are their direct customers. However, they rather try to limit their capital and operational expenditures such as costs for content servers and bandwidth. Finally, Internet service providers (ISPs) are responsible for delivering the content to end users. Their aim is to provide high-speed and resilient network access, but at the same time network load and interconnection cost to other ISPs should be kept small.

A promising and widely used solution for efficient content distribution are overlay networks based on the peer-to-peer (P2P) paradigm. This method saves costs for content providers since users support the distribution process by uploading pieces of the already downloaded content to other users. This reduces load on content servers considerably or renders such servers unnecessary. In recent years, BitTorrent networks have been the most popular overlays for that purpose and have been responsible for a large share of the total Internet traffic [71]. Therefore, in this monograph we investigate traffic optimization techniques for overlay networks on the example of BitTorrent, which is widely used for file-sharing and serves also as a basis for video transmission overlays.

Today's content distribution overlays and in particular BitTorrent suffer from the problem that they are underlay-agnostic, i.e., they are not aware of the physical network infrastructure. In contrast, they build a logical *overlay* network on top of it. The routing of traffic demands is determined within the overlay network and ignores the properties of the physical network infrastructure. This leads to an inefficient usage of the physical *underlay* network of the ISPs [51, 54, 66]. For the concrete example of BitTorrent this implies that the BitTorrent software running on the computers of the users (also called peers) determines the other peers for the actual data exchange. Since this decision does not take into account the physical network infrastructure, two peers located in the networks of different ISPs might exchange data although in the network of both ISPs other peers might be present and able to serve the same piece of data. This results in unnecessary inter-ISP traffic and increases inter-connection cost of some ISPs.

This problem is severe in particular for small or medium-sized ISPs since the Internet is a hierarchical network of ISPs. Smaller (lower tier) ISPs typically pay larger (higher tier) ISPs for connecting them to the rest of the Internet. For this inter-connection service the larger ISP charges the smaller one based on the volume of the inter-ISP traffic, i.e., traffic that was forwarded from and to the rest of the Internet. Consequently, smaller ISPs can save inter-connection costs if they manage to reduce their inter-ISP traffic. As a result, traffic optimization techniques for overlay networks are useful for lower tier ISPs to prevent inter-connection costs due to unnecessary inter-ISP traffic. However, performance degradations experienced by the end users, e.g., low transmission speeds, should be avoided since unsatisfied users are likely to switch to competitor ISPs.

Two types of traffic optimization mechanisms are currently discussed and investigated. The first one is called *locality-awareness*. This solution equips the overlay nodes, i.e, peers, with knowledge about the underlying network topology so that they can preferentially exchange data with other peers in the network of the same ISP. Two such locality-awareness mechanisms for BitTorrent networks are *biased neighbor selection* [51] and *biased unchoking* [17]. The efficiency of these solutions consequently depends on the existence of other peers within the same ISP.

The other prominent solution to reduce inter-ISP traffic of content distribution overlays is *caching*. To this end, an ISP provides an additional entity – a *cache* – in its network that stores a copy of popular files. Requests for parts of those files can then be served by the cache and do not need to be downloaded from outside the network of this ISP. The idea is well-known from web traffic [31, 32, 37]. However, overlay networks exhibit significant differences to normal web pages. For example, overlay nodes fetch different parts of the same file from different locations (multi-source download). Therefore, caching algorithms have to be revisited and their performance needs to be evaluated with respect to the changed circumstances.

The objectives of this monograph can be summarized as follows. The first goal is to provide a thorough understanding of the nature of today's overlay networks using the popular example of BitTorrent. This knowledge permits to define scenarios of practical relevance for the performance evaluation of traffic optimization techniques in overlay networks. Furthermore, we use this knowledge to estimate the optimization potential of locality-awareness at Internet scale. An additional objective is to investigate and optimize the performance of caching as a traffic optimization technique by proposing enhancements and new algorithms or giving recommendations for suitable configurations. A detailed description of the scientific contribution in this monograph is given in the following.

1.1 Scientific Contribution

This section summarizes the contribution of this monograph to the field of traffic optimization in P2P-based overlay networks. It gives an overview of the content of the studies and explains their relations.

Figure 1.1 classifies the publications according to the investigated topic on the y-axis and the investigation methodology on the x-axis. The investigated topics consist of caching and locality-awareness, which are the two main approaches to traffic optimization in overlay networks. The methodologies comprise real-world Internet measurements, experiments in controlled environments, simulations and analytical modeling. Some studies cover only a single aspect and methodology while others overlap different areas in both dimensions. This means that a single subject is investigated using different methodologies or a study comprises caching and locality-awareness mechanisms at the same time.

The first major contribution presented in this monograph is a comprehensive characterization of overlay networks which are currently used in the Internet. We study the nature of BitTorrent swarms – the most popular overlay networks in today's Internet – in a large measurement campaign. Based on the measurement results, we model the characteristics of BitTorrent networks by providing statistical distributions, e.g., for the number of users per swarm or the size of the exchanged file. In addition, we investigate how peers are distributed over the different ISPs in the Internet. This knowledge is used to define scenarios for performance evaluation of both locality-awareness and caching in the other studies. This is valuable information to ensure that performance results are obtained in scenarios of practical relevance. Furthermore, we derive the Internet-wide optimization potential of locality-awareness mechanisms based on this data, cf. [5]. This estimation shows that locality-awareness can save a large fraction inter-ISP traffic because most of the BitTorrent peers participate in the distribution of a moderate number of popular files. There, a large number of opportunities exist to save inter-ISP traffic by sharing these file with other peers from the same ISP. Since this study uses both real-world measurements and modeling, it is not only

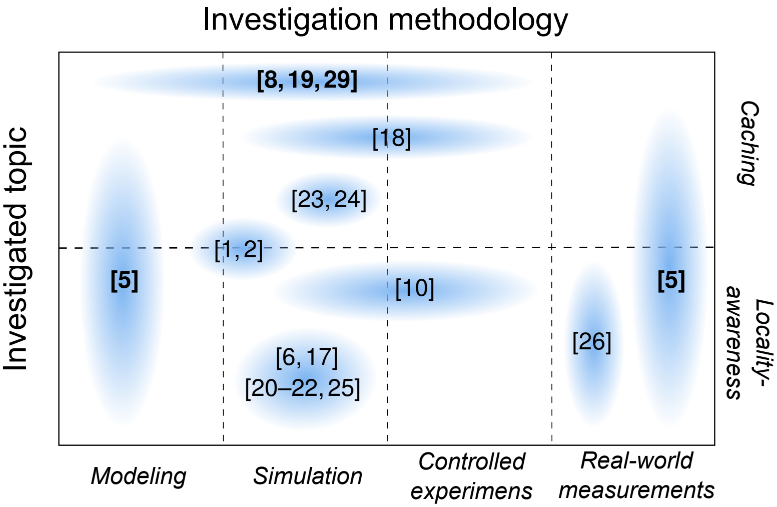


FIGURE 1.1: Cartography of scientific contributions of the author on traffic optimization in overlay networks. The content of references in bold is presented in this monograph.

located in the right part of Figure 1.1, but also in the left one.

The second major contribution presented in this monograph considers caching of overlay traffic as a means to reduce inter-ISP traffic. In a first step, we provide a two-dimensional Markov model of the impact of a cache on a single BitTorrent-based overlay network and validate the model via simulations and experiments with real BitTorrent clients, cf. [8, 19]. The investigation shows that the upload capacity of the cache has a considerable impact on inter-ISP traffic savings and that the same amount of upload capacity can reduce more inter-ISP traffic in some swarms than in others. It is even possible under certain conditions that increasing the upload capacity of the cache leads to more outgoing inter-ISP traffic. This is counter-intuitive, but it can be explained by the fact that peers which are fed by the cache can serve other peers outside their ISP better. Therefore, we argue that the cache upload capacity needs to be actively managed in multi-swarm scenarios to maximize the traffic savings and we consequently propose different allocation policies of the cache upload capacity and evaluate the performance, cf. [23, 24, 29]. Our simulations show that an appropriate allocation policy can reduce inter-ISP traffic savings by up to 50 % more than the demand-driven allocation for the same upload capacity of the cache.

Beyond this monograph, the contribution to the field of traffic optimization in P2P-based overlay networks comprises a number of studies on locality-awareness. In [26], the BitTorrent measurements [5] are combined with the Internet topology obtained from caida.org [92]. This study shows that almost no peers are located in very large (tier-1) ISPs. Furthermore, it considers different implementation options of locality-awareness and studies which ISPs could benefit most from them. The remaining studies investigate the performance of locality-awareness by means of mathematical modeling, simulations, and experiments in controlled environments such as G-Lab [60] or Planet-Lab [58]. In the area of locality-awareness, we propose a new mechanism called *biased unchoking* and evaluated its performance via simulations [6, 17, 20]. The results show that it is a powerful complement to *biased neighbor selection*, which is proposed and evaluated in literature, e.g., [51, 54]. Besides the studies on pure file-sharing networks, video streaming overlays are also investigated since video traffic is expected to dominate the total Internet traffic in the near future [87]. Therefore, a cooperative traffic management approach for these overlays for video streaming is proposed and evaluated in [10].

1.2 Outline of Thesis

The remainder of this monograph is structured as follows. Chapter 2 provides a detailed explanation of the functionality of overlay networks using the popular example BitTorrent. Furthermore, it introduces the problem of inter-ISP traffic caused by overlay networks. Finally, it describes the two prominent solutions for traffic optimization in overlay networks and reviews related work.

In Chapter 3 we study the nature of overlay networks in today's Internet. To this end, we present the results of our large scale measurement project. We investigate for example the number of users per swarm and their distribution in the Internet. This also allows us to derive an estimate of the optimization potential of locality-awareness in today's Internet. Finally, we provide a statistical characterization of BitTorrent networks that can be used as input for performance evaluation of traffic optimization mechanisms.

Chapter 4 studies overlay caches as a means to reduce inter-ISP traffic. We first investigate the impact of caches on the swarm dynamics for the example of BitTorrent. Subsequently, we estimate the amount of inter-ISP traffic that can be saved by caching. Results from simulations and experiments in controlled environments are shown to assess the accuracy of the estimates. Finally, we extend the scenario to multiple swarms and investigate policies for the allocation of the cache upload capacity. Chapter 5 summarizes this work and draws conclusions.

2 Content Distribution Overlays and Traffic Optimization

The inherent drawback of client-server architectures for content distribution is their limited scalability. If a large number of users download data simultaneously from the same server, congestion is likely to occur on the server since its network transmission capacity is shared among all users. As a consequence, the time to complete the download increases [35].

The most popular solution to this problem in today's Internet are overlay networks which work according to the P2P principle. Such networks create a logical structure on top of the physical network topology. The users, who are often called *peers*, help in the data distribution process by offering a part of their upload capacity. Therefore, the available upload capacity increases with the number of peers which exchange a given file, which speeds up the distribution process compared to pure client-server architectures. As a consequence, such networks are widely used in the Internet and responsible for a large fraction of the total Internet traffic [71, 87]. Hence, traffic optimization in overlay networks is important to achieve an efficient distribution process both from the perspective of the ISPs and of the users downloading the content.

This chapter gives background information on P2P-based content distribution and traffic optimization. Furthermore, it reviews related studies and explains how the content of this monograph extends previous work. The chapter is divided into three sections. First, it describes the functionality of BitTorrent-based content distribution overlays in Section 2.1 and presents measurement studies of live BitTorrent swarms. Second, the problem of the large amount of inter-ISP traffic which is generated by such content distribution overlays is introduced in Section 2.2. For that purpose, we give a short overview on inter-domain routing and the ISP hierarchy in the Internet. Finally, the chapter explains mechanisms to decrease the inter-ISP traffic of P2P overlays. Such mechanisms are currently under discussion in the IETF and studied in the research community. They can be divided into the two categories *locality-awareness* and *caching*. Both approaches are presented in Section 2.3 and related studies are reviewed.

2.1 Content Distribution Overlays in the Internet

Although other protocols for content distribution exist, this study focuses on BitTorrent-based networks since they have been the most popular ones in recent years and generated the largest share of Internet traffic [71]. Therefore, this section starts with an overview of the BitTorrent functionality. Afterwards, it presents studies on measurements of live BitTorrent networks and analytical performance models of BitTorrent.

2.1.1 BitTorrent and its Mechanisms

BitTorrent networks form a separate, mesh-based overlay for every file which is exchanged within the network. Such an overlay is called a *swarm* in BitTorrent terminology and comprises all peers that participate in the exchange of a given file. Peers which already have the entire file are called *seeders*. They participate only by uploading it to other peers. Peers which do not yet have the entire file are *leechers*. An overview of the key components and mechanisms of BitTorrent is given in Figure 2.1. In [40] and [50], a more comprehensive description of the BitTorrent protocol can be found. We focus on the most important aspects and on typical configurations in the following.

To facilitate that leechers can contribute their upload capacity to the distribution process of the file, the file is split in smaller parts called *chunks*, which are in turn split into a number of *blocks*. As soon as a leecher has a complete

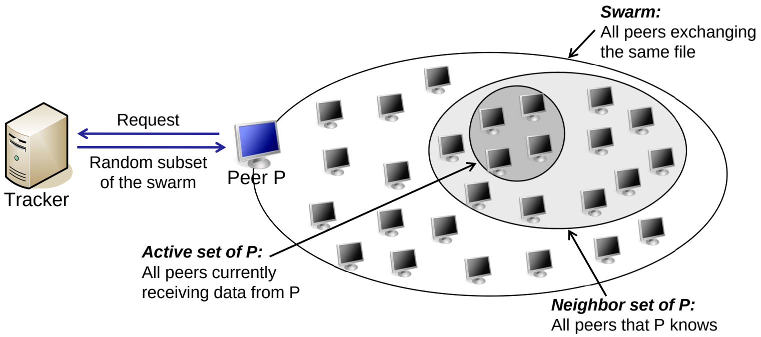

Swarm:
All peers exchanging the same file
Request
Random subset of the swarm
Tracker
Peer P
Active set of P:
All peers currently receiving data from P
Neighbor set of P:
All peers that P knows

FIGURE 2.1: Key components and mechanisms of BitTorrent.

chunk, it can upload it to other leechers. In addition, a leecher can download different parts of the file concurrently from a number of peers. This feature is called *multi-source download*.

If a new peer wants to join an existing swarm, it contacts the *tracker*, a central entity that keeps track of all peers participating in a swarm. The tracker returns a list of active peers in the swarm. Typically, this list contains around 50 to 100 peers, but the concrete number depends on the configuration of the tracker. The new peer uses these addresses to establish contacts to other peers in the swarm. If another peer accepts the contact request then both peers add each other to their *neighbor set* and exchange information about the chunks they already have by sending a *bitfield message*. If peer A has chunks which peer B still needs then peer B is interested in peer A and sends an *interested* message to peer A. As a consequence, every peer in the swarm knows which neighbors want to download data from it.

To decide which neighbor actually receives data, the peers employ the *choke* algorithm. This algorithm determines the *active set* of a peer, which contains all peers that are currently receiving data from this peer. The choke algorithm has different modes for leechers and seeders. In leecher mode, i.e., when the peer itself wants to download from others, the peer uploads to those k interested peers from which it receives data at the highest download rate. Typically, k has small values, for example 3 or 4. The selection process is repeated every 10 seconds in the default configuration. In BitTorrent terminology, the peers in the active set are called *unchoked* peers whereas the rest of the neighbors is *choked* by the peers. This strategy is called tit-for-tat and provides an incentive so that every leecher contributes upload capacity to the distribution process. Otherwise, it is unlikely that other leechers unchoke it if the load in the swarm is high, i.e., if the number of leechers is large compared to the seeders. In addition to these 3 or 4 *regular unchoke slots*, an *optimistic unchoke slot* exists. Every 30 seconds the peer assigns this slot to a randomly chosen interested neighbor. The intention is to test if other peers might provide better download speeds and to bootstrap those leechers which do not yet have a complete chunk to share and cannot upload any data.

In seeder mode, the peer has already the complete file. Therefore, the download speed from other peers is no longer a reasonable metric to determine the active set of this peer. In this mode, different implementations of the choke algorithm exist. For example, the seeder can unchoke those peers to which it has the best upload speed. This maximizes the upload utilization of the seeder, but it might also promote free-riding since some peers can download the entire file very fast and then leave the swarm. Another option is that seeders choose the leechers in their neighbor set in a round robin manner. For that purpose, they randomly select a choked leecher every 30 seconds (like for the optimistic unchoke) and unchoke it. In addition, they choke the unchoked peer which has received data for the longest time. This leads to a more balanced distribution of the upload capacity of the seeder among the leechers, but it can reduce the upload speed of the seeder since some peers might not be able to download the content fast enough.

When a peer completes the download of a chunk, it informs its neighbors about this fact by sending a *have message*. However, in some implementations such messages are only sent to the neighbors which do not yet have the just finished chunk. The rationale for this *have message suppression* is that neighbors which already have this chunk will not change their interested status in the considered peer due to the newly completed chunk. This feature is intended to reduce the overhead caused by frequent *have messages*.

Unchoked peers can specify which chunks they want to download when they are unchoked. The corresponding algorithm is called *chunk selection* and it is guided by a set of policies. The most important policy is the *rarest first policy*. This policy makes the peer select that chunk for download which has the lowest number of copies within the neighbor set of the peer. In this way, it balances the number of copies of every chunk within the swarm and avoids that some chunks are very rare. The latter situation can lead to problems if the peers which have those rare chunks leave the swarm since the other peers cannot complete the download of the file any more.

The tracker is the only central component in a BitTorrent network. It keeps track of the peers in the swarm and sends possible contacts to requesting peers. In practice, there are also tracker-less swarms. In these swarms, peers exchange information about other peers in the swarm via the *peer exchange protocol (PEX)* that is specified in [63]. This has the advantage that no central component is required in the network. Still, at least one peer has to be known in order to join such a tracker-less swarm.

All this functionality of BitTorrent describes only how files can be shared within a swarm. However, it is not part of the BitTorrent protocol to provide access control or means to search for a given content. For that purpose, so-called *torrent index servers* exist. These servers contain meta-information about a large number of shared files, and the files are typically grouped by their content such as movies, music or software. From these servers, users can download a *.torrent* file, which contains information such as the address of the tracker, the size of the file and check sums of all chunks of the file. This permits that peers can easily verify the integrity of data received by other peers.

2.1.2 Measurements of Live BitTorrent Swarms

Chapter 3 presents results of a large measurement campaign of live BitTorrent swarms. Therefore, this section reviews related studies and discusses in which respect our work differs.

A survey on how to measure live BitTorrent networks in the Internet is given in [83] and a comprehensive overview on the entire BitTorrent ecosystem including all its components such as different torrent index servers and the popularity of different client implementations can be found in [78]. Other measurement studies [73, 88] investigate the incentives to publish content in BitTorrent-based content distribution networks.

In addition, a number of studies [77, 80, 82] measure live BitTorrent swarms and use their results as input for performance evaluations without a comprehensive presentation and discussion of the measurement results. Some results about the distribution of BitTorrent peers among autonomous systems (ASes) can be found in [72, 81], but it is not the main focus of these papers. In contrast, Chapter 3 is intended to provide input for other performance studies of BitTorrent and in particular for mechanisms which reduce inter-ISP traffic. For that purpose, a large variety of aspects is considered such as time dynamics, distribution of file sizes and peculiarities of swarms sharing regional content. Furthermore, statistical characterizations of these parameters are provided.

The authors of [42] follow the lifetime of one particular swarm (a Linux Redhat 9 distribution of size 1.77 GB) for 5 months in 2003 to analyze the performance of the BitTorrent distribution mechanism. To this end, they obtain the tracker log, which contains the peer population over time and upload/download statistics. The peer population clearly exhibits a flash-crowd behavior in the first few days with up to 4500 concurrent peers, but it decreases rapidly and stays below 500 peers after one month. Additionally, the authors investigate the session durations during the first 5 days and provide a geographical analysis of the peers in the swarm by mapping the IP addresses contained in the tracker log to countries. In contrast to this study, Chapter 3 is not intended to investigate the performance of BitTorrent, but to characterize the nature of real BitTorrent swarms. Therefore, it considers a large number of different swarms from different index servers and provides statistics, e.g., on the distribution of file sizes and peers over ASes.

The work in [49] is more closely related to Chapter 3 since the presented measurements are explicitly intended to provide input for mathematical modeling of BitTorrent swarms. Like in Chapter 3 of this monograph, the authors consider a large number of swarms and obtain statistics from different torrent index servers. They analyze the arrival and departure process of the peers and study the session durations. They observe typical flash-crowd behaviors shortly after the birth of new torrents and argue that Poisson processes are not suitable to model the arrival of peers over the lifetime of an entire swarm. In a similar way, live BitTorrent swarms are studied in [47]. There, the authors observe that the availability of files in BitTorrent becomes poor quickly since the peer arrival rate often decreases exponentially after a short popular phase of the torrent. Furthermore, Guo et al. [47] build a graph-based multi-torrent model and study the inter-torrent collaboration. The work of Chapter 3 extends these two studies in the following way. First, it investigates additional properties of the measured swarm such as the AS affiliation of the peers, which is important to evaluate the performance traffic optimization mechanisms, and whether the peer population exhibits diurnal patterns. Furthermore, it provides statistical characterizations these properties, which can be used as input for performance evaluations.

In [86], the authors investigate the download characteristics and the popularity of files in BitTorrent networks. They compare these metrics observed in a university campus with the ones in the global Internet. To this end, they measure the communication of peers inside the campus network with trackers. In addition, they track the torrent index server mininova.org and contact the trackers listed there to obtain the swarm statistics. Their main observations are that campus users typically download larger files than the average user and that files become popular on campus network earlier than at a global scale. Furthermore, most swarms experience their peak popularity not directly after their birth but several weeks later. While popularity of files is also one of the metrics studied in Chapter 3 of this monograph, the authors of [86] do not study the distribution of peers over ASes, which is a major aspect in Chapter 3.

2.1.3 Analytical Performance Models of BitTorrent

Since we provide an analytical model to investigate the impact of caches on BitTorrent-like P2P networks in Chapter 4, we shortly review related studies on analytical models of P2P networks in the following.

A very tight relation to the work presented in Chapter 4 have the analytical models of the system dynamics in BitTorrent systems [43,45]. In [45] the authors investigate the service capacity of P2P networks. They divide the evolution of the average throughput of such a network into two phases. The first one is the exponential growth of throughput, where every completed download provides a new source for the content. This transient phase is modeled by a branching process. The second phase describes the state of steady throughput, where some sources disappear while new ones are created. This phase is analyzed via a two-dimensional Markov chain model. The authors validate their analytical results by traces obtained from live BitTorrent swarms.

The fluid model presented in [43] is inspired by the Markov chain model of [45]. However, it is a deterministic model and it is used to study the system dynamics in a BitTorrent swarm, i.e., the evolution of the number of seeders and leechers, via a system of two coupled differential equations. The authors provide closed form solutions for the average number of seeders and leechers in the steady state by differentiating the two cases whether the upload capacity or the download capacity of the peers is the limiting factor in the system. In Chapter 4 we use this model and extend it to capture the impact of caching on the system dynamics of the swarm. In addition to the fluid model, the authors of [43] propose a Gaussian approximation to study the variability of the number of seeders and leechers around the mean value predicted by the deterministic fluid model. They validate their results with experiments in a local and in an Internet-wide setup.

The work in [46] also extends the fluid model of [43], but in another direction. It introduces classes of peers with different upload rates and evaluates how the allocation of upload capacity between these classes affects the performance of the system. The effects of churn and the download completion ratio are studied with an analytical model in [53]. Finally, Rimac et al. [64] use a fluid model to dimension the server capacity in an hybrid P2P content distribution network. Unlike these studies, we use our analytical model in Chapter 4 to evaluate the performance of caches in BitTorrent-like content distribution overlays.

2.2 Problem of Inter-ISP Traffic

In this section, we motivate the importance of optimizing the traffic in overlay networks for content distribution. To this end, we report findings from studies which measure and predict the amount of such traffic in the Internet. Afterwards, the section explains relations between ISPs and commonly used charging models. Finally, it describes the need for traffic optimization.

2.2.1 Traffic of Content Distribution Overlays

Overlays for content distribution are one of the major sources of Internet traffic. In recent years, they generated about 40 to 70 % of the total Internet traffic depending on the considered continent [71]. The highest fraction was observed in Eastern Europe at a value of slightly below 70 %. These numbers are based on traffic measurements which covered around 850000 Internet users and more than 1200 TB of traffic in the years 2008/2009.

More recent numbers are provided in the *Cisco Visual Networking Index: Forecast and Methodology, 2011-2016* [87] published in May 2012. Their approach to forecast the traffic composition in the near future comprises five steps: First, the number of Internet users are estimated based on information received from external analysts. Second, the application adoption is estimated, i.e., the popularity of certain applications. The third step is to estimate the minutes of use for each application type. Those can then be transformed to bitrates, which lead in the end to the traffic estimate.

According to this study [87], the largest amount of consumer Internet traffic today is owed to P2P-based content distribution overlays. In 2011, the total consumer Internet traffic worldwide accounted for more than 20 exabytes per month. P2P file transfer accounted for a share of around 22 %, i.e., for around 4.6 exabytes per month. The authors expect that the absolute volume of this P2P traffic will increase at a compound annual growth rate of about 8 % until it reaches 10 exabyte per month in 2016. As a consequence, such P2P-based content distribution overlays will remain a significant source of Internet traffic in the near future, although other types of traffic such as video transmission (real-time or video-on-demand) will grow faster than P2P traffic. However, one has to keep in mind that content distribution overlays and the respective mechanisms can also be used for these purposes, which is explicitly not taken into account in [87]. This might increase the fraction of consumer Internet traffic which is guided by P2P content distribution mechanisms even further in the future.

2.2.2 ISP Relations and Charging Models

The Internet of today consists of a large number of autonomous systems (ASes) operated by many different companies called Internet service providers [34]. These ASes are interconnected at Internet exchange points. All ASes run the Border Gateway Protocol (BGP) to determine the path between two hosts in different ASes. However, besides the physical interconnections, commercial relations exist between ISPs, which determine the BGP configurations, i.e., which ISP forwards data of which other ISP to the rest of the Internet.

These relations are often confidential and complex, but can be simplified and classified in three groups: provider-to-customer (p2c), peer-to-peer (p2p)[1] and sibling-to-sibling (s2s), cf. [38, 55]. In the following we describe these relations

[1] The term *peer-to-peer* might be a bit misleading in the context of this monograph since it is used for the relation of two ISPs and for the distribution paradigm. To avoid confusion, we rigorously use *p2p relation* for the ISP relation and *P2P network* for the distribution network in this section.

and explain the structure of the Internet in a simplified form. This description does not capture all details, but it is still sufficient to illustrate the qualitative impact of inter-ISP traffic on the different types of ISP.

The p2c relation is a direct relation between two ISPs. In this relation, the provider (ISP A) provides Internet transit to its customer (ISP B), i.e., ISP A connects ISP B to the Internet. That means that ISP A forwards traffic from ISP B (and the customers of ISP B) to the rest of the Internet and traffic from the rest of the Internet to ISP B (and to the customers of ISP B). As a financial compensation ISP B pays ISP A for the transit service. This payment is typically based in some way on the traffic volume, but the actual charging model may vary. A popular charging model is the 95th-percentile model. Here, the rate of the exchanged traffic is monitored in time slots of five minutes and the price is calculated at the end of the accounting period, typically one month. The customer then has to pay for a data rate which is large enough so that it is not exceeded in 95 % of the time slots.

ISPs of similar size often agree on a p2p relation. That means that ISP A forwards traffic originating in ISP B or in the customers of B to destinations located within ISP A or the customers of A. This exchanged traffic is normally not charged by the ISPs but is expected to be balanced, i.e., both ISPs should sent roughly the same amount of such traffic. The s2s relation between two ISPs means that they both provide transit, i.e., Internet connectivity for each other. Such relations can for example be used for connection backup in case that another transit provider fails.

These relations lead to hierarchical structure of the Internet with a number of *tier 1* ISPs at the top. These tier 1 ISPs have no providers and are interconnected with each other using p2p relations. Below them, a number of *tier 2* ISPs exist. These are typically of smaller size and are customers of a set of *tier 1* ISPs. Even smaller ISPs are grouped into the category of *tier 3* ISPs.

2.2.3 The Need for Traffic Optimization

As stated above, P2P-based overlays for content distribution contribute a large fraction to the total Internet traffic. As a consequence, the first reason to optimize such traffic is to improve the network efficiency and to reduce the load in the network. Furthermore, the end users of these networks might also benefit from traffic optimization, for example through faster downloads. Finally, there are financial reasons to optimize the traffic of content distribution overlays, in particular to reduce inter-ISP traffic.

For *tier 1* ISPs, increased inter-ISP traffic is a potential source of revenues. However, ISPs of smaller size have to pay their providers for Internet transit. Therefore, such ISPs have a financial incentive to reduce inter-ISP traffic. From a theoretical point of view, it might also be more profitable for a *tier 2* ISP that the peers located in its network exchange more traffic with peers within its customer ISPs so that less traffic is routed to its provider ISPs. The feasibility of such optimization mechanisms might however be limited and such sophisticated scenarios are out of scope of this study. An evaluation of the potential of these selfish strategies can be found in [26]. Within this monograph, the focus of traffic optimization is exclusively on the reduction of inter-ISP traffic.

2.3 Approaches to Traffic Optimization

Due to the expensive nature of inter-ISP traffic especially for small and medium-sized ISPs the research community has put considerable efforts into the investigation of mechanisms to mitigate this problem. These mechanisms can be grouped into the two categories *locality-awareness* and *caching*.

Locality-awareness represents a class of mechanisms which equip the participants of the overlay network, i.e., the peers, with information about the underlying physical network topology. This facilitates that network topology information can be taken into account to determine the actual data exchange procedures within the overlay. In this way, the overlay network can optimize its distribution process according to the physical network topology, which is expected to decrease inter-ISP traffic significantly. Implementations of such mechanisms for BitTorrent and other content distribution overlays are under discussion in the IETF working group on application layer traffic optimization (ALTO) [62]. As a consequence, such proposals are also called *ALTO mechanisms*. Section 2.3.1 explains the concept in detail, presents proposals for concrete implementations and reviews related evaluation studies.

The concept of caching is not specific to content distribution overlays, it has already been applied for several years to web traffic [31, 32, 37]. The goals of web traffic caching are to reduce server load and decrease the required network bandwidth. For that purpose, proxy caches which are closer to the user serve requests on behalf of the original web server. However, caching is not limited to web traffic, it is also applied by ISPs for P2P-based content distribution networks [39, 48]. Some commercial products such as PeerApp's UltraBand [90] and OverSi's OverCache P2P [75] have already been available on the market for some years. In Section 2.3.2 we give an overview on caching techniques for P2P traffic, review related studies and describe how the work presented in Chapter 4 differs from them.

2.3.1 Locality-Awareness

In most of today's overlay networks for content distribution the actual data exchange is exclusively determined by overlay metrics. Among them are for example the availability of a specific piece of content, but also the current transmission speed between two participants or fairness metrics such as the sharing ratio, which represents the relation between the amount of uploaded and downloaded data. Since the data exchange is mostly guided by such information, it might lead to inefficiencies from the network perspective, i.e., from the view point of the ISP.

Locality-awareness tries to address this problem by incorporating network topology information in the process which determines the data flow within the overlay. The aim is that peers which are located close to each other in the physical network preferentially exchange data with each other. Most proposals to implement this idea in the current Internet require a form of localization service, which provides access to topology information to the overlay network. Furthermore, different proposals exist about which information should be accessible by the overlay network since ISPs typically do not want to disclose their actual network topology. In addition, the algorithms of the overlay network need to be adapted to take into account network information. In the remainder of this section we discuss these issues in detail and explain proposals taken from literature.

Localization Services

One of the earliest proposals for a localization service is presented in [54]. In this work Aggarwal et al. investigate an *oracle* service, which is provided by the ISP. Peers can send a list of possible sources for the content they desire to this oracle and the oracle orders the list according to the ISPs preferences. For example, the oracle can rank the peers based on their AS affiliations, the number of AS hops or the distance to the edge of the AS of the requesting peer. Furthermore, geographical information can be included in the ranking process or performance information such as available bandwidth or congestion in case the oracle has access to such information.

In this way, the use of an oracle service facilitates that the ISP can influence the traffic exchange in the overlay network, which was not possible before. In addition, this process is not only expected to improve the ability of the ISP to do traffic engineering of its overlay traffic, but also to improve the performance experienced by the peers in the overlay network in terms of low latency and high throughput [54]. Since the oracle service only ranks candidate peers based on certain metrics, it does not need to disclose concrete information about the physical network topology of the ISP. This permits that the ISP can guide the peer selection process in the overlay network without revealing confidential information about its network topology. Nevertheless, some form of reverse engineering of such information might be possible if the oracle is asked to rank a large number specifically targeted candidate lists of peers.

The authors of [54] investigate the performance of the oracle by graph based simulations, simulations of the Gnutella protocol [41] and testbed experiments with a modified version of Gnutella client [79]. They show that graph properties of the overlay network are not adversely impacted by the use of the oracle service. In addition, it increases the intra-AS localization of the overlay traffic significantly.

Another form of localization service proposed by Xie et al. [66] is the *iTracker*. In fact, the authors propose an entire architecture called *provider portal for applications (P4P)* which is not only targeted at content distribution overlays, but the study uses these overlays to illustrate and investigate the P4P architecture. The iTracker can be queried by peers or application trackers, but the evaluation in this work focuses on the case that the iTracker only communicates with the application trackers. For this communication, the iTracker has defined interfaces of which the most important one is the *p-distance* interface. It allows applications to query the iTracker about the "cost" of the path between two end points. This p-distance can be based on current status of the network, preferences of the ISP regarding traffic engineering or the distance of the two end points. The authors provide an example how to calculate the p-distance to minimize the maximum link utilization in the network of the ISP and explain how this problem can be decomposed so that a distributed solution on a number of iTrackers is possible.

For scalability and privacy reasons, every IP address is associated with an opaque ID (PID), which can serve as an aggregation node. A PID can for example be a point-of-presence (PoP) in the network of the ISP and aggregate all IP addresses connected to this PoP. The iTracker computes the p-distance p_{ij} between two PIDs i and j and communicates the result to the application. However, other definitions of a PID are also possible. This permits to tune the PIDs between fine- and coarse-grained localization information. In general, fine-grained localization might facilitate better optimization algorithms, but it faces scalability problems at the iTracker and exhibits a very detailed view on the network of the ISP, which probably contradicts privacy policies of the ISP. This type of communication is currently under standardization in the ALTO working group of the IETF [85].

For the evaluation of the P4P architecture, the authors modify a number of P2P protocols including BitTorrent to communicate with the iTracker and to incorporate the p-distance in the peer selection process. The modified version of the tracker selects the neighbors for a requesting peer in three steps. First, up to 70 % of the m neighbors are selected from the same PID if available. Second, neighbors from the same AS are selected so that the total number of selected neighbors does not exceed 80 %. Finally, neighbors from other ASes are selected. The last two steps are determined by the p-distance. This modification is a form of biased neighbor selection, which is discussed later in this section in more detail.

To quantify the performance, simulation studies are performed as well as Planet-Lab [58] based Internet experiments. The studied performance metrics are (1) the completion time (time to download the file), (2) the bandwidth distance product (BDP, average number of backbone links that a unit of P2P traffic traverses), (3) the P2P traffic on top of the most utilized link and (4) the charging volume (according to 95th percentile charging model). The investigations show that all these metrics can be reduced considerably by the use of the P4P architecture compared to the legacy application peer selection in simulations and in the Internet experiments.

Choffness et al. [59] propose a localization service that does not rely on additional infrastructure like for example an iTracker or an oracle. Instead, they propose to use the existing infrastructure of content distribution networks (CDN) such as Akamai [84] in the following way. CDNs maintain a large number of servers around the world to serve content to end users at a high performance. To dynamically select a server that is close to the end user and to permit load balancing within the CDN, it uses the domain name system (DNS). The user requests the content at a static hostname, but this hostname can be resolved to different IP addresses depending on the geographic location of the users.

Choffness et al. provide a plug-in called Ono for the popular BitTorrent client Vuze that exploits this fact in the following way. It repeatedly queries the Akamai

DNS servers for a number of hostnames and saves the redirection ratios to different IP addresses. To judge how close another candidate peer is, Ono compares the redirection ratios of itself and the other peer using the cosine similarity. If the redirection behavior is similar, those peers preferentially exchange traffic.

In their evaluation, the authors show that the CDN redirection behavior is in fact a good predictor of short paths between peers in terms of AS and IP hops and of high performance in terms of high throughput and low latency. However, a considerable fraction of peers in a swarm need to be Ono-enabled since only those peers can exchange information about the redirection behavior. That means that Ono peers cannot determine the proximity of non-Ono peers. As a consequence, Piatek et al. argue based on Planet-Lab experiments that the impact of Ono is small in practice [70].

The topology-aware BitTorrent (TopBT) client [77] goes one step further in not relying on the CDN infrastructure. It tries to obtain all information about network proximity from the network itself by operating system commands such as *ping* and *traceroute* for latency and IP hops. For the AS affiliations and the number of AS hops, it downloads prefix-AS mappings and BGP routing table dumps from public databases such as RoutesViews, RIPE NCC and China CERNET.

Overlay Modifications to Incorporate Localization

To increase the locality of P2P traffic from content distribution overlays it is not only required to provide information about the physical network topology to the overlay network. In addition, it is also necessary that the overlay networks incorporate this information in the mechanisms that guide the traffic exchange process within the overlay. In the following we present possible modifications of the BitTorrent protocol which take into account topology information.

An early proposal by Bindal et al. [51] is to bias the neighbor selection of BitTorrent so that it contains a large fraction of peers from the same AS. To implement this behavior in practice, the authors propose two strategies. The first one is to modify the overlay software. If the tracker has access to AS affiliations of the peers, it can select the appropriate set of candidate peers when it is asked

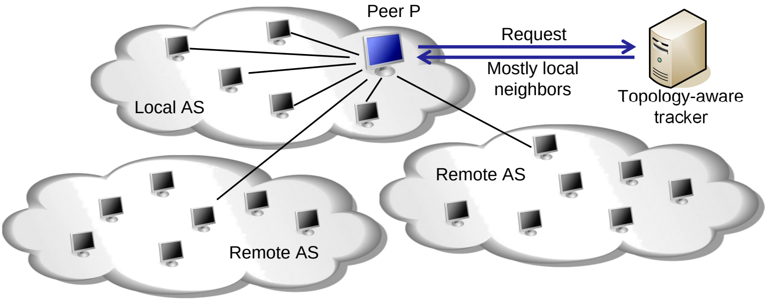


FIGURE 2.2: Biased neighbor selection. The tracker has information about the AS affiliation of the peers and returns lists of possible neighbors that preferentially include peers from the same AS as the requesting peer.

for possible neighbors by a new peer joining the swarm. This process is illustrated in Figure 2.2. The other option mentioned in [51] is to use deep packet inspection (DPI). In this way the ISPs can keep track of peers within their network and intercept requests of new peers to the tracker to replace the peer list with peers from the same AS.

Bindal et al. investigate the performance of biased neighbor selection via simulations of the modified BitTorrent protocol. They simulate a BitTorrent network with 700 peers, which are equally distributed among 14 ASes. They investigate the impact of the fraction of local peers in the neighborhood, the impact of the so-called university nodes and the one of bandwidth throttling by ISPs. Performance metrics are the download time of the peers and the traffic redundancy, i.e., the amount of data downloaded from peers in remote ASes divided by the file size. The simulations show that bandwidth throttling reduces traffic redundancy but increases the download time at the same time. In contrast, biased neighbor selection permits to reduce traffic redundancy even more without a negative impact on the download time, even if all but one neighbor are from the same AS as the requesting peer. Furthermore, biased neighbor selection is in particular helpful

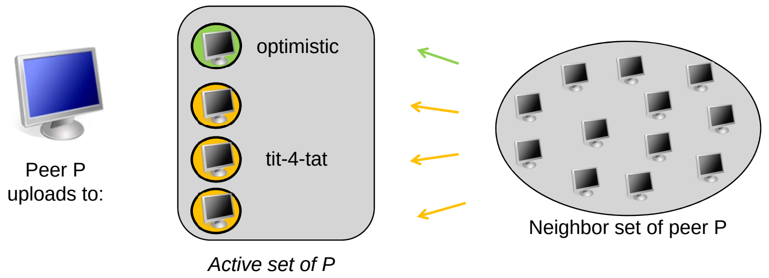


FIGURE 2.3: Biased unchoking. In BitTorrent, a random neighbor is chosen for the optimistic unchoke slot. In contrast, biased unchoking selects a random neighbor from the same AS for this slot.

for the peers to avoid the bandwidth bottleneck if the ISP throttles connections leaving its AS.

However, the neighbor selection mechanism is not the only option to bias the traffic exchange in the overlay. The actual traffic exchange in BitTorrent is determined by the choke algorithm, which decides to which neighbor a peer uploads data. As a consequence, the choke algorithm is also suitable to increase the locality of BitTorrent traffic. The corresponding concept is called biased unchoking and is proposed in [17]. Since it does not rely on a particular neighbor selection mechanism, it can be used as a complementary mechanisms to biased neighbor selection.

With biased unchoking, only the selection of the peer for the optimistic unchoke slot is changed. The regular unchoke slots, which are allocated according to the tit-for-tat policy in the leecher mode, are not modified to keep the sharing incentives of BitTorrent untouched. In the default BitTorrent algorithm, a random, choked and interested neighbor is chosen for the optimistic unchoke slot. In contrast, biased unchoking selects a random, choked and interested neighbor from the same AS with a probability p, where $p = 1$ is used in the evaluation in [17]. This process is illustrated in Figure 2.3.

The performance evaluation in [17] is based on simulations with homogeneous peer distributions over ASes. The authors investigate the impact of the number of peers per AS and the impact of bandwidth throttling by the ISPs, like in [51]. In addition, they study the performance when biased unchoking is used in combination with biased neighbor selection. The simulations show that this combination is in particular effective both to reduce inter-ISP traffic and to mitigate the negative effects of bandwidth throttling for the peers. The reason is that biased neighbor selection achieves that many neighbors of the peers are from the same AS and biased unchoking facilitates that these neighbors are used for the actual traffic exchange.

The study in [22] investigates the performance of biased neighbor selection and biased unchoking when the bias is based on metrics derived from BGP preferences of the ISPs. This approach is also suggested in [66]. The evaluation shows that it can also reduce inter-ISP traffic considerably compared to the default BitTorrent case.

Performance Evaluation Studies

The localization of overlay traffic has received considerable attention in the research community during recent years. Therefore, a number of studies exist which evaluate the performance of these proposals under different scenarios and by different means. In the following, we summarize these studies. Finally, this section describes how this monograph differs from them.

The seminal study of Karagiannis et al. [48] investigates the potential of traffic localization in P2P overlay networks based on BitTorrent tracker logs and payload packet traces. They study the hit ratios (files, bytes and pieces that have already been downloaded in the respective ISP) and the peer overlap in time since only concurrent peers in the same AS can exchange data. The measurements show that around 10 % of the files are downloaded by at least 2 peers simultaneously within the monitored ISP and this is the case for 30 to 70 % of the trace time depending on the considered file. Using the traces they simulate different content distribution scenarios including client/server, random P2P, locality-aware P2P and caching. The results show that random P2P consumes considerably more inter-ISP traffic than client/server, but locality-aware can mitigate this negative effect and reduce inter-ISP traffic.

Similarly, a set of studies [6,20,80,81] extend the above presented evaluations of biased neighbor selection [51] and biased unchoking [17] to more realistic scenarios derived from real-world traces. The studies [6,20] use a simplified form of the distribution of peers over ASes proposed in Section 3.3 to define realistic scenarios. In these scenarios, they discover the effect that biased unchoking can lead to unbalanced download times, i.e., some peers can download faster than others depending on their AS affiliations. In addition, countermeasures for that problem are proposed and evaluated.

Blond et al. [80] also base their scenario definition on real-world tracker traces. For that purpose, they crawl 200,000 torrents spread among more than 9,500 ASes. Using experiments in controlled environments with real BitTorrent clients, they investigate the two questions (1) how far locality can be pushed and (2) how large is the reduction of traffic by locality at the scale of the Internet. For that purpose, they experiment with homogeneous and skewed, real-world distributions of peers over ASes. The authors propose improvements to traditional locality mechanisms and show that their locality mechanisms could have saved 40 % of the total traffic generated by the swarms contained in their measurement trace without a negative impact on the download time.

Cuevas et al. [81] also base their evaluations on tracker traces of more than 40,000 swarms. By mathematical modeling they derive upper and lower bounds for the inter-ISP traffic reduction across hundreds of ISPs. In addition, they investigate three different locality policies, which they also implement in the Mainline BitTorrent client and test by connecting to live swarms with the modified client. Finally, they highlight the fact that the win-win situation of end users and ISP is given in many torrents, although a few *unlocalizable* torrents exist with only very few peers in the same AS, where locality is either not possible or harmful for the user experience.

The work contained in this monograph differs from the aforementioned studies on localization services, overlay modifications and the respective performance evaluations in the fact that it does not propose and evaluate concrete localization services for content distribution overlays. Instead, it provides a comprehensive measurement study of the nature of BitTorrent swarms in the current Internet in Chapter 3. This is intended as a basis for performance evaluation of localization services and overlay modifications. For that purpose, this chapter also presents statistical characterizations, which serve as input for simulations and scenario definitions. In addition, this monograph investigates the performance of P2P caching as an alternative means to reduce inter-ISP traffic in Chapter 4. This is not considered in the aforementioned studies. Literature related to caching is reviewed in the following.

2.3.2 Caching

Caching of P2P traffic has been a hot research topic in recent years and several studies exist which are related to the work that is presented in Chapter 4 of this monograph. In the remainder of this section we explain the basic concept, describe the content of related studies and discuss the differences to the work in Chapter 4 of this monograph.

Caching of P2P traffic is inspired by traditional caching of web traffic. The basic idea is that the ISP provides a cache in its network, which stores content that is requested multiple times. Consequently, all requests to that content can be served by the cache and the content does not need to be downloaded from remote locations. This reduces incoming inter-ISP traffic. This process is illustrated in Figure 2.4. Different types of implementation are possible for this concept, but this basic design holds for all of them. For example, in some implementations the peers are aware of that requests are served by the cache. Furthermore, different ways exist of how and whether peers are informed of possible caches. In Section 4.1, we present the different types and their functionality in more detail.

The earliest works on caching of P2P traffic [39, 48] focus on the achievable cache hit ratios, i.e., how often the cache can serve a request to a given content. The aims of Leibowitz et al. [39] are to measure the characteristics of P2P traffic,

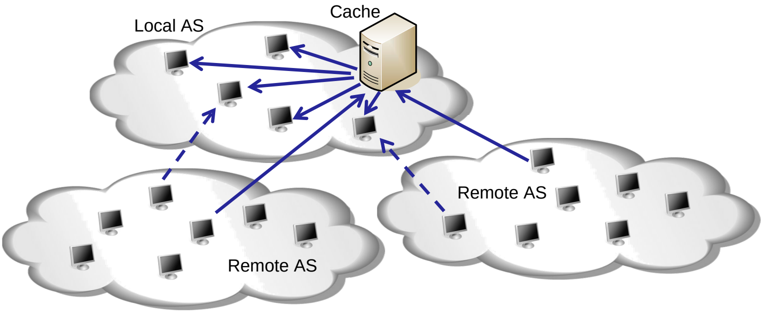


FIGURE 2.4: Caching of P2P traffic. The cache stores popular content and serves requests of the peers within the network of the ISP. This avoids that this data needs to be downloaded from remote peers and decreases incoming inter-ISP traffic. Depending on the cache design peers can still download other parts from remote locations (dashed arrows), which is not considered in related studies.

to compare it to http traffic and to investigate whether P2P traffic can be cached. For that purpose, they transparently collected the P2P traffic at a major Israeli ISP for a duration of one month. They observe that wide majority of P2P traffic is caused by the download of movie files and that around 20 % of the downloaded files account for 80 % of the total P2P traffic. This finding from the year 2002 is in line with the results that we present in Chapter 3, although the typical size of the files has changed since then from around 5 MB to several hundred MB in 2009.

In addition, Leibowitz et al. investigate the theoretical caching potential and the empirical performance of a caching mechanism. The theoretical investigation is based on replays of the measured trace where the most popular files are identified and the theoretical byte hit rate is calculated depending on the available disk space. The results show that the achievable byte hit ratio can reach up to 67 %, i.e., this fraction of traffic could have been saved by using a cache. Furthermore, only 200 GB of disk space are required to achieve a byte hit ratio of 60 %. In their empirical validation using the implemented caching algorithm in the network of the ISP, they measure a byte hit rate of 50 %.

The aforementioned study [48] by Karagiannis et al. does not only consider locality-aware P2P content distribution but also caching of P2P traffic. In their study, they consider three payload packet traces from an ISP in the year 2004 with a duration slightly longer than one day. They report possible savings of a perfect caching infrastructure, where each involved ISP installs a cache with unlimited disk storage, of about 90 % compared to a pure client server model.

Since disk storage of the cache is not unlimited in practice, the efficiency of different cache eviction policies is studied in [44] for the FastTrack file-sharing protocol based on the same traces as used in [39]. Cache eviction or replacement policies are used by the cache to decide which content should be removed from the disk to store new content if the disk storage is exhausted. Such policies can operate both on file level, i.e., only entire files can be replaced, or on range level, i.e., certain ranges of a file can be replaced. The policies investigated in [44] comprise policies such as *least recently used (LRU)*, *least frequently used (LFU)*, *minimum size (MINS)*, *least sent bytes (LSB)* and some more sophisticated ones. Their results show that the eviction policies have a large impact on achievable hit ratio of the cache and that the hit ratio can reach up to 80 %. Finally, the LSB policy performed best in their evaluations when entire files are replaced.

Hefeeda and Saleh [61] build on [44] and design and evaluate an improved caching algorithm termed *proportional partial caching*. This algorithm is based on segmentation and admits new files and evicts old files only partially. The algorithm is inspired by the nature of the object popularity distribution which the authors measured during a period of nine months in the Gnutella P2P network. The investigation of this trace shows that the object popularity follows a Mandelbrot-Zipf distribution. In contrast to the Zipf distribution, which is used to model the popularity of web objects, the Mandelbrot-Zipf distribution has an additional parameter and a flattened head. This permits that this distribution fits well the popularity of P2P objects, where the most popular objects are not as often requested as predicted by a pure Zipf distribution (cf. [44, 61]).

The facts (1) that the set of popular objects is not limited to a very small number of files and (2) that most files are very large make replacement policies ineffective. This is in particular true for caches which store complete files upon the first request. Therefore, the idea of proportional partial caching is to partially admit new files to the disk storage of the cache and to increase the portion of the file if more requests to this file occur. The evaluation of this mechanisms using the Gnutella traces shows that this algorithm outperforms the ones presented in [44].

The work presented in [82] investigates how different ISPs can cooperate in caching P2P content and to which degree this increases traffic savings. Cooperation means here that the cache of an ISP serves not only requests of the peers within its network but also requests of peers in cooperating ISPs. The study uses game theory to model cooperative caching as n-person non cooperative game. The author shows that a pure strategy Nash equilibrium exists. In addition, he proposes two algorithms to solve the game and investigates the potential benefits for the ISP via trace-driven simulations on a real AS topology of northern Europe.

The focus of the aforementioned works was mostly on the achievable cache hit ratios [39, 48], on the efficiency of different cache eviction policies [44, 48, 61] or on cooperation algorithms between ISPs. All of them used trace-driven simulation in order to assess the performance of the studied algorithm. The traces are measured in networks of ISPs or taken from popular file-sharing servers. For their evaluations most of these studies assume that (1) peers inside the ISP download all content available at the cache exclusively from there and (2) do not change their uploading behavior due to the data received from the cache, which are both typical assumptions for trace-driven evaluations. In contrast to these studies, we do not rely on these two assumptions in Chapter 4 of this monograph. Instead, we study the efficiency of caches in BitTorrent-like P2P networks under the assumption that all required files are available at the cache. We develop a model of the impact of caches on the peer population and derive the inter-ISP traffic based on the peer population in the different ISPs. This shows that these two effects have an important impact on inter-ISP traffic. As a consequence, the work presented in Chapter 4 is complementary to the aforementioned studies [39,44,48,61,82].

3 Overlay Networks in Today's Internet: Measurements and Characterizations

The performance evaluation of traffic optimization techniques in overlay networks requires a thorough knowledge of the nature of current overlay networks. Such knowledge is helpful to define scenarios, models, and parameters for performance evaluations that reflect the characteristics of real-world overlay networks. In this way, it ensures that performance results give meaningful insights for real-world scenarios.

The most popular overlay network for content distribution today is BitTorrent. It is also responsible for a large fraction of intra- and inter-ISP traffic in the Internet [71]. Therefore, we use it as an example overlay network in this chapter and investigate the properties of today's BitTorrent networks. To this end, we perform a large-scale measurement study of live BitTorrent swarms and derive important characteristics relevant for traffic optimization in overlay networks. The measurement results comprise a comprehensive set of swarms for different types of content listed at the index servers *mininova*[1] and *piratebay*.[2] We have measured the swarm size as well as the swarm dynamics in terms of number of leechers and seeders, and the distribution of peers over ASes per swarm. We have also analyzed the details of individual swarms to understand content clustering, e.g., the availability of certain content in specific regions only. The measurements have

[1] http://www.mininova.org/

[2] http://thepiratebay.se/

been performed from June 2008 to May 2009 using the PlanetLab [58] and G-Lab experimental facilities [60]. Some additional measurement results are provided in our technical report [68]. Based on these measurements and an additional public data set, we derive characterizations of the swarm size, the distribution of the peers over ASes, the fraction of peers in the largest AS, and the size of the shared files. In addition, we present multivariate correlation matrices of these parameters to show to which degree these values depend on each other. In particular, our characterization of BitTorrent swarms reflects that peers are not homogeneously distributed among ASes, but most of the peers are located in a small number of top ASes. Furthermore, quantitative results on the skewness of the peer distribution based on the measurements are provided.

The measurement results and the characterizations serve as input for the performance evaluation of traffic optimization in overlay networks to gain insights into the behavior of proposed solutions for traffic optimization in overlay networks under real-world conditions. Therefore, we use them in Chapter 4 to investigate the optimization potential of different caching strategies in BitTorrent networks. In addition, our measurement results show the composition of a large set of swarms observed in the Internet, which can be used to assess the overall gain of a proposed solution if the gain achieved in some typical scenarios is known. In particular, they show that 80 % of the BitTorrent peers are located in 20 % of the swarms. Finally, a deeper understanding of AS-level properties of real BitTorrent swarms helps in refining current proposals and in designing new mechanisms.

The content of this chapter is mainly taken from [5]. Its remainder is organized as follows. We explain the measurement setup in Section 3.1 and provide the measurement results in Section 3.2. Based on these results, we present the corresponding statistical characterizations for BitTorrent swarms in Section 3.3. Finally, Section 3.4 summarizes this chapter.

3.1 Setup of Overlay Measurements

The measurement setup described in this section aims at gathering data about live BitTorrent swarms. The data serves as input to derive characterizations of the parameters relevant to traffic optimization in overlay networks in Section 3.2.

3.1.1 Conducted Measurements

To gain a more diverse view on the characteristics of existing swarm types than in the literature, we chose specific sets of swarms to measure. These are defined by a number of selection criteria which help to define a number of swarm classes. In contrast to [72], we do not only analyze swarms found on one index and only distributing videos. Instead, we expand the insights gained from observing these swarms to other classes of swarms as well. According to a certain selection criterion and the desired type of content, the `.torrent` files are downloaded from a torrent index. As *selection criteria*, we consider (a) all available torrents, (b) the most popular torrents in terms of number of peers in the swarm, and (c) the most recent files which have been published in the last 24 hours. As *type of content*, we distinguish between (1) music files, (2) TV series, (3) movies, (4) so-called “regional” movies which are in a certain language (German, Spanish, French, Italian, Dutch), and (5) all media independent of the type of content. These types are based on the user classifications at the torrent index servers. The considered *torrent index servers* cover the currently most popular ones in the Internet, (i) PirateBay, (ii) Mininova, and (iii) Demonoid [78]. Here, the criteria (a)(3) and (a)(4) correspond to the class of swarms evaluated in [72]. Thus, we additionally consider other content types and indexes as well as specific subsets of swarms.

Table 3.1 summarizes the measurement experiments conducted over the period from June 2008 to May 2009. Each measurement experiment is assigned a unique identifier `ID`, which is used when describing the measurement results. In particular, we measure in each experiment the swarm size, the swarm dynamics, and the distribution of peers over ASes (’peer-dist.’). In order to measure the total number N of peers in a swarm and their corresponding ASes, we contacted the tracker and requested a list of peers. As a result, the number of seeders S and leechers L, and a set of k different IP addresses of peers are returned.

Since a tracker typically returns $k = 50$ IP addresses for a single request, we used a large number of machines with BitTorrent clients running on each of them. They contact the tracker simultaneously in order to get the IP addresses from all peers in the swarm at a single point in time, i.e. a snapshot of the swarm. In particular, several requests are sent within 5 minutes from all 219 nodes in PlanetLab and 153 nodes in G-Lab, respectively, until $N = S + L$ different IP addresses are obtained. Then, the IP addresses are mapped to the origin AS using the RIPE database[3]. This measurement method is referred to as *distributed monitoring* in the remainder of the paper. However, for measuring the swarm size only, it is sufficient to monitor the tracker (denoted as 'tracker monitored' in Table 3.1 for setups `Pop` and `24h`) or to parse the website of the torrent index ('website parsed'), as done in experiment `TV`. Additionally, we consider a publicly available data set from Khirman [69] with measurement results of the swarm sizes of torrents on different torrent index servers (`KPi`, `KDe`, and `KMi`). With all three techniques ('website parsed', 'tracker monitored', and 'distributed monitoring') we can measure the swarm size. The distribution of peers over ASes can only be measured in those experiments where we used distributed monitoring of the tracker (cf. columns 'techniques' and 'observed' in Table 3.1) which is an extension of the method 'tracker monitored'.

To study the time dynamics of a swarm, several samples of the swarm size and the distribution of peers over ASes are captured over a longer period of time which is denoted as "xx samples every yy hours" instead of "snapshot" in the column "measurement per swarm" in Table 3.1. In that case, for example the average swarm size over this period of time is given, which may result in a decimal number, while a snapshot of a swarm always returns an integer value.

The different data sets describe the BitTorrent swarms under consideration with a different level of detail (Table 3.1). For example, the distribution of peers over ASes is only studied for the experiments performed in April 2009, i.e., `Grp`,

[3] RIPE NCC, http://www.ripe.net/data-tools/stats/ris/riswhois

Mov, Mus, Reg and Ele. The reason is that we started with a rather basic technique ('website parsed') in June 2008 and improved it during the course of this work. Therefore, we are not able to present the distribution of peers over ASes for the experiments TV, Pop, and 24h, and this information is also not contained in the data sets KPi, KDe, and KMi we took from [69]. Furthermore, data about the change in the number of peers over time is only available for the experiments TV, Grp, and Ele. This is partially owed to feasibility reasons, in particular for the Mov and Mus experiments the number of swarms was too high to take hundreds of samples of the swarm via distributed monitoring. While one needs to be aware of the aforementioned issues when interpreting the data, we suppose that their impact on the presented results is small and that the measurements remain comparable. For example, we will show for the Mov and Mus data sets that the IP addresses obtained via distributed monitoring are in good accordance with the number of peers obtained by tracker monitoring (cf. Figure 3.1).

Some BitTorrent swarms exist without a tracker and are therefore called *tracker-less*. In these swarms the peers exchange the addresses of other peers in the swarm among each other using the peer exchange (PEX) protocol [63] or the Kademlia DHTs built into uTorrent and Vuze. Since it is not possible to monitor those torrents with the aforementioned techniques, tracker-less torrents are not considered in this study.

3.1.2 Distributed Monitoring of a Tracker

The distributed monitoring of a BitTorrent tracker for obtaining the distribution of peers over ASes relies on experimental facilities, like PlanetLab [58] or G-Lab [60], with a large number of nodes. They are controlled by a central unit C which is located at the University of Wuerzburg in our measurements. C has established connections to the used PlanetLab and G-Lab nodes Ω. C is responsible for the distribution of the .torrent files to these monitoring nodes Ω, the initialization of the monitoring on Ω and the collection of the created result files from Ω. The monitoring on each node itself is realized with a python script that queries

TABLE 3.1: Overview on conducted measurement setups.

ID	torrent index	selection criteria	type of content	meas. per swarm	#torrents	technique	observed	meas. date
TV	PirateBay	all available	TV series	96 samples over 36 hours	63,867	website parsed	swarm size	Jun. 2008
Pop	PirateBay	most popular	movies	snapshot	4,463	tracker monitored	swarm size	Mar. 2009
24h	PirateBay	last 24 hours	all media	snapshot	1,048	tracker monitored	swarm size	Mar. 2009
Grp	Mininova	groups w.r.t. size & language	movies	440 samples over 88 hours	16	distributed monitoring	swarm size and peer-dist.	Apr. 2009
Mov	Mininova	all available	movies	snapshot	126,050	distributed monitoring	swarm size and peer-dist.	Apr. 2009
Mus	Mininova	all available	music	snapshot	135,679	distributed monitoring	swarm size and peer-dist.	Apr. 2009
Reg	PirateBay	top 30	regional movies	snapshot	120	distributed monitoring	swarm size and peer-dist.	May 2009
KPi	PirateBay	all available	all media	snapshot	1,682,355	data taken from [69]	swarm size	Mar. 2009
KDe	Demonoid	community selected titles	all media	snapshot	11,759	data taken from [69]	swarm size	Mar. 2009
KMi	Mininova	legal torrents promotion	all media	snapshot	4,514	data taken from [69]	swarm size	Mar. 2009
Ele	open movie "Elephants Dream"			8,640 samples over 24 hours	1	distributed monitoring	swarm size and peer-dist.	Apr. 2009

a tracker n times every t seconds. In our measurements, t is set to $15\,\mathrm{seconds}$ to avoid overloading the tracker, while n is chosen according to N, using the analysis described below.

In the following, we derive the number Y of required monitoring nodes in order to obtain all IP addresses of N peers in a swarm. Upon each request, the tracker returns a subset of $k = 50$ peers which are randomly chosen from all N peers. Denote by X the number of times the tracker has to be contacted to get N different IP addresses. The derivation of X is known as the *coupon collector's problem* [56]. [52] derives an exact solution which is given in the following.

Let $P(j,i)$ denote the probability to observe j different IPs after the i-th tracker response. It is $P(j,i) = 1$ for $j \le k$ and $i > 0$ since the first tracker response returns k different IPs. It is $P(j,i) = 0$ for $j > \min(ik, N)$, since a maximum of ik different IPs are retrieved after the i-th tracker response and there are only N different IPs. This allows to recursively compute $P(j,i)$ for all other cases according to

$$P(j,i) = \sum_{m=0}^{k} \frac{\binom{j-m}{k-m} \cdot \binom{N-j+m}{m}}{\binom{N}{k}} \cdot P(j-m, i-1)\,, \tag{3.1}$$

which simply considers the number of possibilities to obtain $k - m$ old and m new IPs, normalized by the number of possibilities for k different IPs of a tracker response. As a result, we obtain the distribution X of the number of required tracker responses to get all N IPs which is $P(X = i) = P(N, i)$.

An upper bound of the average number of required tracker responses $E[X] = \sum_{i=0}^{\infty} iP(N,i)$ can be approximated [56] using the harmonic number $h_N = \int_0^1 \frac{1-x^N}{1-x} dx$,

$$E[X] \approx \frac{N \cdot h_N}{k}\,, \tag{3.2}$$

which is exact for $k = 1$. For example, to get a snapshot of the distribution of peers over ASes of a swarm with $N = 20,000$ peers, around $n = 20\,\mathrm{requests}$ have to be sent from each of the 219 used PlanetLab nodes. This takes $n \cdot t = 5\,\mathrm{minutes}$. The computation of the number of tracker requests allows to estimate the required number of monitoring nodes and to adjust appropriately the parameters t and n if a time frame of 5 minutes is allowed for capturing the snapshot.

However, it has to be noted that Equation 3.2 only returns the average number of required tracker responses. Checking the percentage of missing IP addresses in our measurements, we observed that only for a small number of swarms some IP addresses are missing. In particular, we checked the percentage of missing IP addresses when observing the distribution of peers over ASes of a swarm which we did for the data sets `Mus`, `Mov`, `Reg`, `Grp`, and `Ele`. Figure 3.1 shows the cumulative distribution function (CDF) of the percentage of missing IP addresses when measuring the distribution of peers over ASes for the movies (`Mov`) and music files (`Mus`). For 97.5 % of all movies (`Mov`) and more than 98.5 % of all music files (`Mus`), all IP addresses in the swarm were captured. For the `Reg` data set, which contains 120 swarms, all IP addresses are available for 118 swarms and in the `Grp` data set we have them for all swarms. A reason for missing IPs is the fact that peers may go offline during the measurement interval of 5 minutes. This has no effect on the numerical values or on the conclusions.

To conclude this section, we describe as a side note one peculiarity we discovered during our measurement study. In our measurements, we found one swarm (`Ele`) for which we discovered only 10 % of the peers. In particular, the tracker returned a swarm size of 400,000 peers, however, we only observed 30,000 IP addresses. We used 219 PlanetLab nodes and requested the tracker every 10 seconds from each machine over 24 hours. Thus, we received more than one million tracker responses with 50 IPs. In that case, we should observe at least around 375,000 different IPs.

There are two possible reasons for this observation. The first one is that the tracker always returns the same IP addresses. This could be the case when locality-awareness mechanisms are implemented by the tracker. However, this is not the case here; the nodes in PlanetLab are distributed world-wide. Thus, it seems reasonable that the random generator or the function which returns a random subset of all peers is wrongly implemented.

The second possible explanation is that the tracker returns wrong information

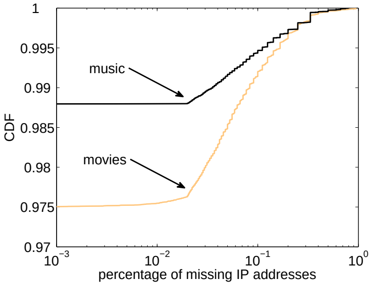


FIGURE 3.1: CDF of the percentage of missing IP addresses.

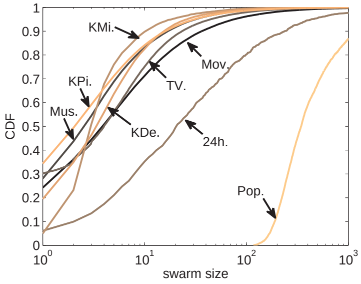


FIGURE 3.2: CDF of the number of total peers in a BitTorrent swarm.

about the number of seeders and leechers in the swarm. Since this tracker hosts only a single file (Ele), we cannot check this hypothesis using other swarms hosted at the same tracker. Still, the second explanation seems more likely to be the case, but we cannot prove it without investigating the source code of this tracker. In both cases, the question arises how an ALTO mechanism can reliably monitor swarms for badly implemented trackers.

3.2 Measurements of Real-World BitTorrent Swarms

In this section, we describe the results of the measurements. We focus on observations where previous studies provide only a general impression or where the results for specific swarm types contradict the accepted knowledge. In particular, we are interested in the characteristics of the swarm size and its temporal development. Additionally, we consider the distribution of peers over ASes and over different countries, the clustering of peers in ASes and the correlation between the number of peers in an AS and its AS degree since these parameters are assumed to have important implications for the viability of locality promoting mechanisms. Finally, we report our findings on content that is popular only in specific geographic regions and summarize our main findings as well as limitations of this study.

3.2.1 Population Sizes in Swarms

First we take a look at the size of the measured swarms. For this purpose, we analyzed the seeder and leecher population of swarms for different content types, e.g., movies, TV shows and music files, which are registered at different BitTorrent index websites.

Figure 3.2 shows the observed swarm sizes for the data sets `TV`, `Pop`, `24h`, `Mov`, `Mus`, `KPi`, `KDe` and `KMi`. The distribution of the number of peers is similar for all data sets except for the `24h` and `Pop` set. An explanation for this divergence is the fact that these two sets feature swarms with specific characteristics due to the popularity of the shared content. While the `Pop` set of swarms contains swarms with highly sought content by definition, it is a reasonable assumption that the recently added files of the `24h` set are also more popular than the average since users are interested in new content which is available for the first time.

The according data for all measurements is given in Table 3.2. It contains the statistics for the total number of observed swarms, the mean value μ and coefficient of variation c_{var} of their sizes in terms of number of peers, the skewness, kurtosis and maximum of the swarm size distribution as well as the 95th percentile q_{95} both as an absolute value and normalized by the mean swarm size. Finally, the fraction of swarms π_{80} that contain 80% of the peers and the correlation $C(S, L)$ between the number of seeders and leechers in all swarms of the whole data set is shown.

The first observation we make is that the swarm size depends on the shared content. This is in line with the observations for video file swarms from [72]. The swarms which distribute movies are the largest on average whereas smaller music files are shared by less peers on average. This can be attributed to the

TABLE 3.2: Statistics on the number of peers in a swarm.

ID	swarms	μ	c_{var}	skew.	kurtosis	max.	q_{95}	q_{95}/μ	π_{80}	$C(S, L)$
Mov	126,049	25.46	8.48	51.89	3,573.01	20,079	76	2.98	0.13	0.84
TV	63,867	15.53	6.47	29.45	1,246.99	7,276	45	2.88	0.17	0.71
Mus	135,679	9.76	4.24	28.43	1,432.57	3,813	32	3.28	0.25	0.61
KPi	1,682,355	11.12	13.42	216.52	69,248.60	72,988	31	2.79	0.18	0.85
KMi	4,514	6.99	3.17	19.78	535.82	763	19	2.72	0.45	0.53
KDe	11,759	9.73	4.64	22.90	663.79	1,883	27	2.78	0.31	0.65
Pop	4,463	691.14	2.08	9.87	144.06	30,691	2,068	2.99	0.45	0.73
24h	1,048	146.68	5.37	17.20	386.37	19,748	435	2.97	0.12	0.65

fact that larger files take longer to download, leading to a longer online time of peers and therefore a higher population in the swarm. This should be offset by the resulting additional upload bandwidth offered to the swarm. However, it can be shown analytically, e.g., by adapting the analysis of [43], that download times do increase in such swarms. A further reason for the larger swarm sizes could be that movie content is more popular than music.

The skewness[4] and the kurtosis of the swarm sizes provide further insights into the distribution of the number of peers in the different data sets. They characterize to which degree some very large swarms are contained in the data sets. The column q_{95} in Table 3.2 contains the 95th percentile, which also characterizes the distribution of the swarm sizes. In particular, it shows the swarm size which is reached or exceeded by 5 % of the swarms in the data set.

Regarding the different data sets, the coefficient of variation of the swarm size is in the same range, with the exception of the Khirman set [69] of PirateBay swarms (`KPi`). This set also differs significantly in terms of skewness, kurtosis and maximum swarm size. Although we cannot judge the source of this discrepancy with our data and the other data sets from Khirman, we still observe that at least the 95th percentile normalized by the mean value is comparable to the corresponding values for the other data sets.

Another general observation is that the Pareto principle holds for most of the evaluated data sets: the π_{80} value, i.e., the fraction of top swarms that contain 80 % of all peers in all swarms of the set, is around 0.2 for all sets except the top movies and the Khirman data for the Mininova and Demonoid sites. This means that 80 % of the peers belong to 20 % of the swarms. It is plausible that the most popular content as covered by the `Pop` data set do not show this Pareto property, since the different files here are equally popular and represent only a very specific part of the total shared content.

Finally, there is a strong correlation $C(S, L)$ between the number of seed-

[4] To calculate skewness and kurtosis of a set of n samples $x_1, \ldots, x_n$, we transform the samples to $z_i = \frac{x_i - \bar{x}}{s}$, where $\bar{x}$ is the average and s is the standard deviation of the samples x_i. Skewness and kurtosis are then defined as the third and forth empirical moment of the samples z_i, respectively, where the j-th empirical moment is $m_j = \frac{1}{n}\sum_{i=1}^{n} z_i^j$.

ers and the number of leechers in a swarm. This is intuitively clear since more leechers mean a larger number of potential seeders, and swarms with only a few seeders are normally not popular due to long download times.

From these observations we draw some conclusions on how they could impact a locality-aware mechanism. The type of shared content has an impact on the swarm size and therefore potentially on the effectiveness of different locality promoting solutions. We will see in the next sections that this is also true for the topological characteristics of a swarm, which also depend on the shared content. In general, the swarm size distribution is heterogeneous with a Pareto-like distribution of the total peer population on the different swarms. Also, recently released and popular content leads to much larger swarms in comparison to the average values.

In addition, there is a significant amount of rather small swarms containing less than 40 peers. With typical BitTorrent client parameters, each peer in such a swarm will know all other peers because it tries to have at least 40 neighbors. The result is thus a fully meshed swarm. Consequently, accepted solutions using Biased Neighbor Selection (BNS) as introduced in [51], where peers close in the topology are preferred as neighbors, will probably have a low impact on these swarms since there is no choice to be made in the neighbor selection.

On the other hand, the share of traffic that can be influenced by targeting only the comparably few top swarms, including new and popular content, is significant (around 80 %, the corresponding estimation is presented in Section 3.2.4). The effort to do so is possibly much lower than when trying to cover all or at least most of the swarms because algorithms do not need to cope with special characteristics of small swarms. To optimize the monitoring of swarms, in order to find these candidate swarms, it might help to just keep track of the seeder population since it is strongly correlated to the number of leechers and thus the total population of a swarm. These statements are not meant to be true in general and for every mechanism, they rather show examples how the data provided in this section can be important for the assessment of locality-aware mechanisms.

3.2.2 Time-Dynamics within a Swarm

In this section, we investigate in which way the population of a swarm varies over time. The evolution of BitTorrent swarm populations during the whole life time of a swarm has already been analyzed in literature, e.g., in [42] or [47]. However, we focus here on a shorter time scale and investigate how fast the population typically grows or diminishes during our measurement period lasting 36 hours. In addition, we analyze which fraction of swarms is subject to diurnal fluctuations and how pronounced these fluctuations are.

For this section, we focus on the data set `TV` since this contains more than 60,000 swarms and their temporal evolution. In order to illustrate some examples, we also consider the `Grp` data set. However, this set contains only 16 swarms and is therefore less suitable for statistical analysis. For all other data sets we do not have measurements about the temporal evolution of the swarm sizes.

Increasing, Constant, and Decreasing Swarms

While it may be efficient to promote locality in a swarm that was measured as being large at a given time instant, it may be less efficient when the swarm shrinks quickly after that snapshot. To gain insights into the time-dependent behavior of swarms, we measured 96 samples of the swarm sizes $n_i(s)$, $i \in \{1, 96\}$ for every swarm s of the data set `TV`. The samples were equally distributed over 36 hours. For all swarms, we calculate the average swarm size $\mu(s)$, the standard deviation $\sigma(s)$, and the coefficient of variation $c_{var}(s)$ of the 96 samples $n_i(s)$. In addition, we define the span of a swarm during the measurement period as $\Delta(s) = \max_i(n_i(s)) - \min_i(n_i(s))$. This metric represents the largest variation of the swarm population we observed in terms of peers. We call all swarms s with $\Delta(s) = 0$ constant swarms. The remaining swarms are increasing if their minimum value n_i has a lower index i than their maximum value. Otherwise, we denote them as decreasing.

We make the following observations in the data set `TV`: All three groups (constant, increasing, and decreasing) contain almost the same fraction of swarms

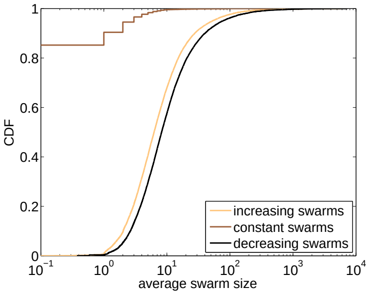


FIGURE 3.3: CDF of the swarm sizes for increasing, constant, and decreasing swarms (measurement setup TV).

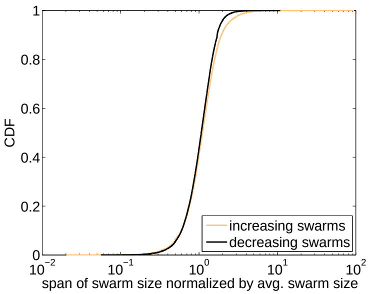


FIGURE 3.4: CDF of the span of swarm sizes for measurement experiment TV.

(33.81 %, 32.88 %, and 33.31 %, respectively). However, the constant swarms are all very small (cf. Figure 3.3). In addition, there is no significant difference between the CDFs of the sizes of increasing and decreasing swarms which is reasonable since the fact that a swarm is growing or shrinking is not correlated with its current size.

Figure 3.4 shows CDFs for the span $\Delta(s)$ of the swarms normalized by the average swarm size over the 36 hour time period. We observe that for only 10 % of the swarms their span is below 60 % of their average size and for 50 % of the swarms it is higher than the average swarm size. Furthermore, the span $\Delta(s) < 2 \cdot \mu(s)$ for almost all decreasing swarms and $\Delta(s) < 5 \cdot \mu(s)$ for almost all increasing swarms. This difference can be explained by flash-crowd arrivals in some new and very popular swarms which lead to a large increase in peer populations. In summary, we conclude that – already in a time frame of 36 hours, which is rather small compared to the lifetime of a swarm – the swarm populations can vary heavily. It is important to keep that in mind if parameter settings of locality-aware mechanisms need to be adjusted based on current swarm populations.

Next, we study how the time dynamics correlate with the swarm sizes, i.e.,

TABLE 3.3: Coefficients of correlation ρ of the average swarm size $\mu(s)$ and the variation ($\Delta(s)$, $\sigma(s)$, and $c(s)$) for increasing and decreasing swarms.

	$\rho(\mu(s), \Delta(s))$	$\rho(\mu(s), \sigma(s))$	$\rho(\mu(s), c(s))$
increasing swarms	0.695	0.668	-0.038
decreasing swarms	0.672	0.653	-0.060

whether large swarms are subject to large variations or not. To this end, we calculate the coefficient of correlation ρ of the average swarm size to three values representing the variation: the span $\Delta(s)$, the standard deviation $\sigma(s)$ and the coefficient of variation $c(s)$ (cf. Table 3.3). We observe that the span $\Delta(s)$ and the standard deviation $\sigma(s)$ is strongly correlated to the average swarm size for increasing and decreasing swarms ($\rho > 0.65$). However, these correlations vanish if we take the coefficient of variation $c(s)$ instead of the standard deviation $\sigma(s)$. That means that larger swarms tend to have larger variations of the swarm population which is not very surprising. However, the variation normalized by the average size $\mu(s)$, i.e., the relative change in the swarm population is not correlated with the swarm size. Hence, large swarms do not grow or shrink disproportionally fast.

Finally, we illustrate the correlation between the average swarm size $\mu(s)$ and the coefficient of variation $c(s)$ of the swarm size with a scatter plot in Figure 3.5 for the swarms of the `TV` data set, sorted by swarm size. The coefficient of variation $c(s)$ for most of the swarms s is between 0 and 1, on average it is 0.2795. In addition, we observe a set of swarms (around $1\,\%$ of the measured swarms) where $c(s)$ is very close to 1. The reason for this result are frequent jumps of the swarm size (reported at the PirateBay website) between 0 and the actual swarm size which we attribute to an error in this website. However, this should have only a minor impact on our results since only $1\,\%$ of the `TV` data set shows this behavior and the `TV` data set is the only one we measured by parsing the website (cf. Table 3.1). In order to show that the peculiar shape of the scatter plot is not owed to chance, we present a short mathematical derivation for the theoretical minimum of the coefficient of variation

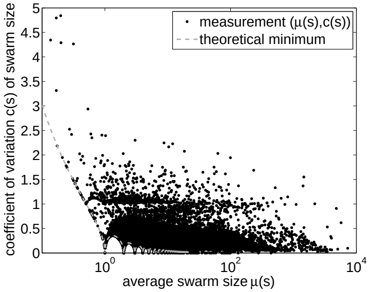


FIGURE 3.5: Coefficient of variation c_i of the size of swarm i vs. average size of swarm i for measurement experiment TV.

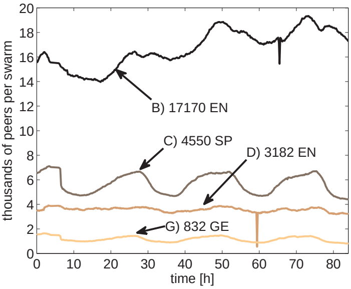


FIGURE 3.6: Total swarm size of exemplary swarms (measurement setup Grp) as defined in Table 3.5.

$c(s)$. Since we capture $R = 96$ samples of the size of a swarm s for the TV experiment, the minimum standard deviation $\sigma(s)$ for a given average swarm size $\mu(s) \in [a; a+1[$ is obtained when we measure k times a size of a and $R - k$ times a size of $a + 1$ (for $a \in \mathbb{N}$). Thus, it is $\mu(s) = \frac{ka+(R-k)(a+1)}{R}$ and $\sigma(s) = \sqrt{\frac{ka^2+(R-k)(a+1)^2}{R} - \mu(s)^2} = \frac{1}{R}\sqrt{(R-k)k}$ which explains the shape of the theoretical minimum for the measurements.

Diurnal Fluctuations

Now we take a closer look at the fluctuations. The evolution of the size of four example swarms, which are taken from the set summarized in Table 3.5, is depicted in Figure 3.6. The selection of these swarms allows us to show principal differences between swarms even if they share the same type of content. Here, the swarm population over time is shown, with the base unit of the y-axis being 10^3 peers.

We observe that there are variations in the population of each swarm, as well as quantitative and qualitative differences in these variations between the swarms. While swarm D), which is sharing a movie in English, shows only small changes

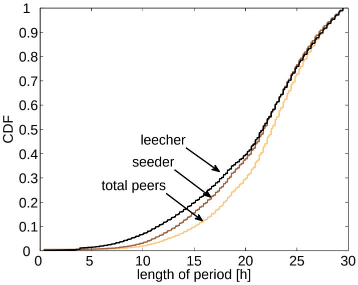


FIGURE 3.7: Length of period for TV by calculating the periodicity transform using the *M-Best Algorithm* [36].

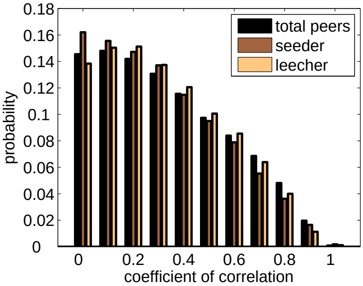


FIGURE 3.8: Autocorrelation to the best period for TV.

in its peer population, the size of swarm C) exhibits a periodic behavior. We attribute this to the fact that in this swarm, a movie in Spanish is distributed. In order to check how many peers of that swarm are located in Spain we use the GeoIP service of MaxMind [89] to map the IP addresses to countries. In fact, more than 94 % of the peers are from Spain and only about 2 % from South America. Therefore, the swarm population increases during the daytime in this region and decreases again afterwards. Swarm G), sharing a German movie, shows a similar characteristic. The fluctuations are not as clearly visible as for swarm C), but in relation to the average swarm size, the population of swarm G) fluctuates to roughly the same degree as swarm C).

The development of the peer population of swarm B) is a superposition of a continually increasing popularity and a 24 hour cycle like for swarms C) and G). While swarm D) distributes content that seems not to be preferred regionally, the movie shared in swarm B) seems to be more popular in a specific part of the world.

We now want to determine the amount of swarms that show a diurnal behavior similar to swarms B), C) and G), in order to judge the relevance of this effect for the performance evaluation of locality-awareness mechanisms. To that end, we use a method called periodicity transform which automatically detects periodicities for a given data set. In particular, we rely on the 'M-best' algorithm as introduced in [36] that returns a list of the $M = 10$ best periodicities. From the M best periodicities that are $\{\tau_i : 1 \leq i \leq M\}$ we calculate the autocorrelation ρ_i at lag τ_i and select the best period of duration τ_k with maximum, positive autocorrelation ρ_k, i.e. $k = \arg(\max\{\rho_i : 1 \leq i \leq M\})$. We also tried the other methods described in [36], but the M-best algorithm delivered the best results in finding periodicities of around 24 h.

Figure 3.7 shows the CDF of the length of the 'best' period for the number of seeders, the number of leechers, and the entire swarm size for the `TV` data set. It can be seen that the three different curves show a similar behavior. In particular, the curves for the number of leechers and seeders are almost identical, showing that the leechers mainly determine the diurnal behavior. Furthermore, we observe that roughly for 60 % of the swarms the 'best' period is between 21 h and 27 h. There is no discontinuity in the CDF at 24 h since the M-best analysis is not able to completely ignore all other effects changing peer populations such as increasing or decreasing popularity of the content or flash-crowd arrivals.

Figure 3.8 shows the autocorrelation ρ_k to the best period of duration τ_k. Again, the three different curves are quite similar. We observe that from the swarms in the `TV` data set only 8.36 % show a strong correlation $\rho_k > 0.7$. As a summary of the time-dynamics analysis, we see that for roughly 5.7 % of the swarms a day-night behavior can be observed. To be more precise, for these swarms the autocorrelation is larger than 0.7 for the best period, while the duration of the period is about 1 day, i.e. between 21 hours and 27 hours.

3.2.3 Distribution of Peers over ASes

One important performance indicator for locality-aware mechanisms, typically used in related studies [17,20,51,80], is the amount of inter-ISP traffic which can be saved by their application. In such investigations, the distribution of peers over ASes can play a major role for the potential savings [20, 80]. As a consequence,

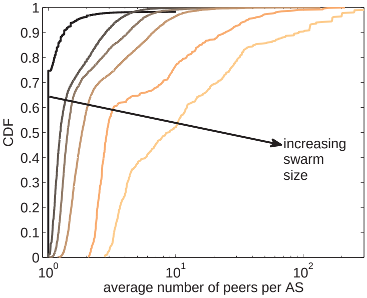


FIGURE 3.9: CDF of average number of peers per observed AS. Swarms (`Mov`) are grouped according to their size, cf. Table 3.4.

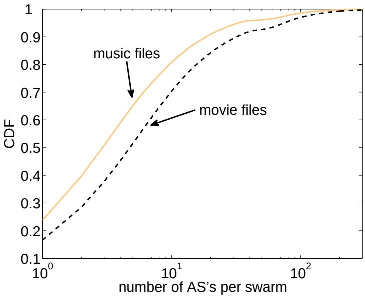


FIGURE 3.10: Number of ASes per swarm (`Mus`,`Mov`).

we consider in this section statistics on the number of ASes which contain peers participating in the same swarm and on the average number of peers located in one AS. For this purpose, we use the `Mov` and `Mus` data sets since they contain a large number of swarms together with the IP addresses of the peers so that we can map them to ASes. The distribution of peers over ASes of swarms sharing regional content (`Reg` data set) is presented in Section 3.2.7.

We present the CDFs for the average number of peers per AS for swarms of the `Mov` data set in Figure 3.9. Note that the x-axis is scaled logarithmically. The swarms are grouped according to their average size as shown in Table 3.4 together with the relative size of each group. We observe that for an increasing mean swarm size, the average number of peers per AS grows. However, this value is still small even for the largest swarms. This is in line with literature [70, 72, 80, 81]. Considering the `Mus` data set leads to the same conclusions. In

TABLE 3.4: Percentage of swarms grouped according to their size for movie files (`Mov`).

swarm size	[0; 25[	[25; 50[	[50; 100[	[100; 500[	[500; 1e3[	[1e3; ∞[
fraction of swarms	0.8580	0.0703	0.0294	0.0347	0.0040	0.0036

fact, the average number of peers per AS is even smaller for these swarms. The concrete numbers corresponding to Table 3.4 and Figure 3.9 can be found in our technical report [68]. In Section 3.2.6, where we analyze the distribution of peers over countries, we show CDFs also for the maximum number of peers per AS (cf. Figure 3.16).

Another important characteristic of a swarm is the absolute number of ASes because swarms that are distributed over fewer ASes but with more peers per AS can likely utilize locality promotion mechanisms more efficiently. To this end, we consider the movie files (`Mov`) as well as the music files (`Mus`). Figure 3.10 shows the CDF of the number of ASes per swarm for both data sets. Since there are more peers involved in swarms offering movie contents, there are also more different ASes involved than in swarms providing music files. On average, there are 65 % more ASes involved in movie swarms than in music swarms. In particular, if the CDF of the number of ASes for movie swarms is normalized by a factor of 1.65, it is nearly identical to the CDF for music swarms. The maximum number of observed ASes is 1,744 for movie swarms and 809 for music swarms, respectively. We will explore the distribution of peers over ASes in more depth in Section 3.3 where we provide a model for the probability that a peer belongs to a certain AS.

3.2.4 AS Clustering of Peers

A fundamental pre-condition of keeping BitTorrent traffic within a given AS is that several peers sharing the same file are present in that AS. Therefore, we study in this section which fraction of swarms actually have the possibility to exchange data with local neighbors. To that end, we count the number of peers in every swarm which are located in an AS with at least α peers of the swarm. To obtain the AS clustering of peers δ_α of a swarm, we normalize this number by the swarm size. In other words, δ_α represents the fraction of peers in the swarm having at least $\alpha - 1$ other peers of the same swarm in their AS. In Figure 3.11 we show CDFs of the AS clustering for $\alpha \in \{3, 4, 5\}$. We observe that in roughly

89 % of the Mus swarms no AS exists where at least 3 peers are present ($\delta_3 = 0$) and only in about 3.5 % of the swarms the majority of peers ($\delta_3 = 0.5$) can be clustered in their ASes. Considering the movie files (Mov), the probability to find clusters of peers within an AS is higher since these swarms are larger. Still, in about 88 % of the movie swarms there is no AS with at least 5 peers. Thus, locality-awareness will only be useful in a rather small fraction of the swarms. However, this statement does not fully address the question about which fraction of the total BitTorrent *traffic* can be influenced by locality-awareness.

To answer this question, we first study the total amount of traffic produced by the swarms in the Mus and Mov data set. For that purpose, we take the number of peers in a swarm as an indicator of how much traffic a swarm produces in relation the other swarms and assume that peer access capacities are not correlated with the swarm sizes. Therefore, they can be neglected in our simple approximation. Figure 3.12 presents the cumulative estimates ('music traffic T_0', 'movie traffic T_0') for the top x % of the largest swarms normalized by the total amount of traffic. The figure reveals that 10 % of the swarms of the Mov data set contain 80 % of the peers and are consequently responsible for the same fraction of the total traffic according to the aforementioned assumptions. If we weight the number of peers in a swarm with the size of the exchanged file ('traffic T_f', legend: 'with file sizes') to estimate the amount of traffic, we obtain almost the same results as for taking just the number of peers ('w/o file sizes'). This is in particular true for the movie traffic. For the music files the difference is small.

Next, we develop a very simple and optimistic approximation for the potential of locality-awareness. This approximation is based on the results for the AS clustering δ_α. For each swarm, we calculate δ_2, i.e., the fraction of peers in the swarm which are not the sole peer in their AS. We assume an ideal locality algorithm which achieves that those peers produce no inter-ISP traffic and neglect which peers are seeders and leechers and possible performance degradations for simplicity reasons. Then, δ_2 is the fraction of 'potentially local traffic' of that swarm. This value weighted by the total resulting traffic of the swarm for the music and movie files is presented in Figure 3.12. The figure shows that it has al-

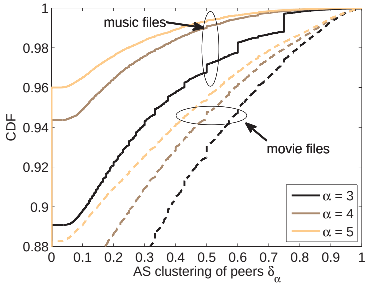


FIGURE 3.11: CDF of AS clustering of peers δ_α for music (Mus) and movie (Mov) files.

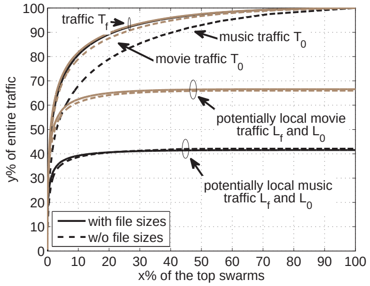


FIGURE 3.12: Total and potentially local traffic of the top x % of the swarms (Mus,Mov) with and w/o considering the size of the exchanged files.

most no impact on this approximation whether we take into account the file sizes ('L_f') or not ('L_0') for the calculation of the total traffic a swarm produces. Furthermore, the figure confirms our finding that locality-awareness is only useful in a small subset of all swarms. However, it shows in addition that the potential savings of inter-ISP traffic are quite larger in the big swarms which are responsible for the vast majority of BitTorrent traffic. Therefore, the overall optimization potential of locality-awareness is about 65 % for the movie files (Mov) and roughly 40 % for the music files (Mus). In other words, around 35 % (60 %) of the overall movie (music) traffic is produced by peers which are the only one in the AS. Therefore, no locality-awareness mechanism can avoid this inter-ISP traffic. In summary, we conclude from this section that the overall optimization potential for locality-awareness is large even if the mechanisms will only be useful in the top 20 % of the swarms.

3.2.5 Relation of Number of Peers and AS Degree

In this section we investigate to which degree the size of an AS is correlated with the number of peers it contains. For that purpose, we study two metrics

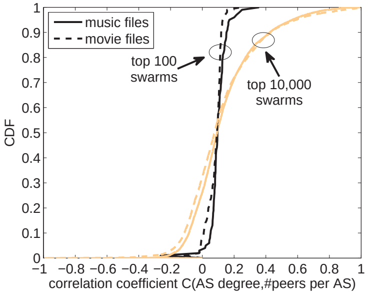


FIGURE 3.13: CDF of the correlation of the AS degree and the number of peers per AS for every swarm (Mus,Mov).

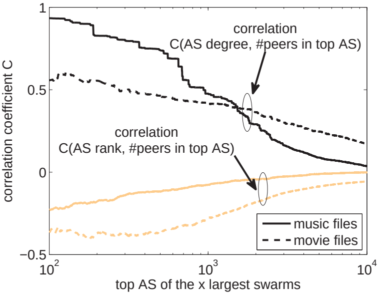


FIGURE 3.14: Correlation of the AS degree and AS rank with the number of peers in the top AS of each swarm (Mus,Mov).

representing the "size" of an AS: the AS rank and the AS degree. Both metrics are provided by CAIDA [92]. The AS degree is defined as the number of ASes to which a given AS is connected. Like in [38] we use the AS degree as an indicator for the size of the AS. To obtain the AS rank of a given AS, CAIDA basically orders all ASes according to their size and defines the AS rank of a given AS as its index in this ordered list. For this investigation we use the Mus and Mov data set since these contain large numbers of swarms and their distribution of peers over ASes.

First, we check the correlation between the total number of peers per AS and the size of the AS. To this end, we calculate the total number of peers in a given AS as the sum of the number of peers in this AS of all swarms in the data set. Then, we correlate the total number of peers per AS with the AS degree and the AS rank obtained from CAIDA. This calculation shows that the total number of peers in an AS is neither correlated to the AS rank nor to the AS degree. The concrete values for the correlation to the AS rank are -0.0962 and -0.0834 for the Mus and Mov data set, respectively. The corresponding values for the correlation to the AS degree are 0.1492 (Mus) and 0.1020 (Mov).

Next, we calculate the correlation of the number of peers per AS with the corresponding AS degree for each swarm. That means, we get one correlation coefficient for each swarm in the data set and plot CDFs of this value for the 100 and 10,000 largest swarms (cf. Figure 3.13). Although some swarms exist in the top 10,000 swarms of both data sets where the correlation is high, most of the swarms do not have this strong correlation. In particular, these swarms are not among the 100 largest swarms. Therefore, we conclude that within a given swarm it is quite unlikely that the number of peers per AS is correlated with the AS degree. A possible explanation for that rather unexpected result is that there is a large number of ASes in every swarm which contain only 1 or 2 peers. Still, these ASes may have a high AS degree which leads to low values for the correlation.

To avoid this influence of the large number of ASes with only a few peers, we now focus on the top AS of every swarm. In this way, we limit our investigation to those ASes with a large number of peers. In Figure 3.14 we calculate the number of peers in the top ASes of the x largest swarms and correlate these x numbers to the corresponding AS degree and AS rank. We observe that the correlation with the AS degree is stronger than the one with the AS rank. Furthermore, the correlation decreases when we increase x, i.e., when we take into account more swarms. In particular, the correlation of the AS degree and the number of peers in the top AS of the 100 largest swarms (`Mus`) is close to 1. That means, for the ASes where the number of peers is large, this number is correlated to the AS degree.

3.2.6 Alternative Metric for Locality: Country Codes

While AS affiliations are a popular metric describing which peers are nearby, other metrics such as the number of IP- or AS-hops, similarity of CDN redirection behavior [59], or geographic proximity can also be used for that purpose. In this section we investigate in which way the results of the previous section are affected if we use a different criterion than the AS affiliation. For feasibility reasons we select the geographic proximity out of the aforementioned example metrics and

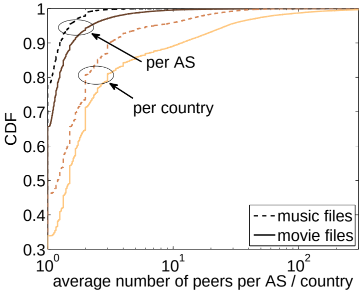


FIGURE 3.15: CDF of average number of peers per observed AS and per country for the Mov and Mus data sets.

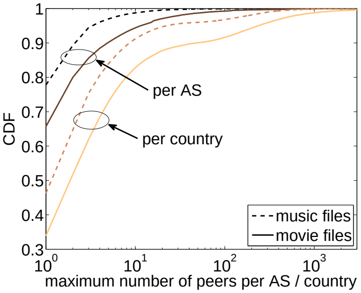


FIGURE 3.16: CDF of the maximum number of peers in a country per swarm (Mus,Mov).

map every IP address to a country code using the MaxMind GeoIP service [89].

First, we compare the number of peers per AS to the number of peers per country. For that purpose, we calculate the average number of peers per AS and per country for every swarm (Mus and Mov) and show CDFs over all swarms in Figure 3.15. We observe that the number of peers per country is higher than per AS. For the Mus data set the mean number of peers per country (averaged over all swarms) is about 2.3 times higher than the mean number of peers per AS. For the Mov data set the same relation is about 6.2. This seems reasonable since most countries contain several ASes. Figure 3.16 is similar to Figure 3.15 but presents the maximum number of peers per AS and per country instead of the average numbers. That means that we select from each swarm that AS and that country with the highest number of peers. Again, we observe that the number of peers per AS is lower than per country. Second, we investigate the number of countries per swarm in analogy to Figure 3.10, which is based on the AS affiliations. The corresponding figure for the country codes is very similar to Figure 3.10, and we therefore omit it. The only difference is the one already observed in Figure 3.15 that there are on average more peers per country than per AS.

Hence, using country codes instead of AS affiliations leads to a coarser classification of peers and consequently higher numbers of peers in the same class. Therefore, keeping traffic local in a given country should be easier since it is more likely to find local neighbors than in the same AS.

3.2.7 Characteristics of Regional Swarms

We have already seen the effect regional content has on the evolution of the swarm size over time. We now take a closer look at the topological characteristics of swarms sharing this content. These swarms are contained in the data sets `Reg` and `Grp`. The `Grp` data set comprises 16 example swarms of different average sizes distributing movies in German, Spanish, Chinese or English (cf. Table 3.5). For these swarms, we analyze the number of ASes and the top AS fraction of the swarm, i.e., the maximum number of peers in an AS of that swarm normalized by the swarm size (cf. Figure 3.17). In this figure, the swarm size (given in Table 3.5) is indicated by different colors on a logarithmic scale. Swarms sharing regional content have a high top AS fraction (20 to 50 %) and are spread over comparably few ASes. In contrast, swarms sharing internationally interesting content, i.e., in English, have a small top AS fraction (below 10 %) and are spread over more ASes.

Swarm D is an exception here. It shows the highest skewness in terms of number of peers per AS compared to the other swarms. In particular, 30 % of the peers belong to the same AS with the AS number 30058. A closer look reveals that the company responsible for this AS offers its customers to rent dedicated or virtual servers located in this AS. This permits a single customer to run a large number

TABLE 3.5: Individually measured swarms over time (`Grp`) using the following notion: *ID) average swarm size & language.*

A) 21,351 EN	B) 17,170 EN	C) 4,550 SP	D) 3,182 EN
E) 1,390 SP	F) 972 GE	G) 832 GE	H) 626 GE
I) 579 SP	J) 479 EN	K) 473 GE	L) 351 GE
M) 289 GE	N) 258 EN	O) 217 SP	P) 81 CN

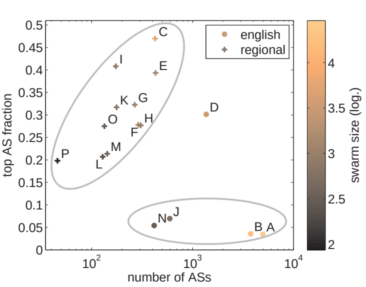


FIGURE 3.17: Scatter plot of the number of ASes and the top AS fraction (Grp data set, cf. Table 3.5).

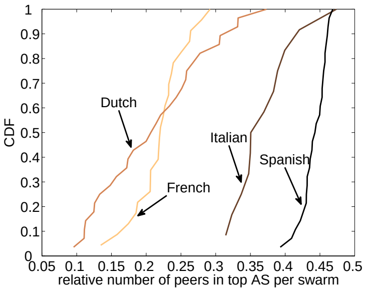


FIGURE 3.18: Relative number of peers in a swarm’s top AS (Reg).

of peers on different virtual nodes which could be used to insert fake peers in the swarm in order to disturb the distribution process. This might be an explanation of the high fraction of peers in swarm D in AS 30058.

Next, we move from the Grp data set with 16 example swarms to the Reg data set containing 120 swarms exchanging regional movies observed at the index server PirateBay.org in May 2009. This set is more suitable for statistical analysis since the number of swarms is higher and the swarms are not selected by hand as it is the case for Grp. The fact that users are interested in regional content leads to a high top AS fraction, which is the relative number of peers in a swarm’s top AS. This is especially true for Spanish content, see Figure 3.18. Here, the top AS of each swarm in the Reg set is used for comparison, i.e., the AS containing most peers from a swarm. In this graph, a CDF of the relative share of peers that are located in these ASes is plotted for swarms with Dutch, French, Italian and Spanish content.

While in all cases there are at least 10 % of the total swarm population in the top AS, this share is between 40 and 48 % for the Spanish content, implying a high degree of peer grouping. To judge whether this phenomenon only exists for a single AS, we evaluated also the second to fifth largest ASes of the swarms in

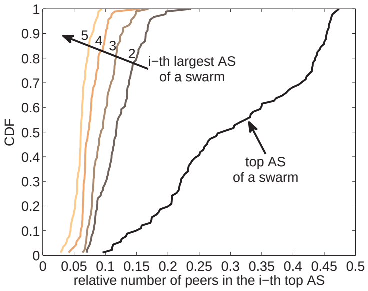


FIGURE 3.19: Relative number of peers in i-th top AS (Reg).

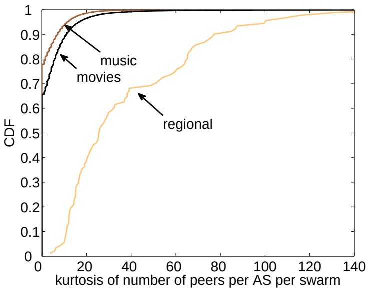


FIGURE 3.20: Kurtosis of number of peers per AS (Reg, Mus, Mov).

the Reg data set, cf. Figure 3.19. It appears that the top AS of a swarm contains significantly more peers than the other ASes, although these are still holding around 5 % of the total swarm population.

We affirm this result by comparing the kurtosis, i.e., the fourth moment of a distribution that indicates statistical peaks, of the number of peers per AS for the swarms in the Reg, the Mus and Mov sets. The results are shown in form of a CDF in Figure 3.20.

The regional swarms show a much higher kurtosis than the two larger and more general sets. This leads us to the conclusion that the concentration of a larger fraction of the swarm in the same AS is much more common in regional swarms. This means that the regional interest in a shared file can play a significant role in the suitability of the according swarm for locality promotion, something previously underestimated. In particular, the high kurtosis values for a certain fraction of swarms providing music or movie files in Figure 3.20 indicates that this phenomenon of regional interests with many peers in the top AS can be observed for any kind of content.

3.2.8 Summary of Measurement Results

From the results presented above, we make the following main observations for the characterization of BitTorrent swarms and their distribution in the Internet.

Considering the swarm statistics according to the offered content (i.e., TV shows, movies and music) we observe that the larger the offered content is in terms of data volume, the larger the average and maximum number of peers is in such a swarm, as already shown in less detail in [72]. Additionally, our results show that the distribution of peers among the swarms follows the Pareto principle for the different measurement sets (1), (4) and (5), which contain random files. This means that 80 % of all peers belong roughly to the top 20 % swarms for all media types. The Pareto principle cannot be observed for measurement sets (2), (3), and (6) since we only consider popular or recently published content there. These recently published torrents are highly popular. This is reasonable since users are typically interested in new contents, recently broadcasted movies etc.

We studied the distribution of peers over ASes of the swarms and showed that the average number of peers per AS is small for most of the swarms. However, the distribution of peers over ASes is skewed so that a high fraction of the peers is contained in the few top ASes of the swarm. Previous studies, e.g., [20, 80], revealed that this can have a strong impact on the performance of traffic optimizations schemes, especially for swarms sharing regional content, where the skewness in the peer distribution is higher. Hence, quantitative characterizations (cf. Section 3.3) of the distribution of peers over ASes are required for a meaningful performance evaluation of traffic optimization schemes.

In addition, our measurements show that the fraction of swarms with ASes where more than 5 peers are located in at least one AS is quite small. Nevertheless, the optimization potential of locality-aware mechanisms remains high since peers in the large swarms, which produce the majority of the traffic, can be clustered in their AS. As a consequence, it would be an option to concentrate traffic optimization efforts on the relatively low number of swarms with larger content and high popularity because the potential gains are much higher than for small swarms. Not only does a larger content lead to more traffic, but also the possibilities for locality promotion are more numerous in larger swarms, where there are more peers in one AS in general.

When the classification of peers is done on the basis of the country code instead of the AS affiliation, we observe that more peers are in the same class and therefore it is easier to keep traffic within that class of peers. Finally, the measurements reveal that for a very small number of swarms (which are not the large ones) the number of peers in an AS is correlated to the AS degree.

3.2.9 Limitations of the Measurement Study

There are some limitations of our measurement study. We describe them here so that they can be taken into account when using our results. First, we studied only swarms which use a tracker to request an initial set of peers and no tracker-less swarms. Second, our measurements rely on the assumption that the information obtained from the websites and the trackers is correct. Furthermore, we did not try to contact the peers we received from the trackers. Therefore, it is possible that some company inserted fake peers in order to disturb the distribution progress which would result in a smaller number of peers actively participating in a swarm than the one we measured. Third, we used different measurement methods for different data sets because we refined our measurement techniques during the course of this work. This has two consequences: (1) not all types of data are available for all data sets (namely the distribution of peers over ASes and measurements over time) and (2) the results might be influenced by the used measurement technique. Overall, we argue that these limitations have only a minor impact on the presented results. To support this we cross-checked the results using all data sets for which the corresponding type of measurement was available, provided explanations of differing results, and compared our results to the ones described in literature.

3.3 Statistical Characterizations of BitTorrent Swarms

Based on the measurements presented in Section 3.2 we develop a set of characterizations for BitTorrent swarms which can be used for performance evaluations of locality-awareness solutions for BitTorrent. Namely, we model the distribution of peers of a single swarm over ASes and fit the swarm population, the number of ASes over which a swarm is distributed, the fraction of the swarm located in the top AS, and the size of the shared file with stochastic distributions for the data sets `Mus`, `Mov`, and `Reg`. Finally, we present the correlation of these values as multivariate correlation matrices.

3.3.1 Power-Law of the Distribution of Peers over ASes

As we have seen from the measurement results presented in Section 3.2, one key aspect for modelling BitTorrent swarms is the skewed peer distribution. In this section, we present a simple model which returns the probability $P(k)$ that a peer belongs to the k-th largest AS within a swarm consisting of n different ASes. In particular, we investigate whether the peer distribution among the different ASes follows a power-law, i.e.,

$$P(k) = a/k^b + c\,. \tag{3.3}$$

Therefore, we consider all swarms $\mathfrak{I}_n$ consisting of exactly n different ASes from `Mus` and the `Mov` data set, respectively. For each swarm $i \in \mathfrak{I}_n$, we measure the ratio $\widetilde{P}_i(k)$ of peers belonging to the k-th largest AS in swarm i for $k = 1, 2, \cdots, n$. Then, we compute the average ratio $\widetilde{P}(k)$ over all swarms, yielding at

$$\widetilde{P}(k) = \frac{1}{|\mathfrak{I}_n|} \sum_{i \in \mathfrak{I}_n} \widetilde{P}_i(k)\,. \tag{3.4}$$

Figure 3.21 shows the measured ratio $\widetilde{P}_i(k)$ of peers belonging to the k-th largest AS within a swarm consisting of $n = 40$ different ASes. All swarms

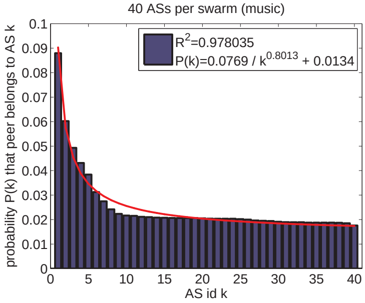


FIGURE 3.21: Comparison of the measured ratio $\widetilde{P}(k)$ and the theoretical probability $P(k)$.

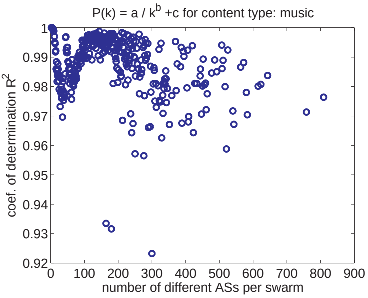


FIGURE 3.22: Goodness-of-fit between the measurement data and the power-law model, cf. Equation 3.5.

consisting of exactly n different ASes are considered from the Mus data set. The observed ratio $\widetilde{P}_i(k)$ is then compared with the power-law model function as defined in Equation 3.3. The parameters a, b, c of this 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 data. Optimal in this case means to find the unknown parameters a, b, c in Equation 3.3 such that the mean squared error is minimized. As a result, we obtain $P(k) = 0.0769/k^{0.8013} + 0.0134$ which is plotted as solid curve. Figure 3.21 indicates that the power-law describes quite well the peer distribution among ASes.

The goodness-of-fit for the model function $P(k)$ is expressed by means of the coefficient of determination R^2. A value close to one means a perfect match between the model function and the measured data. For the measurements given in Figure 3.21 and the obtained model function, the coefficient of determination is $R^2 = 0.978035$ indicating the good match in a statistical way. In our case, the coefficient of determination can be computed as follows

$$R^2 = 1 - \frac{\sum_{k=1}^{n} \left(\widetilde{P}(k) - P(k)\right)^2}{\sum_{k=1}^{n} \left(\widetilde{P}(k) - 1/n\right)^2} \,. \tag{3.5}$$

In the following, we have computed the optimal parameters of the power-law function as defined in Equation 3.3 for all swarms consisting of exactly n different ASes. Again, the coefficient of determination R^2 is used to measure the goodness-of-fit. Figure 3.22 shows a scatter plot of the number n of different ASes in a swarm vs. R^2 for the `Mus` data set. The maximum number of observed ASes is 1,744 for movie swarms and 809 for music swarms. As we can see, the match between the measurement data and the power-law model function is very good and the coefficient of determination is above 0.9. In [68], the power-law describing the distribution of peers over ASes of BitTorrent swarms was also shown for the `Mov` data set. In order to provide a model for BitTorrent swarms, the file size, the size of a swarm, and the number of ASes per swarm is required in addition to the parameters of the power-law model. This will be discussed in the following.

3.3.2 Additional Parameters of BitTorrent Swarms

In order to provide input for the evaluation of locality-awareness mechanisms under more realistic conditions, we introduce statistical characterizations for music files, movie files, and files of regional interest based on the measurements for the `Mus`, `Mov`, and `Reg` data sets, respectively. The considered features of BitTorrent swarms relevant for traffic optimization comprise (a) the size of a swarm, (b) the number of ASes per swarm, (c) the top AS fraction, and (d) the size of the provided file in the swarm.

Tables 3.6, 3.7, and 3.8 show the distribution model of these features f, the mean value $\mu(f)$, the coefficient of variation $c(f)$, and the corresponding model parameters. For the `Mov` and `Mus` data sets, we excluded swarms with less than 10

TABLE 3.6: Characterizations for music swarms with at least 10 peers.

feature f	$\mu(f)$	$c(f)$	model	model parameters		R^2
swarm size	46.15	2.66	log-normal	$\mu = 3.18$	$\sigma = 0.89$	0.96
#ASs per swarm	28.31	1.39	log-normal	$\mu = 2.97$	$\sigma = 0.74$	0.98
top AS fraction	0.13	0.65	log-normal	$\mu = -2.19$	$\sigma = 0.54$	1.00
file size	218.04	2.05	log-normal	$\mu = 4.53$	$\sigma = 1.40$	0.97

TABLE 3.7: Characterizations for movie swarms with at least 10 peers.

feature f	$\mu(f)$	$c(f)$	model	model parameters		R^2
swarm size	85.34	5.64	log-normal	$\mu = 3.42$	$\sigma = 1.06$	0.97
#ASs per swarm	33.67	1.95	log-normal	$\mu = 3.01$	$\sigma = 0.86$	0.98
top AS fraction	0.18	0.84	log-normal	$\mu = -1.98$	$\sigma = 0.75$	1.00
file size	887.05	0.76	Gamma	$a = 1.91$	$b = 463.3$	0.86
file size (impr.)	975.74	0.97	Weibull	$\lambda = 985.97$	$k = 1.03$	0.99

TABLE 3.8: Characterizations for regional swarms with at least 1 peers.

feature f	$\mu(f)$	$c(f)$	model	model parameters		R^2
swarm size	1350.86	1.39	log-normal	$\mu = 6.60$	$\sigma = 1.04$	0.92
#ASs per swarm	77.45	0.54	Gamma	$a = 3.58$	$b = 21.65$	1.00
top AS fraction	0.31	0.38	Gamma	$a = 6.17$	$b = 0.05$	0.97
file size	1367.81	0.81	log-normal	$\mu = 7.00$	$\sigma = 0.60$	0.83

peers from our consideration since most of the BitTorrent users (around 80 %, cf. Figure 3.2) do not belong to these swarms and locality-awareness is expected to have only a very small impact in these swarms (cf. Section 3.2.4). The Reg data set does not contain those small swarms and we therefore included all swarms from this set in the characterizations. Using the measurement data, the maximum likelihood estimates of the parameters for the different model distributions were calculated. The goodness-of-fit (gof) of the model distribution and the measurement data is expressed by the coefficient of determination R^2 which takes values from 0 to 1. A value of $R^2 = 1$ shows that the model function and the measurement data are identical. Thus, we can see a very good match between the measurement data and the model functions. An exception is the size of movie files (Mov) and regional files which only have a gof of $R^2 = 0.86$ and $R^2 = 0.83$, respectively. This can be explained by the fact that the distributions of these file sizes show a strong peak. In particular, 45.85 % of all movie files have a size be-

TABLE 3.9: Multivariate correlation matrix for music swarms with at least 10 peers.

	#peers	#ASs	top AS	file size
#peers	1.0000	0.9100	-0.1364	-0.0048
#ASs	0.9100	1.0000	-0.2979	-0.0071
top AS	-0.1364	-0.2979	1.0000	0.0129
file size	-0.0048	-0.0071	0.0129	1.0000

TABLE 3.10: Multivariate correlation matrix for movie swarms with at least 10 peers.

	#peers	#ASs	top AS	file size
#peers	1.0000	0.8281	-0.0084	0.0043
#ASs	0.8281	1.0000	-0.2160	-0.0000
top AS	-0.0084	-0.2160	1.0000	0.0086
file size	0.0043	-0.0000	0.0086	1.0000

TABLE 3.11: Multivariate correlation matrix for regional swarms with at least 1 peer.

	#peers	#ASs	top AS	file size
#peers	1.0000	0.6102	0.5744	-0.0707
#ASs	0.6102	1.0000	0.0450	0.1259
top AS	0.5744	0.0450	1.0000	-0.2670
file size	-0.0707	0.1259	-0.2670	1.0000

tween 650 MB and 750 MB which corresponds to the size of a regular compact disc. In addition, about 8.46 % of the swarms have a file size between 1350 MB and 1450 MB. Fitting only the file sizes of the remaining 53.31 % of the swarm gives significantly higher gof of 0.99 ('file size (impr.)' in Table 3.7). This is very similar for the Reg data set. 51.65 % of the swarms have a file size between 650 MB and 750 MB and 23.08 % of them are between 1350 MB and 1450 MB. The number of the remaining swarms is too low to provide a meaningful fitting and we therefore suggest to use the corresponding values of the movie files.

However, as we have outlined in Section 3.2, there is a strong correlation between some of the features of BitTorrent swarms. Tables 3.9, 3.10, and 3.11 show the multivariate correlation matrix for music, movie, and regional files, respectively. We observe a strong correlation (> 0.8 for Mus and Mov, and > 0.6 for Reg) between the number of peers in a swarm and the number of different ASes.

In order to generate a random BitTorrent swarm based on this model, approximate methods for sampling correlated random variables from partially specified distributions can be used which are well known in literature, e.g. [33]. For these approximations, the information from the tables presented in the section can be used, respectively.

3.4 Lessons Learned

In this chapter we measure and characterize real-life BitTorrent swarms. The results can serve as input for the design and assessment of traffic optimization techniques as presented in the following chapter or currently discussed in the ALTO working group [62] of the IETF. Still, they are of a generic nature and therefore not limited to this purpose. A core part of our investigation is the result of a large-scale measurement campaign, where a comprehensive set of swarms has been investigated using a distributed tracker monitoring system. Measurements include swarm size distributions, ratio of seeder and leecher populations, time dynamics within a swarm, the distribution of peers over ASes and over countries of swarms, and characteristics of swarms with a certain content or region focus. We show that real-life BitTorrent swarm distributions are highly skewed and that this is in particular true for regional swarms.

On the one hand, more than 90 % of the observed ASes contain less than 10 peers and the average number of peers per AS is below 2 peers for 99 % of the swarms with a very high variation leading to many single peer ASes. On the other hand, most of the peers (about 80 %) belong to the top 20 % of the swarms. Therefore, we argue that there is a large optimization potential for locality-awareness since these large swarms are (1) responsible for the majority of the BitTorrent traffic and (2) especially suitable for locality-aware mechanisms. For this reason, we specify a simple AS swarm characterization for music, movie, and regional files provided in BitTorrent swarms that takes into account the swarm size, the number of different ASes per swarm, the top AS fraction, and the file size. These measurement results and the provided characterizations enable researchers to design algorithms as well as simulation studies and experiments for ALTO solutions based on real-world characteristics of BitTorrent swarms.

4 Performance Evaluation of Caching Strategies in Overlay Networks

One technique that is commonly used by ISPs to decrease their transit traffic is caching. With caching, the ISPs store some of the contents within their network and users can obtain copies of this content from the cache instead of fetching it from remote locations, which results in transit traffic for the ISP. This concept is known, e.g., from web contents, but it is also applied to content distribution overlay networks [61].

Since P2P-based file-sharing systems are one of the major sources of Internet traffic, we investigate them in this chapter and consider again the popular content distribution overlay for file-sharing BitTorrent. For this system P2P caches are already commercially available, e.g., PeerApp's UltraBand [90] and OverSi's OverCache P2P [75]. These solutions follow fundamentally different design principles, yet all of them promise substantial savings in terms of inter-ISP traffic.

The question we address in this chapter is how one can assess the efficiency of P2P caches that follow different design principles in terms of decreasing the inter-ISP traffic, without actually deploying them. To answer this question we develop a fluid model of the system dynamics of BitTorrent-like file-sharing systems that incorporates the effects of P2P caches. We consider the case of a single and of multiple ISPs, and provide a closed-form solution for the equilibrium system state as a function of the cache capacities installed at the different ISPs. We show that under certain conditions a system with two ISPs is sufficient to model swarms spread over multiple ISPs. We develop a simple model of inter-ISP traffic, and use the model to illustrate that one cannot accurately assess the impact of caches on the amount of inter-ISP traffic without considering the effects of the caches on the peer dynamics. We also show that, contrary to intuition, caches can under certain conditions increase the amount of outgoing transit traffic of an ISP. To avoid this phenomenon, we propose a proximity-aware peer selection scheme and evaluate its impact on the cache efficiency. We validate the analytical results via extensive simulations and provide experimental results with real BitTorrent clients to support our results. For the definition of the validation scenarios and for the numerical results we rely on the insights gained from the large-scale measurement study presented in Chapter 3 of this monograph. This ensures that the chosen parameters reflect the nature of live BitTorrent networks in the Internet.

Our model is inspired by the fluid model of BitTorrent-like P2P systems presented in [43]. In that study the authors use the fluid model to derive the average number of peers in the system and to study the service capacity and the effectiveness of file-sharing in such networks. We extend this model in two ways. First, we include the effects of caching in the fluid model of [43]. Second, we provide a simple means to analyze the amount of inter-ISP traffic in scenarios with multiple ISPs. In addition, we show that a given amount of cache upload capacity can lead to different amounts of traffic savings when allocated to different swarms.

Based on this finding we investigate whether ISPs should manage the upload capacity of their caches and actively allocate more upload capacity to certain swarms than to others. To this end, we simulate multi-swarm scenarios and present an allocation policy that can increase inter-ISP traffic savings considerably compared to the demand-driven allocation, i.e., if the upload capacity of the cache is not actively managed by the ISP.

Large parts of this chapter are taken from [8]. In addition, the chapter contains material from [23] and some results from [24, 29]. The chapter is organized as follows. Section 4.1 briefly describes the different P2P cache designs and our network model. We develop the fluid model of the effects of caches on the system dynamics in Section 4.2, and illustrate its importance in predicting the ISP transit traffic in Section 4.3. We describe and evaluate a scheme to improve the cache efficiency in Section 4.4 and investigate allocation policies for the cache upload capacity in multi-swarm scenarios in Section 4.5. Finally, Section 4.6 summarizes this chapter and presents the lessons learned.

4.1 Background and System Description

In this section we present the different types of P2P caches. In addition, we describe our system model of BitTorrent and the ISP level network topology.

4.1.1 Taxonomy of P2P Caches

Caches for P2P traffic can be grouped into three main categories: transparent caches, ISP managed Ultrapeers, and ISP managed Caches. In the following, we give a short description of these categories.

Transparent Caches

To the first category belong the so-called transparent caches. A transparent cache involves deep-packet-inspection (DPI), i.e., the requests for data sent by a local peer (within the ISP) to an external peer are intercepted, and if the requested data is available in the cache, the data is sent to the local peer from the cache. Hence, a transparent cache decreases the amount of incoming transit traffic. The cache also maintains the connection with the external peer. PeerApp's UltraBand family of caches falls into this category.

Ideally, a transparent cache should upload data to local peers at the same rate at which the external peers would upload the data, this way the ISP does not promote the distribution of illegal contents, and is hence not legally liable. If the cache uploads data at the appropriate rate, then its effect on the outgoing transit traffic of the ISP is negligible. In the rest of the paper the term transparent cache will refer to a transparent cache that uploads at the appropriate rate, i.e., it does not contribute additional upload capacity to the P2P system.

ISP Managed Ultrapeers

To the second category belong the caches that appear as high capacity peers to regular peers. These caches do not involve DPI, but they serve only requests of leechers in the network of the ISP that provides the cache. Regular peers are not aware of the fact that these caches are provided by the ISP, and consequently whether a local leecher downloads data from such a cache depends on the neighbor selection algorithms of the P2P protocols. This category of caches inherently increases the upload capacity in the P2P system. We refer to these caches as ISP managed Ultrapeers (*ImU*). OverSi's OverCache P2P falls into this category.

ISP Managed Caches

To the third category belong the caches that are known to the peers via some information exchange with the ISP. Protocols for obtaining such information were proposed for BitTorrent [63], and resource discovery (e.g., cache discovery) is considered for standardization in the IETF Application Layer Traffic Optimization (ALTO) [62] and DECoupled Application Data Enroute (DECADE) [74] working groups. Since peers are aware of the caches, they can prioritize downloading from these caches over downloading from external peers. Just like the *ImU*s these caches serve only requests of leechers in the network of the ISP that provides the cache, and they introduce additional upload capacity in the P2P system. We refer to these caches as ISP managed Caches (*ImC*). We are not aware of any deployments of *ImC* caches due to the lack of localization and resource discovery services in the Internet.

4.1.2 System and Network Model

We consider a BitTorrent-like file-sharing system spread over several ISPs. The ISPs are in the lower layers of the ISP hierarchy, and are hence interested in decreasing their transit traffic. Our focus in this work is on the amount of incoming and outgoing transit traffic of these ISPs, so we can adopt a simple abstraction of the real Internet topology without limiting the validity of our results. In this simple abstraction each ISP is connected to the other ISPs via a global transit network, which only delivers the traffic. This abstraction does not capture the actual routes of the traffic between the ISPs, but the routes can be neglected due to our focus on traffic volumes.

The BitTorrent system we consider consists of a single swarm in which the peers are located in a set $\mathcal{I} = \{1, \ldots, I\}$ of ISPs. Every ISP can install a cache to decrease its transit traffic. If installed in ISP i, the cache provides an upload capacity of κ_i to the swarm. This abstraction of a P2P cache is novel, but is easy to justify: whatever data is uploaded from the cache does not have to be uploaded from a peer and hence the cache provides additional upload capacity to the swarm.

Initially, the swarm consists only of the initial seed and the caches. Peers arrive in the network of ISP i according to a Poisson process with rate λ_i. While over the lifetime of a swarm (e.g., in the order of months or years) the peer arrival process is not homogeneous, over short periods the peer arrival process can be reasonably approximated by a Poisson process [47], as it can be considered the superposition of a large number of renewal processes [30]. Leechers abort the download at rate θ, that is, the longer it takes to download a content the higher the probability that a peer would abort the download. Seeds leave the swarm at rate γ, i.e., peers stay for $1/\gamma$ time on average after becoming a seed. Similar assumptions were used in most analytical studies for modeling P2P file-sharing systems (e.g., [43, 57]).

Peers have upload capacity μ and download capacity c, and we consider the practically relevant case of $c \geq \mu$. We denote by $\eta \in [0, 1]$ the probability that a leecher can utilize its capacity to upload to some other leecher, and we refer to it as the effectiveness of file-sharing [43]. In the mathematical model we assume without loss of generality that the file size is 1, so that μ,c and κ_i are normalized to the file size. For the sake of simplicity, we assume homogeneous peer capacities. Table 4.1 summarizes the notation used in this chapter.

TABLE 4.1: Frequently used notation.

Parameter	Definition
$\mathcal{I}, I$	Set and number of ISPs, respectively
κ_i	Cache upload capacity in ISP i
λ_i	Arrival rate in ISP i
θ	Abort rate of leechers
γ	Departure rate of seeds
η	Effectiveness of file sharing
μ	Peer upload capacity
c	Peer download capacity
$x_i(t)$	Number of leechers in ISP i at time t
$y_i(t)$	Number of seeds in ISP i at time t
ρ_i^I	Incoming transit traffic in ISP i
ρ_i^O	Outgoing transit traffic in ISP i

4.2 System Dynamics with Caching

In the following we develop a fluid model of a BitTorrent-like file-sharing system spread over several ISPs. Our goal is to capture the effects of caches on the system dynamics and ultimately on the amount of traffic exchanged between the ISPs. We consider two types of caches, *ImU* and *ImC*, and use transparent caches as a baseline for comparison. Our model builds on the model developed in [43], and we use the same notations as much as possible.

We denote by $x_i(t)$ and $y_i(t)$ the number of leechers and the number of seeds in ISP i at time t, respectively. The rate at which leechers can obtain data is limited by the available upload rate in the system and by their download rate. The upload rate $U_i(\mathbf{x}, \mathbf{y}, \kappa)$ available to leechers in ISP i is a function of the number of leechers, the number of seeds and the cache upload rate in the different ISPs, where $\mathbf{x} = (x_1, \ldots, x_I)$, $\mathbf{y} = (y_1, \ldots, y_I)$ and $\kappa = (\kappa_1, \ldots, \kappa_I)$. The exact form of U_i depends on the cache bandwidth allocation policies followed by the ISPs and the neighbor selection policies of the peers. Together with the constraint of the download rate, the rate at which leechers obtain data in ISP i is given by $min(cx_i, U_i(\mathbf{x}, \mathbf{y}, \kappa))$. Following the assumptions used in [43] on the arrivals, aborts, and departures we get that the evolution of the mean number of leechers and seeds in ISP i can be described by a system of coupled differential equations

$$\frac{dx_i(t)}{dt} = \lambda_i - \theta x_i(t) - min\{cx_i(t), U_i(\mathbf{x}, \mathbf{y}, \kappa)\} \tag{4.1}$$

$$\frac{dy_i(t)}{dt} = min\{cx_i(t), U_i(\mathbf{x}, \mathbf{y}, \kappa)\} - \gamma y_i(t). \tag{4.2}$$

We are interested in the steady-state of the system, i.e., when the rate of change of the number of leechers and seeds is zero

$$\frac{dx_i(t)}{dt} = \frac{dy_i(t)}{dt} = 0 \qquad i = 1, \dots, I. \tag{4.3}$$

In the following we consider various scenarios and develop closed form solutions for the steady-state number of leechers and seeds. The results we develop in this section depend only on the available upload rate in the system, hence we do not have to distinguish between the different kinds of non-transparent caches (*ImU* and *ImC*). We will, however, distinguish between the three types of caches in Section 4.3 when estimating the transit traffic between the ISPs.

4.2.1 The Case of a Single System

Let us first consider the case of a single system ($\mathcal{I} = \{1\}$). This scenario allows us to understand the aggregate effect of caches on the system dynamics. For simplicity we omit the subscript i in the rest of this subsection. This scenario differs from the one considered in [43] in that the available upload rate is increased by the cache's upload rate. The available upload rate is the sum of the upload rate of the leechers, the seeds and that of the installed cache, and can be expressed as

$$U(x, y, \kappa) = \mu(\eta x + y) + \kappa. \tag{4.4}$$

Substituting this into Equations 4.1 and 4.2 we get for the steady-state

$$\begin{aligned} 0 &= \lambda - \theta\overline{x} - min\{c\overline{x}, \mu(\eta\overline{x} + \overline{y}) + \kappa\} && (4.5) \\ 0 &= min\{c\overline{x}, \mu(\eta\overline{x} + \overline{y}) + \kappa\} - \gamma\overline{y}. && (4.6) \end{aligned}$$

Let us first consider the download-rate-limited case, when the available upload rate exceeds the maximum download rate of the leechers, i.e., $c\overline{x} \leq \mu(\eta\overline{x}+\overline{y})+\kappa$. It is easy to see that in this case the presence of caches does not affect the steady-state number of leechers and seeds. Hence, they are the same as in [43]

$$\begin{aligned} \overline{x} &= \frac{\lambda}{c(1+\frac{\theta}{c})} && (4.7) \\ \overline{y} &= \frac{\lambda}{\gamma(1+\frac{\theta}{c})}. && (4.8) \end{aligned}$$

The condition under which the download rate is the limit is however different from that in [43]. Given the expressions for the steady-state number of leechers (Equation 4.7) and seeds (Equation 4.8) it is

$$\kappa \geq \frac{\lambda\{c(\gamma-\mu) - \gamma\eta\mu\}}{\gamma(\theta+c)}. \qquad (4.9)$$

Next, we consider the upload-rate-limited case, when the maximum download rate of the leechers exceeds the available upload rate, i.e., $c\overline{x} \geq \mu(\eta\overline{x}+\overline{y})+\kappa$. Here we get

$$\begin{aligned} \overline{x} &= \frac{\lambda}{\nu\left(1+\frac{\theta}{\nu}\right)} - \frac{\kappa}{\mu\eta\left(1+\frac{\theta}{\nu}\right)} && (4.10) \\ \overline{y} &= \frac{\lambda}{\gamma\left(1+\frac{\theta}{\nu}\right)} + \frac{\kappa\theta}{\mu\eta\gamma\left(1+\frac{\theta}{\nu}\right)}, && (4.11) \end{aligned}$$

where $\frac{1}{\nu} = \frac{1}{\eta}(\frac{1}{\mu} - \frac{1}{\gamma})$. Again, given the steady-state number of leechers (Equation 4.10) and seeds (Equation 4.11) we can express the condition under which the upload rate is the limit

$$\kappa \leq \frac{\lambda\{c(\gamma - \mu) - \gamma\eta\mu\}}{\gamma(\theta + c)}. \tag{4.12}$$

Note that since the cache upload rate is non-negative it must be that $\gamma > \mu$, which implies that $\nu > 0$ for an upload-rate-limited system. If $\gamma \leq \mu$ then the system has to be download-rate-limited. From Equations 4.10 and 4.11 we draw the following conclusions.

- For $\kappa = 0$ the results coincide with those in [43], as expected.

- For $\kappa > 0$ the number of leechers is always lower than without a cache in steady-state. The effect of the cache decreases as the peers' upload rates and the effectiveness of file sharing increase because of the cache's diminishing contribution to the upload rate.

- Interestingly, the steady-state number of seeds is insensitive to the cache's upload rate if peers never abort downloads ($\theta = 0$), but for $\theta > 0$ the number of seeds increases with κ. The increase is inversely proportional to the peers' upload rates and the effectiveness of file sharing. Consequently, when $\theta > 0$, installing a cache increases the available upload rate more than the cache's upload rate itself through an increased number of seeds by a factor of $\theta/\eta\gamma\left(1 + \frac{\theta}{\nu}\right)$. This phenomenon is explained by the fact that due to the increased upload capacity, leechers become seeds faster and hence the number of aborting leechers decreases.

- If $\theta/\gamma > 1$ then the number of peers in the system increases linearly with the amount of cache capacity installed. For $\theta/\gamma < 1$ the contrary is true, while for $\theta/\gamma = 1$ the decrease in the number of leechers equals the increase in the number of seeds.

4.2.2 The Case of Multiple Systems

Let us consider now how installing a cache affects the system dynamics when peers are located in several ISPs. We make the reasonable assumption that the cache operated by ISP i only serves leechers in ISP i, but seeds and leechers upload and download data to and from all peers.

The upload rate available to leechers in ISP i has now three sources: the cache provided by ISP i and the leechers and seeds in all ISPs. The cache upload rate in ISP i is κ_i. The total upload rate from leechers and seeds in the system is $\mu(\eta \sum_{j\in\mathcal{I}} x_j + \sum_{j\in\mathcal{I}} y_j)$. Since this upload rate is shared among all $\sum_{j\in\mathcal{I}} x_j$ leechers, the total upload rate available to the x_i leechers in ISP i is

$$U_i(\mathbf{x},\mathbf{y},\kappa) = \mu\left(\eta x_i + \sum_{j\in\mathcal{I}} y_j \frac{x_i}{\sum_{j\in\mathcal{I}} x_j}\right) + \kappa_i. \tag{4.13}$$

We provide analytical results for two scenarios, when all ISPs are upload-rate-limited (i.e., $cx_i \geq U_i(\mathbf{x},\mathbf{y},\kappa)$), and when all ISPs are download-rate-limited.

In the case when the system is upload-rate-limited in all ISPs, we can substitute $U_i(\mathbf{x},\mathbf{y},\kappa)$ into Equations 4.1 and 4.2 for every $i \in \mathcal{I}$ and solve the system of equations to get the steady-state number of leechers and seeds

$$\begin{aligned}
\overline{x}_i &= \frac{\lambda_i}{\nu\left(1+\frac{\theta}{\nu}\right)} - \frac{\kappa_i}{\mu\eta\left(1+\frac{\theta}{\nu}\right)} - \Delta_i(\mathbf{x},\mathbf{y},\kappa) && (4.14)\\
\overline{y}_i &= \frac{\lambda_i}{\gamma\left(1+\frac{\theta}{\nu}\right)} + \frac{\kappa_i\theta}{\mu\eta\gamma\left(1+\frac{\theta}{\nu}\right)} + \frac{\theta}{\gamma}\Delta_i(\mathbf{x},\mathbf{y},\kappa), && (4.15)
\end{aligned}$$

where

$$\Delta_i(\mathbf{x},\mathbf{y},\kappa) = \frac{\sum_{j\in\mathcal{I}}(\lambda_i\kappa_j - \kappa_i\lambda_j)}{\eta\gamma\left(1+\frac{\theta}{\nu}\right)\left(\sum_{j\in\mathcal{I}}(\lambda_j - \kappa_j)\right)}. \tag{4.16}$$

From Equations 4.14 and 4.15 we can obtain the following insights:

- Increasing the cache upload rate κ_i leads to a decrease of the number of leechers in ISP i independent of the arrival intensities and the cache upload rates in the other ISPs. At the same time it can increase the number of seeds. The changing ratio of leechers and seeds affects the amount of transit traffic, which we will quantify in Section 4.3. To verify that Equation 4.14 is a monotonically decreasing function of κ_i in an upload-rate-limited system, we evaluate the first and second derivatives of Equation 4.14 w.r.t. κ_i. Equation 4.14 has two extrema (minimum and maximum) if and only if $\sum_{j \neq i}(\lambda_j - \kappa_j) > 0$. The minimum is reached at $\kappa_i < \lambda_i + \sum_{j \neq i}(\lambda_j - \kappa_j)$, but at this value $\overline{x}_i < 0$, and the system can not be in the upload-rate-limited regime. The maximum is reached for $\kappa_i > \lambda_i + \sum_{j \neq i}(\lambda_j - \kappa_j)$, which can not be in the upload-rate-limited regime either. Hence, as we increase κ_i, the number of leechers decreases until we reach the download-rate-limited regime. A similar reasoning holds for Equation 4.15.

- $\Delta_i(\mathbf{x}, \mathbf{y}, \kappa)$ given in Equation 4.16 is a function of $\sum_{j \in \mathcal{I} \setminus \{i\}} \lambda_j$ and $\sum_{j \in \mathcal{I} \setminus \{i\}} \kappa_j$. Hence $\Delta_i(\mathbf{x}, \mathbf{y}, \kappa)$ and consequently $\overline{y}_i$ and $\overline{x}_i$ only depend on the sum of the arrival intensities and the sum of the cache upload rates in the other ISPs but not on their individual values.

- Since $\sum_{i \in \mathcal{I}} \Delta_i(\mathbf{x}, \mathbf{y}, \kappa) = 0$, we have that $\sum_{i \in \mathcal{I}} \overline{x}_i = \overline{x}$ as given in Equation 4.10 and $\sum_{i \in \mathcal{I}} \overline{y}_i = \overline{y}$ as given in Equation 4.11. That is, the total number of leechers and seeds in all ISPs only depends on the aggregate peer arrival intensity and the aggregate amount of cache upload rate.

In Section 4.2.3 we show simulation and experimental results to verify these analytical results.

Let us consider now when the system is download-rate-limited in ISP i (i.e., $cx_i \leq U_i(\mathbf{x}, \mathbf{y}, \kappa)$). Then the steady-state number of leechers and seeds in ISP i

is given by

$$\overline{x}_i = \frac{\lambda_i}{c(1+\frac{\theta}{c})} \tag{4.17}$$

$$\overline{y}_i = \frac{\lambda_i}{\gamma(1+\frac{\theta}{c})}. \tag{4.18}$$

In this case the number of leechers and seeds is not directly influenced by the cache upload rate κ_i of ISP i. Nevertheless, whether the system in ISP i is download-rate-limited depends on the cache upload rate κ_i of ISP i, the number of leechers in the other ISPs and hence indirectly on the cache upload rates in the other ISPs.

4.2.3 Model Validation

In this section we validate the model via simulations and experiments with real BitTorrent clients. The simulations allow us to verify the accuracy of the analytical model and the validity of our conclusions based on the model for a wide range of system parameters. The experiments, even though smaller in scale than the simulations, allow us to verify the accuracy of both the model and the simulation results for a limited set of system parameters. Before presenting the numerical results in Section 4.2.3, we briefly describe our simulation and experiment setups.

Simulation Setup

We implemented the BitTorrent protocol in the ProtoPeer [65, 67] framework. The implementation includes the piece selection mechanism, the management of the neighbor set, and the choke algorithm. Furthermore, it covers the message exchange between the peers as well as between the peers and the tracker. For scalability reasons, we use the flow-based network model provided by ProtoPeer. Our implementation is publicly available as a library for ProtoPeer [65].

The size of the shared file is 150 MB which corresponds to a movie or TV show of about half an hour duration in medium quality. Peers join the swarm at a rate of 6.6 per minute and their upload and download capacities are 1 Mbit/s and 16 Mbit/s, respectively. These are typical values for relatively well-provisioned home user Internet access connections in Europe. Normalizing by the file size, these upload and download capacities are equivalent to $\mu = 0.05$ and $c = 0.8$ for the analytical model. Each peer is associated with one ISP and we use this association to calculate the inter-ISP traffic. Each simulation run corresponds to 8 hours, and we discard an initial 2 hours warm-up period. The initial seed leaves the swarm after 1 hour, so it has no influence on the swarm in the steady-state. This setup results in an average number of 3200 peers for each simulation run and swarms with around 120 peers concurrently online in the small scenario. Such swarm sizes are typical for swarms sharing movies according to the measurements presented in Section 3.2.1 of this monograph.

The *ImUs* are implemented as normal BitTorrent clients, but they only upload data to peers in the same ISP. We do not simulate *ImCs* as their behavior is not yet clear (i.e., it is not known what algorithms they would use to select leechers to upload to). The presented simulation results are the averages of 20 simulation runs, and we show confidence intervals at a 95%-confidence level.

If not stated otherwise, in the remainder of this study, peers have an average seeding time of 10 minutes, i.e., $\gamma = 0.1$. Leechers abort the download with intensity $\theta = 0.01$, i.e., on average a leecher waits for 100 minutes until it leaves the swarm if the file is not yet downloaded. For the upload and download rates we use $\mu = 0.05$ and $c = 0.8$, respectively. All these variables have the dimension min^{-1}, we however omit them for the sake of clarity. For the effectiveness of file sharing we use $\eta = 0.9$ in the model, i.e., close to 1 as shown in [43].

Experiment Setup

All measurements are performed in the experimental facility of the German-Lab (G-Lab) project [60]. This experimental facility is distributed over 5 universities in Germany. It consists of 152 nodes running Planet-Lab [58] software (version 4.2.1), the operating system of all nodes is Linux (Fedora Core 8, x86_64). G-

Lab provides a controlled environment in which reproducible experiments can be performed. In contrast to Internet-wide experiments (e.g., on Planet-lab), packet loss rates and latencies between the nodes are very low. However, Rao et al. [76] showed that these parameters have only a marginal impact on BitTorrent performance, and consequently our results are representative for Internet-wide scenarios.

We use the standard BitTorrent client (version 4.4.0-7-fc8) of the Fedora Linux distribution and limit the access speeds of each BitTorrent client on application layer. We use the same arrival, departure and abort behavior for experiments as for the simulations to make them easily comparable. We grouped the nodes of the experimental facility into "virtual" ISPs and calculated the amount of inter-ISP traffic according to the source and the destination of the exchanged messages between the peers. We repeated all experiments 5 times and show 95%-confidence intervals.

The size of the shared file is 7.031 MB and we adjusted the upload capacity of the peers to 6 KB/s so that the normalized upload rate equals that of the simulations $\mu = 0.05$. This reduces the amount of exchanged data in the experimental facility by more than 95% while keeping the results comparable.

Simulation and Experimental Validation

We start with the validation of the observation that the system dynamics in ISP i depend only on the aggregate arrival intensity and the aggregate cache upload rate in the rest of the ISPs. Then, we show results from simulations and experiments for varying cache capacities and compare them to the model.

For the validation we consider a tagged ISP, ISP 1, and the rest of the Internet, which consists of a number I^* of ISPs. Hence, the total number of ISPs considered is $I = I^* + 1$. We set the upload capacity of the cache in ISP 1 to $\kappa_1 = 0.1$ and the arrival rate to $\lambda_1 = 0.6$ and vary the number of the other ISPs $I^* \in \{1, 5, 10, 20\}$. Peers join the other ISPs with an aggregate arrival rate $\lambda^* = 6$ and the aggregate cache upload rate in the other ISPs is κ^*. Note that the cache upload rates κ_1 and κ^* are normalized to the size of the shared file (cf. Section 4.1.2) and their dimension is $\min^{-1}$ to be consistent with the upload and download rates of the peers defined at the end of the simulation setup in this section. The peer arrival intensities and the cache upload capacities are equal in the other ISPs, i.e., for $i \neq 1$ we use $\kappa_i = \kappa^*/I^*$ and $\lambda_i = \lambda^*/I^*$.

We show results from simulations for the number of leechers in ISP 1 $\overline{x}_1$ and in the whole swarm $\overline{x}$ in Figure 4.1. The figure shows that for a given aggregate cache capacity κ^* the number of ISPs I^* has no significant impact on the number of leechers in ISP 1 and in the whole swarm. The simulation results match the values predicted from the model quite well, within 10% accuracy, except for $\kappa^* = 2$. For $\kappa^* = 2$ we observe up to 30% difference between the simulation and the analytical results, and we also observe that the number of ISPs I^* has an effect on the number of leechers. The reason is that for $\kappa^* = 2$ the system is oscillating between a download-rate-limited and an upload-rate-limited state. Therefore, some of the upload capacity of the caches remains unused in periods when the system is download-rate-limited. The oscillation depends on the arrival process of the peers which is stochastic. Consequently, a system which is upload-rate-limited on average can switch to a download-rate-limited system for some time. However, the equations for the steady-state of the model do not account for those fluctuations and that can lead to inaccuracies for parameter settings where the system is not clearly download or upload-rate-limited.

We verified the above two hypotheses also for the number of seeds and for different arrival rates in non-tagged ISPs λ^*, but we omit the figures. The simulation results confirm the conclusions we drew from the mathematical model: the system dynamics in ISP i only depend on the aggregate cache capacity κ^* and the aggregate arrival intensity λ^* of the rest of the ISPs. Therefore, in the rest of the paper we focus on a scenario with two ISPs termed ISP 1 and ISP 2 where ISP 2 represents "the rest of the world". If not stated differently, we set $\lambda_1 = 0.6$ and $\lambda_2 = 6$ so that 10 times more peers join the swarm in ISP 2 than in ISP 1. Furthermore, ISP 2 does not use a cache, i.e. $\kappa_2 = 0$.

In order to further validate the model, we consider the dependency of the system dynamics on the cache capacity κ_1 of ISP 1. We performed simulations and

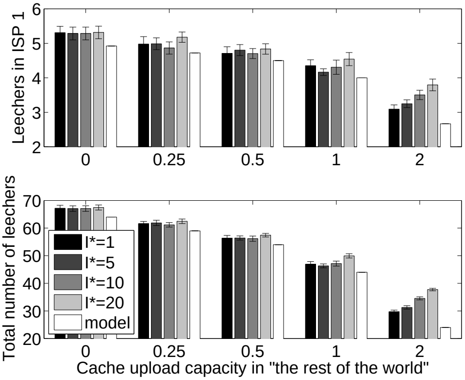
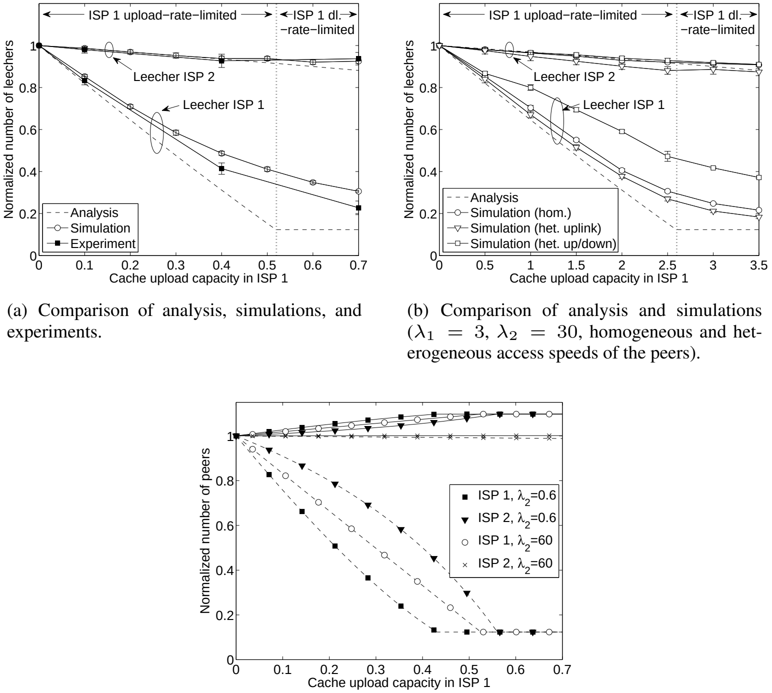


FIGURE 4.1: Average number of leechers $\overline{x}_1$ in ISP 1 (top) and in the whole swarm $\overline{x}$ (bottom) for different numbers of other ISPs I^* and aggregate cache capacities κ^* in "the rest of the world". $\lambda_1 = 0.6$, $\kappa_1 = 0.1$, $\lambda^* = 6$.

experiments with different values of κ_1, and measured the number of leechers and seeds. In Figure 4.2(a) we compare the number of leechers obtained using the analytical model, the simulations, and the experiments. The figure shows the number of leechers $\overline{x}_i$ in ISP i as a function of the cache upload capapcity κ_1 in ISP 1 normalized by the number of leechers $\overline{x}_i|_{\kappa_1=0}$ in the case of no caching. Consequently, for $\kappa_1 = 0$ all results are equal to 1. The figure confirms that the model provides accurate results, in particular for small cache capacities. However, the simulations show that the number of leechers $\overline{x}_1$ in ISP 1 is significantly higher than predicted by the model for $\kappa_1 = 0.5$. The reason for this mismatch is the same as explained above, i.e., a system which is on average upload-rate-limited can get download-rate-limited for a period of time if only very few leechers are online. However, almost all swarms we observe in practice are clearly upload-rate-limited, and for upload-rate-limited systems the model provides very accurate results.



(a) Comparison of analysis, simulations, and experiments.

(b) Comparison of analysis and simulations ($\lambda_1 = 3$, $\lambda_2 = 30$, homogeneous and heterogeneous access speeds of the peers).

(c) Analytical results (dashed lines: leechers, solid lines: seeds).

FIGURE 4.2: Normalized number of leechers $\frac{\overline{x}_i}{\overline{x}_i|_{\kappa_1=0}}$ and seeds $\frac{\overline{y}_i}{\overline{y}_i|_{\kappa_1=0}}$ as a function of the cache upload capacity κ_1 of ISP 1. The figures show the number of leechers $\overline{x}_i$ and seeds $\overline{y}_i$ in ISP i divided by the corresponding values for the case without caching ($\overline{x}_i|_{\kappa_1=0}$ and $\overline{y}_i|_{\kappa_1=0}$).

We conclude the validation of the system dynamics with simulation results for larger swarms. To this end, we increase the arrival rates to $\lambda_1 = 3$ and $\lambda_2 = 30$ which leads to swarm sizes of about 600 peers concurrently online. We simulate three scenarios: i) homogeneous peer upload and downloads speeds; ii) heterogeneous peer upload speeds, and iii) heterogeneous peer upload and download speeds. For the homogeneous scenario, we keep the default upload and download capacities (Section 4.2.3) for all peers. For the two heterogeneous scenarios we create groups of slow, medium, and fast peers and assign every new peer to one of the groups with probabilities $(0.4, 0.5, 0.1)$, respectively. In the scenario with heterogeneous upload speeds, we use upload speeds of $(0.0125, 0.05, 0.2)$ for these groups and keep the same download speed as in the homogeneous scenario. In the scenario with heterogeneous upload and download speeds we use download speeds of $(0.2, 0.8, 3.2)$ in addition. Using these parameters the average access speeds are the same in the homogeneous and in the heterogeneous scenarios.

The results are shown in Figure 4.2(b). The difference between the scenarios with homogeneous and with heterogeneous upload speeds is negligible, which indicates that the model is accurate for swarms where peers have heterogeneous upload speeds, as long as the average access speeds per ISP are the same. We also note that in comparison to Figure 4.2(a) the number of leechers in the simulations is significantly closer to the analytical results. The better match between the analytical and the simulation results is due to the higher number of peers, as the oscillations of the system between the upload and download-rate-limited state are less prevalent. However, for the scenario with heterogeneous upload and download speeds the number of leechers in ISP 1 obtained from the simulations is about 20% higher than predicted by the model. The reason is that most of the slow peers reach their download limit already for small cache capacities κ_1, and a further increase of κ_1 does not reduce their download time and their number in the system.

4.2.4 Numerical Results

The validation presented above allows us to consider two ISPs when evaluating the effects of the cache upload rate κ_i of ISP i on the system dynamics in ISP i. In the following we will use such a simple scenario to evaluate the effects of the cache upload rate on the number of leechers and seeds in the system.

Figure 4.2(c) shows the normalized number of leechers and seeds in steady-state in both ISPs for two values of the arrival intensity in ISP 2. Like in Figures 4.2(a) and 4.2(b), all values are normalized with the values obtained in the case without caching, i.e., $\kappa_1 = 0$. For the case of equal arrival intensities in the two ISPs ($\lambda_1 = \lambda_2$) the effect of the cache capacity on the number of peers in the system is significant in both ISPs. For the case when $\lambda_2 \gg \lambda_1$ the effect of the cache upload rate on ISP 1 is just slightly smaller. In both cases we can observe the cache upload rate at which ISP 1 becomes download-rate-limited, i.e., above which rate the number of leechers and seeds in the ISP does not change. The proportional decrease of the number of leechers is bigger than that of the number of seeds, which might lead to an unwanted effect of the introduction of a cache: more seeds in ISP 1 will upload to leechers in ISP 2 thereby increasing the outgoing traffic of ISP 1. In the following section we investigate under what conditions this unwanted effect can be observed.

4.3 The Impact of Caching on Transit Traffic

To illustrate the importance of the effect of the cache upload rate on the system dynamics, in this section we develop a simple model of the transit traffic of the ISPs and use the model to give analytical and numerical results.

Ideally, one would expect that by installing upload rate κ_i ISP i can decrease its incoming transit traffic ρ_i^I by at least κ_i. This would be the case for traditional Web caching, for example. For the case of P2P let us consider the decrease of the incoming transit traffic ρ_i^I if ISP i installed a transparent cache. The transparent cache serves requests that would generate incoming transit traffic, hence a cache upload rate of κ_i decreases the amount of incoming traffic ρ_i^I by κ_i. Requests are typically much smaller than the replies that contain the actual data, so the effect of the transparent cache on the amount of outgoing transit traffic ρ_i^O is minimal. An alternative expectation can be that if ISP i installs cache upload rate κ_i then it decreases its total transit traffic $\rho_i^I + \rho_i^O$ by at least κ_i.

4.3.1 A Simple Model of Transit Traffic

Estimating the amount of transit traffic generated by a set of peers in an ISP is difficult in general, because the effects of the neighbor selection algorithms (e.g., choking/unchoking in BitTorrent), of the inter-ISP delays and bandwidth bottlenecks are hard to model. The model we describe in the following does not take into account such details, but it provides a way to quantify the effects of the cache upload rate on the amount of transit traffic. More accurate models of the data exchange between peers might give quantitatively different results, but our simulations and experiments show that this simple model captures many of the most important factors.

The approximation we derive in the following is based on two assumptions.

Assumption 1. (Competition) Leechers compete with each other for the available upload rate as long as they would be able to download at a higher rate.

Assumption 2. (Proportionality) Given a single byte downloaded in ISP i, the distribution of its sources is proportional to the amount of upload rate exposed to the leechers in ISP i.

To simplify the notation, we define the publicly available upload rate in ISP i as the available upload rate located in ISP i that can be used by leechers in any ISP, and denote it by u_i^P. For the scenario considered in this section this quantity is given by the upload rate of the leechers and the seeds $u_i^P = \mu(\eta x_i + y_i)$. Similarly, we define the locally available upload rate in ISP i as the upload rate that is only available to leechers in ISP i. For the considered scenario this quantity is given by the upload rate of the cache, $u_i^L = \kappa_i$.

Let us first consider the ISP managed Ultrapeer (*ImU*). The *ImU* appears as an arbitrary peer to the leechers in ISP i. The leechers in ISP i demand data at a total rate of cx_i. The demand is directed to the locally available upload rate u_i^L of ISP i and to the publicly available upload rate $\sum_j u_j^P$ of all ISPs. The leechers demand from the locally available upload rate with a probability proportional to its value u_i^L, i.e., with probability $u_i^L/(\sum_j u_j^P + u_i^L)$. The rest they demand from the publicly available upload rate, so the rate D_i^d that leechers in ISP i demand from the publicly available upload rate can be expressed as

$$D_i^d = cx_i \left(1 - \frac{u_i^L}{\sum_j u_j^P + u_i^L}\right). \tag{4.19}$$

Consider now the ISP managed cache (*ImC*). The leechers demand by preference from the *ImC*, hence their total demand is decreased by the cache capacity κ_i. If the *ImC* can serve the demand then no publicly available upload rate is demanded by the leechers in ISP i. Otherwise, the leechers demand publicly available upload rate with a probability proportional to the amount of publicly available upload rate, at a rate of

$$D_i^d = max(0, cx_i - \kappa_i) \left(1 - \frac{u_i^L - \kappa_i}{\sum_j u_j^P + u_i^L - \kappa_i}\right). \tag{4.20}$$

Since $u_i^L = \kappa_i$, leechers in ISP i demand from the publicly available upload rate what cannot by served by the *ImC*.

If the system is download-rate-limited then the leechers receive the demanded rate. If the system is upload-rate-limited then the received rate of the leechers in ISP i is proportional to the total publicly available upload rate divided by the total demanded rate

$$D_i^r = D_i^d min\left(1, \frac{\sum_j u_j^P}{\sum_j D_j^d}\right). \tag{4.21}$$

The rate that the leechers receive can originate from any ISP, and it is hard to provide an accurate estimate of the share of the traffic that would originate from outside the ISP, as factors such as the available bandwidth between ISPs and the end-to-end delays influence the download process. Applying Assumption 2 again we get the following estimate for the incoming transit traffic of ISP i.

Proposition 1. *Under Assumptions 1 and 2 the estimated incoming transit traffic of ISP i is*

$$\rho_i^I = D_i^r \left(1 - \frac{u_i^P}{\sum_j u_j^P} \right). \tag{4.22}$$

where D_i^r is defined in Equations 4.19-4.21.

We estimate the outgoing transit traffic based on the incoming transit traffic estimates and by using Assumption 2, i.e., the amount of traffic that ISP i uploads to ISP j is proportional to the ratio of the publicly available upload rate in ISP i and the aggregate publicly available upload rate outside ISP j.

Proposition 2. *Under Assumptions 1 and 2 the estimated outgoing transit traffic of ISP i is*

$$\rho_i^O = \sum_{j \neq i} \rho_j^I \frac{u_i^P}{\sum_{k \neq j} u_k^P}. \tag{4.23}$$

In the following we use these simple estimates to quantify the effects of the cache upload rate on the incoming and outgoing transit traffic of the ISPs.

4.3.2 Asymptotic Results of Cache Efficiency

Motivated by the results of Section 4.2.2 we consider the case of two ISPs, a tagged ISP ($i = 1$) and the rest of the ISPs represented by ISP $i = 2$, ($\mathcal{I} = \{1, 2\}$). We analyze the effects of the cache upload rate κ_1 installed by ISP 1 on the amount of traffic exchanged between the two ISPs in the limiting case when $\lambda_2 \to \infty$ and in an upload-rate-limited system. For λ_2 sufficiently large if $\mu\gamma\eta < c(\gamma - \mu)$ then the system is upload-rate-limited (see Equation 4.12).

Proposition 3. *In an upload-rate-limited system the asymptotic transit traffic savings of ISP* 1 *achieved by the* ImU *are*

$$\lim_{\lambda_2 \to \infty} (\rho_1^I|_{\kappa_1=0} - \rho_1^I) = \frac{\kappa_1}{\left(1+\frac{\theta}{\nu}\right)} \tag{4.24}$$

$$\lim_{\lambda_2 \to \infty} (\rho_1^O|_{\kappa_1=0} - \rho_1^O) = \frac{\kappa_1}{\left(1+\frac{\theta}{\nu}\right)} - \frac{\mu\kappa_1}{\gamma}. \tag{4.25}$$

Proof. For an upload-rate-limited system and small cache upload rates κ_i we can give an upper bound on the incoming transit traffic in ISP i as $\frac{x_i}{\sum_{j\in\mathcal{I}} x_j}$ share of the total upload rate from leechers and seeds in all other ISPs $j \neq i$, i.e., $\sum_{j\neq i} u_j^P$,

$$\rho_i^I = \frac{x_i}{\sum_{j\in\mathcal{I}} x_j} \sum_{j\neq i} u_j^P. \tag{4.26}$$

Substituting this expression into Equation 4.23 we get an upper bound on the outgoing transit traffic intensity

$$\rho_i^O = \left(1 - \frac{x_i}{\sum_{j\in\mathcal{I}} x_j}\right) u_i^P. \tag{4.27}$$

Let us now substitute Equations 4.14 and 4.15 into Equations 4.26 and 4.27. By increasing the peer arrival rate in ISP 2 to infinity we get Equations 4.24 and 4.25. □

Both Equations 4.24 and 4.25 are independent of the cache upload rate κ_2 in ISP 2, and the arrival intensity in ISP 1. We also note that since $\nu > 0$ we have $1 + \frac{\theta}{\nu} \geq 1$, so that the incoming transit traffic gain is always less than the cache upload rate installed by the ISP. The same is true for the outgoing transit traffic gain. The sum of the gains can however exceed the cache upload rate. We conclude that a transparent cache is preferable over an *ImU* for an ISP whose transit traffic costs are only a function of the amount of incoming transit traffic. Nevertheless, an *ImU* might be preferable if the ISP is charged based on the maximum of the incoming and the outgoing transit traffic.

For the *ImC* we can formulate a similar result.

Proposition 4. *In an upload-rate-limited system the asymptotic transit traffic savings of ISP* 1 *achieved by the* ImC *are*

$$\lim_{\lambda_2\to\infty}(\rho_1^I|_{\kappa_1=0}-\rho_1^I) = \frac{\kappa_1}{\left(1+\frac{\theta}{\nu}\right)}+\frac{\kappa_1\mu\eta}{c}\frac{\gamma}{(\gamma-\mu)} \tag{4.28}$$

$$\lim_{\lambda_2\to\infty}(\rho_1^O|_{\kappa_1=0}-\rho_1^O) = \frac{\kappa_1}{\left(1+\frac{\theta}{\nu}\right)}-\frac{\mu\kappa_1}{\gamma}. \tag{4.29}$$

Proof. Consider the upper bound for the incoming transit traffic

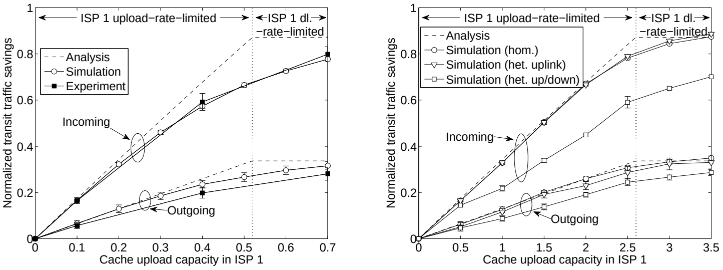
$$\rho_i^I=\frac{cx_i-\kappa_i}{\sum_{j\in\mathcal{I}}cx_j-\kappa_j}\sum_{j\neq i}u_j^P, \tag{4.30}$$

and substitute this into Equation 4.23 to get the upper bound on the outgoing transit traffic

$$\rho_i^O=\left(1-\frac{cx_i-\kappa_i}{\sum_{j\in\mathcal{I}}cx_j-\kappa_j}\right)u_i^P. \tag{4.31}$$

We substitute Equations 4.14 and 4.15 into Equations 4.30 and 4.31 and increase the arrival rate in ISP 2 to infinity to get Equations 4.28 and 4.29. □

Again, the expressions are independent of κ_2, and the arrival intensity in ISP 1. Depending on the value of the rightmost term of Equation 4.28 the efficiency of the cache upload rate for *ImC* can exceed 1. Consequently, an *ImC* can outperform a transparent cache in terms of the decrease of the incoming transit traffic. Comparing Equation 4.24 to Equation 4.28 we observe that the bound for the gain in terms of incoming transit traffic is higher for the *ImC* than for the *ImU* (because $\gamma > \mu$ for an upload-rate-limited system). An intuitive explanation for the superioririty of the *ImC* is that its upload rate is better utilized because leechers download from the *ImC* by preference. Comparing Equation 4.25 to Equation 4.29 we observe, however, that the bounds for the gain in terms of outgoing transit traffic are equal for the *ImU* and for the *ImC*.



(a) Comparison of analysis, simulations, and experiments.

(b) Comparison of analysis and simulations ($\lambda_1 = 3$, $\lambda_2 = 30$, homogeneous and heterogeneous access speeds of the peers).

FIGURE 4.3: Normalized transit traffic savings for ISP 1 vs. its cache upload capacity κ_1. The incoming transit traffic savings $(\rho_1^I|_{\kappa_1=0} - \rho_1^I)$ are normalized by the incoming transit traffic without caching, $\rho_1^I|_{\kappa_1=0}$. The values for the outgoing transit traffic savings are calculated similarly, i.e., $(\rho_1^O|_{\kappa_1=0} - \rho_1^O)/\rho_1^O|_{\kappa_1=0}$.

4.3.3 Model Validation

Before analyzing the effects of the caches on the amount of transit traffic we show simulation and experiment results to validate the simple model of transit traffic. We use the same scenarios as for the validation of the system dynamics (cf. Section 4.2.3) and consider the transit traffic savings, i.e., the difference of the transit traffic without and with caching. We distinguish between incoming transit traffic savings $\rho_1^I|_{\kappa_1=0} - \rho_1^I$ and outgoing transit traffic savings $\rho_1^O|_{\kappa_1=0} - \rho_1^O$. Figures 4.3(a) and 4.3(b) show the incoming and outgoing transit traffic savings normalized by the corresponding transit traffic values without caching, $\rho_1^I|_{\kappa_1=0}$ and $\rho_1^O|_{\kappa_1=0}$ respectively. Consequently, the values in these figures can also be interpreted as the fraction of incoming and outgoing transit traffic that can be saved by installing a cache with upload capacity κ_1.

The simulations and experiments confirm that the model provides accurate estimates of the transit traffic as long as the system is clearly download-rate-limited (Figure 4.3(a)). However, for values of κ_1 close to the transition between an upload-rate-limited system and a download-rate-limited system the difference between the model and the simulation results gets bigger, up to 25%. Further increasing κ_1 the analytical and simulation results get closer as the system becomes dominantly download-rate-limited. The reason is again that due to the changing peer population there are some periods of time when the system is download-rate-limited although it is upload-rate-limited on average. When the peer population is small, the cache can not use its total upload capacity and leechers obtain a larger fraction of the file from other peers.

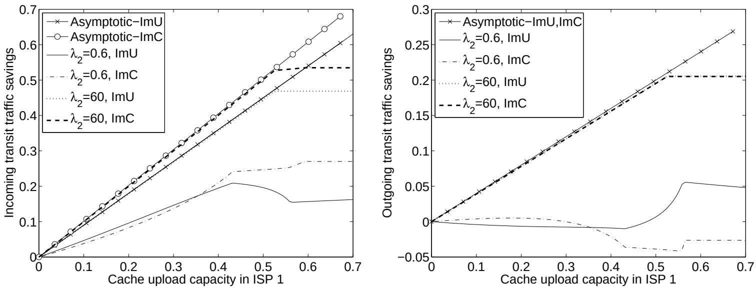
Like in Section 4.2.3, we carry out simulations for a larger swarm ($\lambda_1 = 3$, $\lambda_2 = 30$) with homogeneous and with heterogeneous peer access speeds. The transit traffic savings for these scenarios are presented in Figure 4.3(b). Again, we conclude that the model is more accurate for larger peer populations and that there is hardly any difference between the results for homogeneous and for heterogeneous peer upload speeds. The model overestimates the incoming transit traffic savings for the scenario with heterogeneous peer upload and download speeds. The reason is that the model underestimates the number of leechers x_1 for this scenario (cf. Figure 4.2(b)), which has a big impact on the incoming transit traffic.

4.3.4 Numerical Results and Insights

In the following, we show numerical results based on the simple model of the transit traffic and show that an accurate model of the system dynamics is necessary when investigating the impact of caches on the transit traffic. We present non-normalized transit traffic values in order to be able to show the asymptotic results.

Numerical Results

Figure 4.4(a) shows the savings in terms of incoming transit traffic as a function of the cache upload rate κ_1 for ISP 1. The parameters are the same as the ones used for Figure 4.2(c). For *ImU* the decrease of the incoming transit traffic is



(a) Incoming traffic savings $\rho_1^I|_{\kappa_1=0} - \rho_1^I$. $\lambda_1 = 0.6$.

(b) Outgoing traffic savings $\rho_1^O|_{\kappa_1=0} - \rho_1^O$. $\lambda_1 = 0.6$.

FIGURE 4.4: Analytical results for transit traffic savings of ISP 1 vs. its cache upload capacity κ_1.

always below the amount of cache upload rate used, while for *ImC* it is equal. The asymptotic bounds are rather tight both for *ImU* and for *ImC* until the system becomes download-rate-limited. Once the system is download-rate-limited, the increase of the cache upload rate has only a minor effect on the incoming transit traffic.

There is a big difference in the efficiency of the caches for different values of the arrival rate λ_2 in ISP 2. The decrease of the incoming transit traffic is less than 50% of the cache upload rate for $\lambda_1 = \lambda_2$, while it is close to the asymptotic limit for $\lambda_2 = 60$. The inefficiency of the cache to decrease the incoming transit traffic for swarms for which a significant portion of the peers is in the ISP shows that ISPs might have to actively manage the cache upload rates between the different swarms to maximize the cache efficiency. Based on this result, we develop a cache upload capacity allocation policy that prioritizes swarms with a small fraction of peers inside the ISP with the cache in Section 4.5. Furthermore, we evaluate its efficiency via simulations of multi-swarm scenarios and show that it can outperform the default, demand-driven policy, i.e, if the upload capacity of the cache is not actively managed.

Figure 4.4(b) shows the savings in terms of outgoing transit traffic as a function of the cache upload rate κ_1. The parameters are the same as the ones used for Figure 4.2(c). Surprisingly, we observe that the outgoing transit traffic increases slightly for low values of κ_1. The increase of the outgoing transit traffic is in fact a result of the increase of the number of seeds and the decrease of the number of leechers in ISP 1. The changes in the number of the peers and cache upload rate results in an indirect feeding of the leechers in ISP 2. This phenomenon is the reason for the low efficiency in decreasing the outgoing transit traffic even for $\lambda_2 = 60$. The asymptotic bounds are rather tight both for *ImU* and for *ImC*.

These results suggest that a transparent cache is rather efficient in terms of decreasing the incoming transit traffic compared to an *ImU*. With the availability of localization services the deployment of *ImC* can become possible, which can improve the efficiency of non-transparent peer-to-peer caches.

Fluid Modeling vs. Static Overlay

Our simple model of transit traffic is of course not accurate and complex enough to predict the amount of transit traffic in a complex, heterogeneous network, but it can serve to compare the amount of transit traffic if one considers the effects of caches on the system dynamics and if one does not consider them.

Figure 4.5 shows the mismatch of the estimate of the transit traffic savings if one did not use the fluid model described in Section 4.2 to model the change of the number of peers as a function of the cache upload rate, but used the number of peers without a cache to estimate the transit traffic as a function of the cache upload rate using Equations 4.22 and 4.23. The figure shows that one underestimates the decrease of the incoming transit traffic by almost up to a factor of 20 if one does not consider the change of the number of peers. At the same time one overestimates the decrease of the outgoing transit traffic by up to a factor of 10. The actual ratios depend on the considered scenario, but in general, the error introduced by not modeling the change of the number of peers can be substantial.

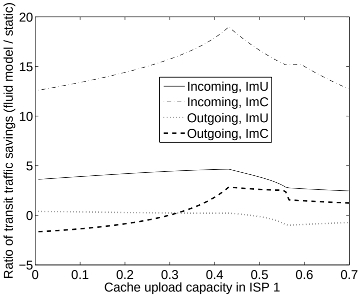


FIGURE 4.5: Ratio of the estimated transit traffic savings using the fluid model and without using it. $\lambda_1 = \lambda_2 = 0.6$.

4.4 Improving the Cache Efficiency

In the previous sections we showed two controversial effects of caching. First, under certain scenarios the upload rate provided by the cache is not entirely used to decrease the transit traffic of the ISP. Second, under certain scenarios the cache upload rate can lead to an increase of the ISP's outgoing traffic contrary to the expectations. In the following we investigate how restricted neighbor selection (RNS) could help to avoid these effects. The idea behind RNS is to prevent seeds from indirectly relaying the cache's upload rate to external leechers. To achieve this, the seeds follow a proximity-aware upload policy: they only upload to local leechers as long as there are any. Leechers may still upload to remote peers. This simple scheme ensures that small swarms scattered over several ISPs do not starve in the presence of seeds.

In the following we first describe a possible implementation of RNS in BitTorrent-based P2P systems. Then, we adapt our model of the system dynamics and the transit traffic to RNS and validate it via simulations and experiments. Finally, we investigate how such a simple scheme could improve the cache's efficiency.

4.4.1 Implementation of RNS in BitTorrent

In BitTorrent the so-called choke algorithm determines to which other peers a peer uploads data. A possible implementation of RNS could change this algorithm in a way such that a seed prefers local leechers over remote leechers. The required information whether another peer is local or remote can be obtained using ISP provided localization services developed in the IETF Application Layer Traffic Optimization (ALTO) working group [62]. Another source for this information are public databases such as [91].

Nevertheless, if peers know only a randomly selected, small subset (e.g., around 50 peers in some BitTorrent implementations) of all peers in the swarm, it might happen that a seed has no direct connection to local leechers even if local leechers are present in the swarm. In this case the seed would upload to remote leechers which leads to inter-ISP traffic. To avoid this situation, we modify the BitTorrent clients so that seeds keep track of the number of local leechers in their neighbor set. If this number reaches 0, they contact the tracker to obtain addresses of more local leechers. For this purpose, the tracker needs to know the AS affiliations of the peers. A tracker supporting such a mechanism is for example the so-called *iTracker* which is proposed and investigated in [66]. This scheme ensures that the data delivery from the seeds to the leechers is kept as local as possible.

4.4.2 System Dynamics under RNS

In the following we develop a fluid model of the system dynamics for RNS. We use the same notation as in Section 4.2.2. We keep the assumptions that the cache operated by ISP i only serves leechers in ISP i, and that leechers upload and download data to and from all peers (i.e., they are proximity unaware), but impose the limitation that seeds only upload to local leechers.

The upload rate available to leechers in ISP i has three sources: the cache provided by ISP i, the leechers in all ISPs and the seeds *local* to ISP i. The cache upload rate in ISP i is κ_i. The upload rate from the local seeds is μy_i. The total upload rate from leechers in the system is $\mu(\eta \sum_{j\in\mathcal{I}} x_j)$. Since the upload rate from the leechers is shared among all $\sum_{j\in\mathcal{I}} x_j$ leechers, the total upload rate available to the x_i leechers in ISP i is $U_i(\mathbf{x},\mathbf{y},\kappa) = \mu(\eta x_i + y_i) + \kappa_i$. Note that this expression of $U_i(\mathbf{x},\mathbf{y},\kappa)$ is the same as that for the single system studied in Section 4.2.1. Consequently, the number of leechers and seeds in the ISPs is the same as if the ISPs were isolated.

When the system in ISP i is upload-rate-limited (i.e., $cx_i \geq U_i(\mathbf{x},\mathbf{y},\kappa)$) we have

$$\overline{x}_i = \frac{\lambda_i}{\nu\left(1+\frac{\theta}{\nu}\right)} - \frac{\kappa_i}{\mu\eta\left(1+\frac{\theta}{\nu}\right)} \tag{4.32}$$

$$\overline{y}_i = \frac{\lambda_i}{\gamma\left(1+\frac{\theta}{\nu}\right)} + \frac{\kappa_i\theta}{\mu_i\eta\gamma\left(1+\frac{\theta}{\nu}\right)}. \tag{4.33}$$

When the system is download-rate-limited the number of leechers and seeds is the same as without restricted neighbor selection, i.e., given by Equations 4.17 and 4.18, respectively. We observe that with restricted neighbor selection the system dynamic in ISP i is not influenced by the cache upload rates of the other ISPs. Using the steady-state number of leechers and seeds the condition for the system to be upload-rate-limited in ISP i is

$$\kappa_i \leq \frac{\lambda_i\{c(\gamma-\mu) - \gamma\eta\mu\}}{\gamma(\theta+c)}, \tag{4.34}$$

identical to that of the single system case. Whether the system is upload or download-rate-limited depends only on the cache upload rate κ_i of ISP i.

4.4.3 Transit Traffic Estimates under RNS

We can obtain the transit traffic estimates for the case of restricted neighbor selection by defining the publicly available upload rate in ISP i as the upload rate of the leechers $u_i^P = \mu(\eta x_i)$, and by defining the locally available upload rate as the sum of the upload rates of the seeds and the cache upload rate $u_i^L = \mu y_i + \kappa_i$.

With these definitions of the available upload rates we can use Equations 4.19-4.23 to approximate the incoming and the outgoing transit traffic in the ISPs.

We can derive an asymptotic upper bound for the outgoing transit traffic for the case of restricted neighbor selection similar to the one in Section 4.3.2. Following the same steps, but substituting Equations 4.32 and 4.33 into Equation 4.27, for the case of the *ImU* we get

$$\lim_{\lambda_2 \to \infty} (\rho_1^O|_{\kappa_1=0} - \rho_1^O) = \frac{\kappa_1}{\left(1 + \frac{\theta}{\nu}\right)}. \tag{4.35}$$

Comparing Equation 4.35 to Equation 4.25 we observe an increase of the bound of the outgoing transit traffic gain due to the restriction of the neighbor selection of the seeds. The condition $1 + \frac{\theta}{\nu} \geq 1$ still holds however, so that the outgoing transit traffic gains are less than the installed cache upload rate.

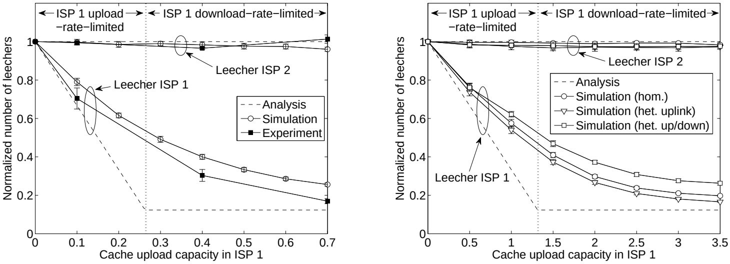
For the case of the *ImC* we substitute Equations 4.32 and 4.33 into Equation 4.31, and get

$$\lim_{\lambda_2 \to \infty} (\rho_1^O|_{\kappa_1=0} - \rho_1^O) \quad = \quad \frac{\kappa_1}{\left(1 + \frac{\theta}{\nu}\right)}. \tag{4.36}$$

Comparing Equation 4.29 to Equation 4.36 we observe that the rightmost term disappears, and hence the upper bound of the outgoing transit traffic gain is higher.

4.4.4 Model Validation

In order to validate the model for RNS, we use the same scenarios as for the unrestricted neighbor selection (cf. Section 4.2.3). The change of the number of leechers is shown in Figure 4.6. Figure 4.6(a) is analogue to Figure 4.2(a) and compares results obtained from the simulations and the experiments for the scenario with $\lambda_1 = 0.6$ and $\lambda_2 = 6$. The simulation and experimental results confirm that the model accurately captures the impact of the cache on the number of leechers in ISP 1 as long as the cache capacity is small. However, in the range of



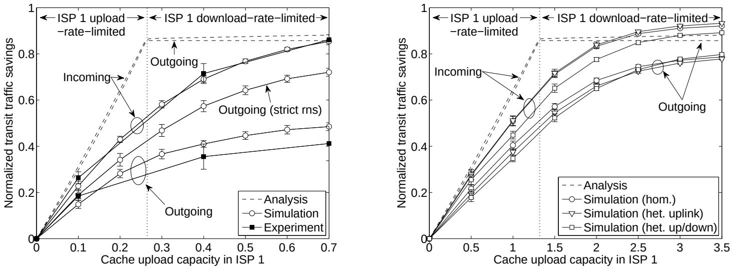
(a) Comparison of analysis, simulations, and experiments.

(b) Comparison of analysis and simulations ($\lambda_1 = 3$, $\lambda_2 = 30$, homogeneous and heterogeneous access speeds of the peers).

FIGURE 4.6: Normalized number of leechers $\frac{\overline{x}_i}{\overline{x}_i|_{\kappa_1=0}}$ and seeds $\frac{\overline{y}_i}{\overline{y}_i|_{\kappa_1=0}}$ as a function of the cache upload capacity κ_1 of ISP 1 for the restricted neighbor selection.

$\kappa_1 \in [0.2, 0.4]$ the model underestimates the number of leechers in ISP 1 considerably. The reason is that sometimes no leecher exists in ISP 1. As a consequence, the cache capacity cannot be fully utilized in this scenario which is neglected by our model. According to the simulations, no leecher is present in ISP 1 for around 14% of the steady-state simulation time in case of $\kappa_1 = 0.4$ and the utilization of the cache upload capacity is about 76%. The experiment with $\kappa_1 = 0.4$ shows that this effect can also be observed using real BitTorrent clients. In addition, the number of leechers in ISP 2 observed in simulation and experiments remains almost constant regardless of the cache capacity κ_1 in ISP 1. The slight decrease can be explained by the fact that sometimes seeds in ISP 1 upload to ISP 2 since no leechers are present in ISP 1.

Figure 4.6(b) corresponds to Figure 4.2(b) and presents the results for a larger swarm ($\lambda_1 = 3$, $\lambda_2 = 30$) with homogeneous and heterogeneous access bandwidth as introduced in Section 4.2.3. As for the case with the unrestricted neighbor selection, the simulation results show that the accuracy of our model is higher for larger swarms. Furthermore, heterogeneous access bandwidths have only a



(a) Comparison of analysis, simulations, and experiments.

(b) Comparison of analysis and simulations ($\lambda_1 = 3$, $\lambda_2 = 30$, homogeneous and heterogeneous access speeds of the peers).

FIGURE 4.7: Normalized transit traffic savings for ISP 1 vs. its cache upload capacity κ_1 in case of restricted neighbor selection. The incoming transit traffic savings $(\rho_1^I|_{\kappa_1=0} - \rho_1^I)$ are normalized by the incoming transit traffic without caching, $\rho_1^I|_{\kappa_1=0}$. The values for the outgoing transit traffic savings are calculated similarly, i.e., $(\rho_1^O|_{\kappa_1=0} - \rho_1^O)/\rho_1^O|_{\kappa_1=0}$.

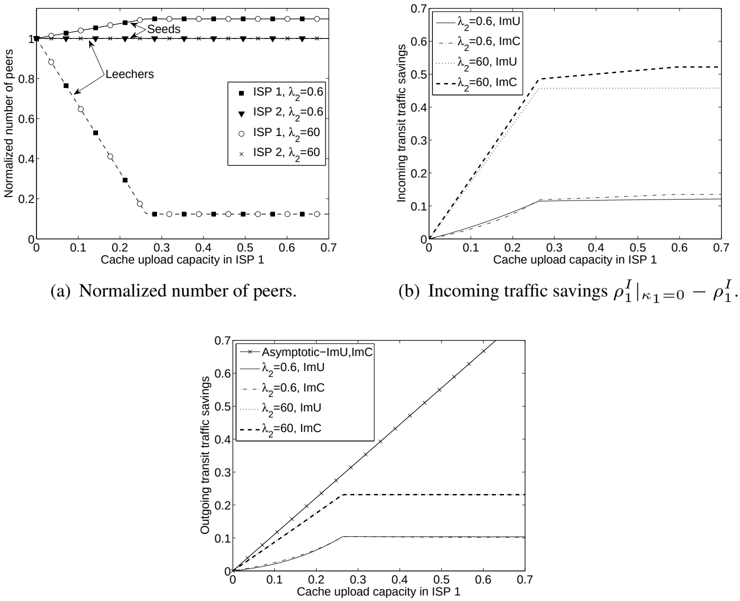
small impact on the number of leechers in ISP 1 and no impact on leechers in ISP 2.

In order to validate our model of the inter-ISP traffic, we consider the normalized transit traffic savings in analogy to Section 4.3.3. In Figure 4.7(a), we compare the results obtained from the model, the simulations, and the experiments. While the model predicts that the normalized savings in incoming and outgoing traffic are very similar, the simulation results and the experiments show that the normalized savings in incoming traffic are higher than those in outgoing traffic. The reason is that seeds in ISP 1 upload to ISP 2 when no leechers are present in ISP 1 whereas the model assumes that the whole upload capacity of seeds is used for local leechers. Therefore, we simulate an additional peer behavior where seeds never upload to remote leechers. The corresponding savings in outgoing traffic (labeled "Outgoing (strict rns)" in Figure 4.7(a)) are significantly closer to the predictions by the model. However, this peer behavior can lead to starvation in swarms scattered over several ISPs and is therefore unlikely to be used in practice. For small and large cache capacities, the model provides accurate predictions of the incoming traffic savings. The overestimation of the traffic savings for $\kappa_1 \in [0.2, 0.5]$ is owed to the underestimation of the number of leechers explained above since this number mainly determines the amount of transit traffic. As for the unrestricted neighbor selection, the accuracy of the model for transit traffic increases considerably for larger swarms (cf. Figure 4.7(b)). Furthermore, the simulations of the scenarios with homogeneous access capacities of the peers, with heterogeneous upload capacities, and with heterogeneous upload and download capacities lead to similar results. Therefore, we conclude that heterogeneity of access capacities has only a minor impact on transit traffic savings under these circumstances.

4.4.5 Numerical Results and Insights

We first consider the system dynamics with RNS. In Figure 4.8(a), the number of leechers and seeds for RNS is shown as a function of the cache upload capacity κ_1. As already pointed out, the number of peers in ISP 2 and their arrival rate has no impact on the system dynamics in ISP 1. Furthermore, the number of peers in ISP 2 is not influenced by the cache capacity κ_1 of ISP 1. As a consequence, the normalized number of seeds and leechers in ISP 2 remains constant at a value of 1. Finally, we observe that less cache capacity κ_1 is required to reach the download-rate-limited state due to the RNS policy.

Figure 4.8(b) shows the incoming transit traffic savings of ISP 1 for RNS obtained from the model. Comparing the figure to Figure 4.4(a) we observe that incoming transit traffic savings are higher with RNS than without it for $\lambda_2 = 60$. While the savings in terms of incoming transit traffic are about as large as the cache capacity for the unrestricted neighbor selection, they are almost doubled with RNS for the *ImU* and the *ImC*. In contrast, the traffic savings with RNS are slightly below the ones without RNS for $\lambda_2 = \lambda_1 = 0.6$. However, this does not mean that RNS leads to more incoming transit traffic. It can be explained by the fact that seeds do not upload to remote leechers when RNS is applied even when



(a) Normalized number of peers.

(b) Incoming traffic savings $\rho_1^I|_{\kappa_1=0} - \rho_1^I$.

(c) Outgoing traffic savings $\rho_1^O|_{\kappa_1=0} - \rho_1^O$.

FIGURE 4.8: Analytical results for the number of peers (a) and the incoming (b) and outgoing transit traffic savings of ISP 1 (c) vs. its cache upload capacity κ_1 in case of restricted neighbor selection.

no cache is used ($\kappa_1 = 0$). Therefore, the incoming transit traffic $\rho_i^I|_{\kappa_1=0}$ with RNS is already considerably lower than without it even when no cache is used. Hence, the savings in incoming transit traffic achieved by the additional cache capacity can be smaller than with unrestricted neighbor selection, although the incoming transit traffic for a given cache capacity with RNS is lower than without RNS for any value of κ_1. The outgoing transit traffic savings of ISP 1 obtained from the model are shown in Figure 4.8(c) for the case with RNS. Again, we present non-normalized transit traffic savings to show the asymptotic limits. Comparing the figure to Figure 4.4(b) we observe that restricting the neighbor selection of seeds eliminates the unwanted increase of the outgoing transit traffic. In general, the outgoing transit traffic savings increase as an effect of RNS both for *ImU* and *ImC*.

4.5 Outlook on Multi-Swarm Scenarios

As shown above, it is not straightforward to assess the impact of caches on the amount of inter-ISP traffic: the influence of the cache on the swarm dynamics and the peer selection mechanisms need to be taken into account to assess the actual benefit. As a consequence, the same cache upload capacity allocated to different swarms can lead to different savings in terms of transit traffic depending on the swarm parameters. Since the savings in terms of transit traffic is a function of the swarm parameters, it is important to understand whether the amount of transit traffic can be decreased by actively allocating the cache upload capacity to the peers in the swarms that coexist in an ISP's network.

Therefore, we study different types of allocation policies in this section. First, we give a precise formulation of the upload capacity allocation problem. Afterwards, we describe a set of allocation policies. Finally, we evaluate the performance of these policies via simulations. The content of this section is taken from [23] with some modifications and extension from [24, 29].

4.5.1 Cache Upload Capacity Allocation Problem

We consider a scenario with a set $\mathcal{I} = \{1, 2, \ldots, I\}$ of I ISPs and a set $\mathcal{S}$ of swarms. Each ISP $i \in \mathcal{I}$ operates a cache with upload capacity κ_i. We denote the cache capacity allocation of ISP i by the vector $K_i = (K_{i,1}, \ldots, K_{i,S})$, with $\sum_{s \in \mathcal{S}} K_{i,s} \leq \kappa_i$. It is known that the cache upload capacity $K_{i,s}$ allocated to swarm s has an impact on the transit traffic of ISP i (cf. Section 4.3). It is, however, not well understood how the cache capacity allocation K_i between swarm affects the amount of transit traffic of ISP i. Our goal is to answer this question.

4.5.2 Cache Capacity Allocation Policies

In order to evaluate whether cache upload capacity allocation can lead to a decrease of the inter-ISP traffic, we consider various policies that ISPs could implement to allocate the available cache upload capacity among the swarms.

Demand-driven Allocation

The baseline cache capacity allocation policy we consider is when the cache capacity is not actively managed by the ISP. The cache serves the requests of the peers according to a first in first out service discipline. As the request rate can exceed the maximum service rate of the cache, the cache is equipped with a drop tail queue in which requests can be stored until they get served. The cache capacity allocated to a swarm s is approximately proportional to the demand that the leechers of swarm s in ISP i put on the cache. Consequently, we refer to this baseline policy as the *demand-driven* policy.

Priority-Based Policies

Intuitively, if cache capacity allocation is to lead to decreased transit traffic compared to the demand-driven policy, then there must be some swarm s whose transit traffic decreases faster as a function of the amount of cache capacity $K_{i,s}$ than that of the other swarms. If there is such a swarm s then the cache capacity

$K_{i,s}$ allocated to it should be increased until the marginal gain of allocating more cache capacity to it equals to the marginal gain of the other swarms. Nevertheless, finding the optimal allocation this way is difficult in practice, because the marginal gain of allocating more cache capacity to a swarm is hard to measure.

To approximate the theoretically optimal policy we consider policies that use priorities to allocate the cache capacity between the swarms. The cache serves the requests of peers using non-preemptive priority scheduling. Various ways are possible for assigning priorities to swarms. In our case, we choose the following priority assignment: to calculate the priority of a swarm s we take the ratio r of the number of leechers $x_{i,s}$ in ISP i to the total number of peers in s outside ISP i and assign the highest priority to the swarm with the smallest ratio r. As a consequence, we call this scheme *smallest-ratio priority*. The motivation of this policy focuses on the incoming traffic savings since these tend to be significantly large than the outgoing savings.

Leechers in swarms with a small ratio r are likely to download data from a remote peer. This is in particular the case under Assumption 2, which states that the distribution of sources over ISPs for a single byte downloaded is proportional to the amount of upload capacity exposed to the downloading peer. In contrast, the amount of traffic that a leecher downloads from a remote peer is smaller if only a small fraction of the peers in this swarm is located outside of ISP i, i.e., in a swarm with a high ratio r. Hence, we argue that the prioritizing swarms with small ratios r in the allocation of the cache upload capacity is a reasonable measure to increase transit traffic savings.

Policies Based on Per-Swarm Capacity Limits

As a comparison for the smallest-ratio priority and the demand-driven policy we consider policies that reserve a fraction of the cache capacity to individual swarms. Such policies were evaluated in [24]. Three capacity reservation schemes are considered there. The first scheme reserves the same amount of cache capacity to all swarms, referred to as *uniform capacity reservation*. The second and third schemes reserve capacity to swarm s proportional to the ratio of local and external number of peers in the swarm, analogous to the priority-based policy. In our evaluations in [24] we showed, that capacity limits can reduce transit traffic, but they tend to waste a part of the upload capacity of the cache. The reason is that upload capacity allocated to a certain swarm cannot be consumed by other swarms even if this swarms is not able to use all the allocated upload capacity. Since this is different with priority-based policies, we consider only those in the remainder in this section.

4.5.3 Performance of Allocation Policies

Our performance evaluation is based on simulations. For the evaluation we consider a BitTorrent system consisting of $|\mathcal{S}| = 12$ swarms. Motivated by the results in Section 4.2.2, we consider a network topology consisting of two ISPs, called ISP 1 and ISP 2. Peers arrive to swarm s in ISP i according to a Poisson process with rate $\lambda_{i,s}$. After downloading the file, they remain in the swarm for an exponentially distributed seeding time. The aggregate arrival rate λ of the peers in all swarms is 0.5 s^{-1} in all scenarios, ISP 1 uses a cache with an upload capacity $\kappa_1 = 30$ Mbit/s. No caches are located in ISP 2.

We implemented the cache capacity allocation policies in the ProtoPeer simulator, which we also used preceding part of this chapter. We are first interested in understanding under which conditions allocation policies have an impact on the amount of transit traffic. For that purpose, we define three scenarios with different distributions of the peers over the swarms and ISPs and measure the transit traffic savings in incoming and outgoing direction, i.e., the fraction of transit traffic that can be saved by installing a cache as compared to the case without a cache.

In the first scenario the peers are distributed uniformly between swarms, i.e., $\lambda_s = \sum_i \lambda_{i,s} = \lambda/|\mathcal{S}|$, but the arrival intensity to ISP 2 is ten times higher than that to ISP 1 (scenario called *uniform,1:10*). The transit traffic savings for the demand-driven and the smallest-ratio priority policies are presented in Figure 4.9. We observe that cache capacity allocation does not make a difference for this scenario. The same is true for the second set of columns, results ob-

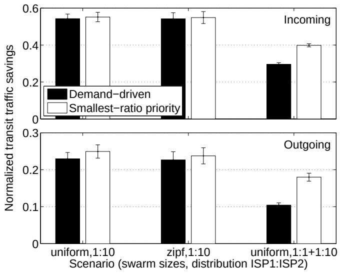


FIGURE 4.9: Transit traffic savings for three different scenarios.

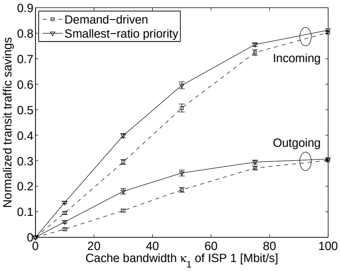


FIGURE 4.10: Impact of cache upload capacity on traffic savings.

tained for a scenario where the arrival intensities λ_s follow a Zipf-distribution, i.e., when some swarms are more popular than others (scenario called *zipf,1:10*). We only observe a difference in terms of the transit traffic savings in the third set of columns. These results correspond again to a uniform distribution of peers over swarms, i.e., $\lambda_s = \sum_i \lambda_{i,s} = \lambda/|\mathcal{S}|$. However, in this scenario there are two symmetric swarms (with respect to their distribution over ISP 1 and ISP 2, i.e., $\lambda_{1,s} = \lambda_{2,s}$), and ten asymmetric swarms where the arrival intensities are distributed 1:10 over ISP 1 and ISP 2 (scenario called *uniform,1:1+1:10*). This means for the asymmetric swarms that $\lambda_{2,s} = 10 \cdot \lambda_{1,s}$.

The results show that the smallest-ratio priority policy performs best; it outperforms the demand-driven policy by almost 30 percent in incoming direction and by almost 50 percent in outgoing direction. The reason is that when the ratio of local leechers is low, peers are likely to download data from peers in other ISPs. Therefore, they should be prioritized by caches. These results indicate that upload capacity allocation is an efficient means of improving the efficiency of P2P caches.

In addition, we investigate the impact of the upload capacity κ_1 of the cache in ISP 1 in this third scenario with the ten asymmetric swarms. To this end, we vary κ_1 from 0 to 100 Mbit/s and measure the incoming and outgoing transit traffic savings for the demand-driven and the smallest-ratio priority allocation. The results are shown in Figure 4.10. We observe that up to 80 percent of the incoming transit traffic can be saved and up to 30 percent of outgoing transit traffic. In both directions the smallest-ratio priority policy clearly outperforms the demand-driven allocation. In particular for medium cache upload capacities $\kappa_1 = 30$ Mbit/s or $\kappa_1 = 50$ Mbit/s, there is a considerable difference in transit traffic savings comparing the two different allocations. For larger cache upload capacities this difference diminishes since nearly all peers can download data at full speed. This decreases the impact of prioritization.

These simulations show that actively managing the upload capacity of a cache can increase transit traffic savings for ISPs. For that purpose, the smallest-ratio priority policy is a reasonable policy since it leads significantly larger traffic savings compared to demand-driven allocation in the considered scenarios. The described motivation for the smallest-ratio priority allocation suggests that this is also the case for a wide range of other scenarios and when popularity of swarms or their distribution among ISPs change over time. A more comprehensive evaluation of upload capacity allocation policies can be found in [29]. This paper also includes a formulation of this problem as a Markov decision process and presents results from Planet-Lab experiments validating the simulation results.

4.6 Lessons Learned

In this chapter we start with single-swarm scenarios of BitTorrent-like peer-to-peer systems and investigate the impact of caches on the inter-ISP traffic. To this end, we develop a simple fluid model of the effects of caches on the system dynamics and show using the model how the caches installed in an ISP affect the system-wide and the local peer-dynamics. Furthermore, we describe a simple model of inter-ISP traffic and use the model to illustrate that the major impact of caches on the transit traffic is via the system dynamics. Hence, one can not neglect the effects of caches on the system dynamics. We provide asymptotic bounds on the efficiency of caches, and give a comparison of the efficiency of caches under our modeling assumptions. We show that caches can sometimes lead to increased outgoing transit traffic, depending on the portion of the peers within the ISP.

In addition, we describe a restricted neighbor selection policy, extended the fluid model to capture its effect on the system dynamics, and show that it can avoid the increase of the outgoing transit traffic due to caching. Our analytical results also show that ISP managed Caches would in general be superior to transparent caches and to ISP managed Ultrapeers in terms of decreasing the transit traffic, except for very small torrents when the difference is negligible. We validate the insights obtained via the fluid model by simulations and experiments with real BitTorrent clients. While the quantitative results on the inter-ISP traffic depend on the traffic model, we expect that the qualitative results would hold for other traffic models.

Finally, we zoom our focus to multi-swarm scenarios and investigate whether ISPs can decrease their inter-ISP traffic by actively managing the upload capacity of their caches, i.e., allocating more upload capacity to some swarms than to other ones. This investigation is motivated by the fact that the same amount of cache upload capacity leads to different amounts of traffic savings depending on the swarm parameters. For this purpose, we present the smallest-ratio priority policy and show via simulations that this policy can outperform the demand-driven allocation in the investigated scenarios by up to 50 percent. Therefore, we conclude that an active management of the upload capacity of a cache is important to reduce transit traffic for ISPs.

5 Conclusion

Overlay networks based on the P2P paradigm are an efficient means to distribute content in the Internet. They are more scalable compared to the traditional client/server architecture since users who download a specific content also contribute upload capacity to the distribution process. As a consequence, the available upload capacity scales with the demand in the system. This is in particular appealing for content providers since they can reduce their distribution costs through the use of the upload capacity of their customers.

However, the distribution process within the overlay networks is mostly unaware of the underlying physical network topology and of network boundaries between different ISPs. Therefore, the distribution process might be less efficient from the point of view of the ISP. For example, several users in the overlay network who are all located in the physical network of the same ISP might download identical content from other users located in different parts of the world instead of sharing this content between each other. This behavior leads to a considerable amount of inter-ISP traffic, which is costly for small and medium-sized ISPs. In addition, such traffic is often unnecessary in the sense that it can be avoided without degrading the user perceived performance if users in the same ISP preferentially share content among each other. Due to the high popularity of such overlay networks, this is a severe cost factor for many ISPs today.

As a consequence, the research community and standardization bodies are trying to optimize the traffic patterns resulting from P2P based overlay networks for content distribution. The main goal is to reduce the amount of inter-ISP traffic without degrading the user experience. For that purpose, two basic approaches are under discussion. The first one is locality-awareness. This concept equips overlay participants with knowledge about the physical network topology so that they can preferentially exchange data with others that are close in the physical network. The other approach is *caching*. This means that the ISP in the example mentioned above saves a copy of the requested content in a cache when the content is downloaded the first time in its network. All subsequent requests to this content within the network of this ISP can then be answered by the cache.

In this monograph, we study the performance of caching techniques in overlays for content distribution. As example of an overlay network we choose BitTorrent, which is currently the most popular overlay for that purpose. We first investigate the structure of overlay networks in today's Internet and use the resulting characterization to estimate the optimization potential of locality-awareness in the current Internet. Second, we model the impact of caching on BitTorrent-like P2P networks and derive estimates for the reduction of inter-ISP traffic. Based on the insights gained from the model we develop allocation policies for the upload capacity of the cache which improve the traffic savings.

The investigation of the nature of current overlay networks presented in Chapter 3 is based on a large scale measurement study, which comprises a large number of swarms from different torrent index servers. To obtain the peers participating in the swarms, we implement a distributed tracker monitoring system. The measurement results contain information about the size of the networks, i.e., number of peers per swarm, and its change over time, the size of the shared files and the distribution of peers over ASes. In addition, we study characteristics of swarms sharing regional content and provide separate statistics for different types of content, i.e., music or movie files.

From these measurements we derive statistical characterizations of BitTorrent swarms. This is an important contribution to the research community and the standardization process because it permits to define realistic scenarios for the performance evaluation of traffic optimization mechanisms for these networks. In this way, it guarantees that evaluation results are obtained in real-world scenarios. Furthermore, we use our measurement results to make the following conclusions about the optimization potential of locality-awareness. In most of the swarms, the average number of peers per AS is below 2, which suggests that locality-awareness cannot be effective in saving inter-ISP traffic. However, we observe in addition that most of the BitTorrent peers are located in a small number of very large swarms. In fact, about 80 % of all peers participate in the top 20 % of the swarms. In these swarms, several ASes exist with a large number of peers per AS so that locality-awareness can achieve high inter-ISP traffic reductions of up to 42 % or 66 % of the total BitTorrent traffic, depending on which data set of the measurements we use, the one of music or the one of movie files, respectively.

In Chapter 4 we study caching as a means to reduce inter-ISP traffic resulting from BitTorrent networks. For that purpose, we take a deterministic fluid model of the number of leechers and seeders in a single swarm from literature and adapt it to capture the impact of caches. We then study how caches change those numbers and derive an estimate of the inter-ISP traffic that can be saved by using a cache. The estimates of the model are compared to simulations and experiments in controlled environment to assess the accuracy of the model.Therefore, our model is highly relevant for ISPs since it allows them to dimension their caches and to compare the cost of caches in terms of capital and operational expenditures to the expected financial savings due to inter-ISP traffic traffic reduction.

One of the major insights that we gain from the model is that the same amount of upload capacity of the cache results in different amounts of traffic savings when allocated to different swarms. This means that the characteristics of the swarms determine the efficiency of the cache. Motivated by this fact, we develop allocation policies for the upload capacity of the cache and investigate their performance in multi-swarm scenarios. Our simulations show that the cache should preferentially serve requests of peers in swarms which have a large share of peers outside the ISP providing the cache. This is plausible since those peers tend to download most of the content from remote locations. This strategy can increase the savings in inter-ISP traffic by up to 50 % in our scenarios compared to a demand-driven upload allocation. This is a very promising result for ISPs since it paves the way to further reduce inter-ISP traffic and inter-connection costs without provisioning more resources in terms of cache upload capacity.

In this respect, the work presented in this monograph improves the understanding of overlay networks for content distribution and it represents an important step towards further optimization possibilities in the future. During the next years the importance of video-on-demand and real-time video is expected to grow and it will soon surpass the one of pure file-sharing. The insights developed in this monograph serve as useful basis to design and evaluate content distribution mechanisms in the future Internet since the fundamental concepts, i.e., to distribute increasing amounts of data to a large number of users in an efficient way, hold for file-sharing as well as for video services.

Bibliography of the Author

— Book Chapters —

[1] 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 *Towards the Future Internet – A European Research Perspective* (G. Tselentis, J. Domingue, A. Galis, A. Gavras, D. Hausheer, S. Krco, V. Lotz, and T. Zahariadis, eds.), IOS Press Books Online, 2009.

[2] I. Papafili, G. D. Stamoulis, R. Stankiewicz, S. Oechsner, K. Pussep, R. Wojcik, J. Domzal, D. Staehle, F. Lehrieder, and B. Stiller, "Assessment of Economic Management of Overlay Traffic: Methodology and Results", in *The future Internet* (J. Domingue, A. Galis, A. Gavras, T. Zahariadis, and D. Lambert, eds.), Berlin, Heidelberg: Springer-Verlag, 2011.

— Journal Papers —

[3] M. Menth and F. Lehrieder, "PCN-Based Measured Rate Termination", *Computer Networks*, Vol. 54, No. 13, 2010.

[4] M. Menth, F. Lehrieder, B. Briscoe, P. Eardley, T. Moncaster, J. Babiarz, A. Charny, X. J. Zhang, T. Taylor, K.-H. Chan, D. Satoh, R. Geib, and G. Karagiannis, "A Survey of PCN-Based Admission Control and Flow

Termination", *IEEE Communications Surveys & Tutorials*, Vol. 12, No. 3, 2010.

[5] T. Hoßfeld, F. Lehrieder, D. Hock, S. Oechsner, Z. Despotovic, W. Kellerer, and M. Michel, "Characterization of BitTorrent Swarms and their Distribution in the Internet", *Computer Networks*, Vol. 55, No. 5, 2011.

[6] F. Lehrieder, S. Oechsner, T. Hoßfeld, D. Staehle, Z. Despotovic, W. Kellerer, and M. Michel, "Mitigating Unfairness in Locality-Aware Peer-to-Peer Networks", *International Journal of Network Management*, Vol. 21, No. 1, 2011.

[7] M. Menth and F. Lehrieder, "PCN-Based Marked Flow Termination", *Computer Communications*, Vol. 34, No. 17, 2011.

[8] F. Lehrieder, G. Dán, T. Hoßfeld, S. Oechsner, and V. Singeorzan, "Caching for BitTorrent-like P2P Systems: A Simple Fluid Model and its Implications", *IEEE/ACM Transactions on Networking*, Vol. 20, No. 4, 2012.

[9] M. Menth and F. Lehrieder, "Performance of PCN-Based Admission Control under Challenging Conditions", *IEEE/ACM Transactions on Networking*, Vol. 20, No. 2, 2012.

[10] K. Pussep, F. Lehrieder, C. Gross, S. Oechsner, M. Guenther, and S. Meyer, "Cooperative Traffic Management for Video Streaming Overlays", *Computer Networks*, Vol. 56, No. 3, 2012.

— Conference Papers —

[11] M. Menth, J. Milbrandt, and F. Lehrieder, "Algorithms for Fast Resilience Analysis in IP Networks", in *Proceedings of the International Workshop on IP Operations and Management (IPOM)*, 2006.

[12] J. Milbrandt, M. Menth, and F. Lehrieder, "A Priori Detection of Link Overload due to Network Failures", in *Tagungsband der Konferenz Kommunikation in Verteilten Systemen (KiVS)*, 2007.

[13] M. Menth and F. Lehrieder, "Comparison of Marking Algorithms for PCN-Based Admission Control", in *Proceedings of the GI/ITG Conference on Measurement, Modeling, and Evaluation of Computer and Communication Systems (MMB)*, 2008.

[14] M. Menth and F. Lehrieder, "Performance Evaluation of PCN-Based Admission Control", in *Proceedings of the International Workshop on Quality of Service (IWQoS)*, 2008.

[15] F. Lehrieder and M. Menth, "Marking Conversion for Pre-Congestion Notification", in *Proceedings of the International Conference on Communications (ICC)*, 2009.

[16] F. Lehrieder and M. Menth, "PCN-Based Flow Termination with Multiple Bottleneck Links", in *Proceedings of the International Conference on Communications (ICC)*, 2009.

[17] S. Oechsner, F. Lehrieder, T. Hoßfeld, F. Metzger, K. Pussep, and D. Staehle, "Pushing the Performance of Biased Neighbor Selection through Biased Unchoking", in *Proceedings of the IEEE International Conference on Peer-to-Peer Computing (P2P)*, 2009.

[18] P. Cholda, J. Domzal, R. Wójcik, R. Stankiewicz, F. Lehrieder, T. Hoßfeld, S. Oechsner, and V. Singeorzan, "Performance Evaluation of P2P Caches: Flash-Crowd Case", in *Proceedings of the Australasian Telecommunication Networks and Applications Conference (ATNAC)*, 2010.

[19] F. Lehrieder, G. Dán, T. Hoßfeld, S. Oechsner, and V. Singeorzan, "The Impact of Caching on BitTorrent-like Peer-to-peer Systems", in *Proceedings of the IEEE International Conference on Peer-to-Peer Computing (P2P)*, 2010.

[20] F. Lehrieder, S. Oechsner, T. Hoßfeld, Z. Despotovic, W. Kellerer, and M. Michel, "Can P2P-Users Benefit from Locality-Awareness?", in *Pro-*

ceedings of the IEEE International Conference on Peer-to-Peer Computing (P2P), 2010.

[21] S. Oechsner, F. Lehrieder, and D. Staehle, “Overlay Connection Usage in BitTorrent Swarms”, in *Proceedings of the Workshop on Economic Traffic Management, collocated with International Teletraffic Congress (ITC)*, 2010.

[22] P. Racz, S. Oechsner, and F. Lehrieder, “BGP-based Locality Promotion for P2P Applications”, in *Proceedings of the International Conference on Computer Communication Networks (ICCCN)*, 2010.

[23] F. Lehrieder, V. Pacifici, and G. Dán, “On the Benefits of P2P Cache Capacity Allocation”, in *Proceedings of the International Teletraffic Congress (ITC), Student Poster Session*, 2011.

[24] I. Papafili, G. D. Stamoulis, F. Lehrieder, B. Kleine, and S. Oechsner, “Cache Capacity Allocation to Overlay Swarms”, in *Proceedings of the International Workshop on Self-Organizing Systems (IWSOS)*, 2011.

[25] R. Stankiewicz, P. Cholda, J. Domzal, R. Wojcik, T. Hoßfeld, T. Zinner, S. Oechsner, and F. Lehrieder, “Influence of Traffic Management Solutions on Quality of Experience for Prevailing Overlay Applications”, in *Proceedings of the Conference on Next Generation Internet Networks (NGI)*, 2011.

[26] V. Burger, F. Lehrieder, T. Hoßfeld, and J. Seedorf, “Who Profits from Peer-to-Peer File-Sharing? Traffic Optimization Potential in BitTorrent Swarms”, in *Proceedings of the International Teletraffic Congress (ITC)*, 2012.

[27] M. Hirth, F. Lehrieder, S. Oberste-Vorth, T. Hoßfeld, and P. Tran-Gia, “Wikipedia and its Network of Authors from a Social Network Perspective”, in *Proceedings of the International Conference on Communications and Electronics (ICCE)*, 2012.

[28] M. Jarschel, F. Lehrieder, Z. Magyari, and R. Pries, “A Flexible OpenFlow-Controller Benchmark”, in *Proceedings of the European Workshop on Software Defined Networks (EWSDN)*, 2012.

[29] V. Pacifici, F. Lehrieder, and G. Dán, “Cache Capacity Allocation for BitTorrent-like Systems to Minimize Inter-ISP Traffic”, in *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM)*, 2012.

General References

[30] D. P. Heyman and M. J. Sobel, “Stochastic Models in Operations Research, Volume I”, 1982.

[31] A. Chankhunthod, P. B. Danzig, C. Neerdaels, M. F. Schwartz, and K. J. Worrell, “A Hierarchical Internet Object Cache”, in *Proceedings of the USENIX Annual Technical Conference*, 1996.

[32] B. Duska, D. Marwood, and M. J. Feeley, “The Measured Access Characteristics of World-Wide-Web Client Proxy Caches”, in *Proceedings of the USENIX Symposion on Internet Technologies and Systems*, 1997.

[33] P. M. Lurie and M. S. Goldberg, “An Approximate Method for Sampling Correlated Random Variables From Partially-Specified Distributions”, *Management Science*, Vol. 44, No. 2, 1998.

[34] G. Huston, “Interconnection, Peering and Settlements – Part I”, *Internet Protocol Journal*, Vol. 2, No. 1, 1999.

[35] K. Kong and D. Ghosal, “Mitigating Server-side Congestion in the Internet through Pseudoserving”, *IEEE/ACM Transactions on Networking*, Vol. 7, No. 4, 1999.

[36] W. A. Sethares and T. W. Staley, “Periodicity Transforms”, *IEEE Transactions on Signal Processing*, Vol. 47, No. 11, 1999.

[37] L. Fan, P. Cao, J. Almeida, and A. Z. Broder, “Summary Cache: A Scalable Wide-area Web Cache Sharing Protocol”, *IEEE/ACM Transactions on Networking*, Vol. 8, No. 3, 2000.

[38] L. Gao, “On Inferring Autonomous System Relationships in the Internet”, *IEEE/ACM Transactions on Networking*, Vol. 9, No. 6, 2001.

[39] N. Leibowitz, A. Bergman, R. Ben-Shaul, and A. Shavit, “Are File Swapping Networks Cacheable? Characterizing P2P Traffic”, in *Proceedings of the International Workshop on Web Content Caching and Distribution (WCW)*, 2002.

[40] B. Cohen, “Incentives Build Robustness in BitTorrent”, in *Proceedings of the Workshop on Economics of Peer-to-Peer Systems (P2PECON)*, 2003.

[41] Gnutella Protocol Development, “Gnutella – A Protocol for a Revolution.” Web page, 2003. http://rfc-gnutella.sourceforge.net.

[42] M. Izal, G. Urvoy-Keller, E. W. Biersack, P. A. Felber, A. A. Hamra, and L. Garces-Erice, “Dissecting BitTorrent: Five Months in a Torrent’s Lifetime”, in *Proceedings of the of Passive and Active Measurement Conference (PAM)*, 2004.

[43] D. Qiu and R. Srikant, “Modeling and Performance Analysis of BitTorrent-like Peer-to-peer Networks”, *ACM SIGCOMM Computer Communincation Review*, Vol. 34, No. 4, 2004.

[44] A. Wierzbicki, N. Leibowitz, M. Ripeanu, and R. Woźniak, “Cache Replacement Policies for P2P File Sharing Protocols”, *European Transactions on Telecommunications*, Vol. 15, No. 6, 2004.

[45] X. Yang and G. de Veciana, “Service Capacity of Peer to Peer Networks”, in *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM)*, 2004.

[46] F. Clévenot-Perronnin, P. Nain, and K. W. Ross, “Multiclass P2P Networks: Static Resource Allocation for Service Differentiation and Bandwidth Diversity”, *Performance Evaluation*, Vol. 62, No. 1, 2005.

[47] L. Guo, S. Chen, Z. Xiao, E. Tan, X. Ding, and X. Zhang, “Measurements, Analysis, and Modeling of BitTorrent-like Systems”, in *Proceedings of the ACM Internet Measurement Conference (IMC)*, 2005.

[48] T. Karagiannis, P. Rodriguez, and K. Papagiannaki, “Should Internet Service Providers Fear Peer-Assisted Content Distribution?”, in *Proceedings of the ACM Internet Measurement Conference (IMC)*, 2005.

[49] J. Pouwelse, P. Garbacki, D. Epema, and H. Sips, “The Bittorrent P2P Filesharing System: Measurements and Analysis”, in *Proceedings of the International Workshop on Peer-to-Peer Systems (IPTPS)*, 2005.

[50] “BitTorrent Specification.” Web page, 2006. http://wiki.theory.org/BitTorrentSpecification.

[51] R. Bindal, P. Cao, W. Chan, J. Medved, G. Suwala, T. Bates, and A. Zhang, “Improving Traffic Locality in BitTorrent via Biased Neighbor Selection”, in *Proceedings of the IEEE International Conference on Distributed Computing Systems (ICDCS)*, 2006.

[52] 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.

[53] Y. Tian, D. Wu, and K. W. Ng, “Modeling, Analysis and Improvement for BitTorrent-like File Sharing Networks”, in *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM)*, 2006.

[54] V. Aggarwal, A. Feldmann, and C. Scheideler, “Can ISPs and P2P Users Cooperate for Improved Performance?”, *ACM SIGCOMM Computer Communincation Review*, Vol. 37, No. 3, 2007.

[55] X. Dimitropoulos, D. Krioukov, M. Fomenkov, B. Huffaker, Y. Hyun, k. claffy, and G. Riley, “AS Relationships: Inference and Validation”, *ACM SIGCOMM Computer Communincation Review*, Vol. 37, No. 1, 2007.

[56] J. Kobza, S. Jacobson, and D. Vaughan, “A Survey of the Coupon Collector’s Problem with Random Sample Sizes”, *Methodology and Computing in Applied Probability*, Vol. 9, No. 4, 2007.

[57] 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 International Teletraffic Congress (ITC)*, 2007.

[58] PlanetLab, “An Open Platform for Developing, Deploying, and Accessing Planetary-scale Services.” Web page, 2007. http://www.planet-lab.org.

[59] D. R. Choffnes and F. E. Bustamante, “Taming the Torrent: a Practical Approach to Reducing Cross-ISP Traffic in Peer-to-Peer Systems”, *ACM SIGCOMM Computer Communincation Review*, Vol. 38, No. 4, 2008.

[60] German-Lab Project, “National Platform for Future Internet Studies.” Web page, 2008. http://www.german-lab.de.

[61] M. Hefeeda and O. Saleh, “Traffic Modeling and Proportional Partial Caching for Peer-to-peer Systems”, *IEEE/ACM Transactions on Networking*, Vol. 16, No. 6, 2008.

[62] IETF, “Chartor of the Working Group on Application-Layer Traffic Optimization (ALTO).” Web page, 2008. http://datatracker.ietf.org/wg/alto/charter.

[63] A. Loewenstern, "DHT Protocol." Web page, 2008. http://www.bittorrent.org/beps/bep_0005.html.

[64] I. Rimac, A. Elwalid, and S. Borst, "On Server Dimensioning for Hybrid P2P Content Distribution Networks", in *Proceedings of the IEEE International Conference on Peer-to-Peer Computing (P2P)*, 2008.

[65] Sourceforge.net, "ProtoPeer." Web page, 2008. http://sourceforge.net/projects/protopeer.

[66] H. Xie, Y. R. Yang, A. Krishnamurthy, Y. G. Liu, and A. Silberschatz, "P4P: Provider Portal for Applications", *ACM SIGCOMM Computer Communincation Review*, Vol. 38, No. 4, 2008.

[67] W. Galuba, K. Aberer, Z. Despotovic, and W. Kellerer, "ProtoPeer: A P2P Toolkit Bridging the Gap Between Simulation and Live Deployment", in *Proceedings of the International Conference on Simulation Tools and Techniques (SIMUTools)*, 2009.

[68] T. Hoßfeld, D. Hock, S. Oechsner, F. Lehrieder, Z. Despotovic, W. Kellerer, and M. Michel, "Measurement of BitTorrent Swarms and their AS Topologies", Technical Report TR463, University of Würzburg, Würzburg, Germany, 2009.

[69] S. Khirman, "Torrent AS Localisation - Raw Data." Web page, 2009. http://www.khirman.com/blog/as_raw_data.

[70] M. Piatek, H. V. Madhyastha, J. P. John, A. Krishnamurthy, and T. Anderson, "Pitfalls for ISP-Friendly P2P Design", in *Proceedings of the ACM Workshop on Hot Topics in Networks (HotNets)*, 2009.

[71] H. Schulze and K. Mochalski, "Internet Study 2008/2009", 2009.

[72] H. Wang, J. Liu, and X. Ke, "On the Locality of BitTorrent-based Video File Swarming", in *Proceedings of the International Workshop on Peer-to-Peer Systems (IPTPS)*, 2009.

[73] R. Cuevas, M. Kryczka, A. Cuevas, S. Kaune, C. Guerrero, and R. Rejaie, "Is Content Publishing in BitTorrent Altruistic or Profit-driven?", in *Proceedings of the International Conference on Emerging Networking Experiments and Technologies (CoNEXT)*, 2010.

[74] IETF, "Charter of the Working Group on Decoupled Application Data Enroute (decade)." Web page, 2010. http://datatracker.ietf.org/wg/decade/charter.

[75] OverSi, "OverCache P2P." Web page, 2010. http://www.oversi.com/en/products/overcache-msp/overcache-p2p.html.

[76] A. Rao, A. Legout, and W. Dabbous, "Can Realistic BitTorrent Experiments Be Performed on Clusters?", in *Proceedings of the IEEE International Conference on Peer-to-Peer Computing (P2P)*, 2010.

[77] S. Ren, E. Tan, T. Luo, S. Chen, L. Guo, and X. Zhang, "TopBT: A Topology-Aware and Infrastructure-independent BitTorrent Client", in *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM)*, 2010.

[78] C. Zhang, P. Dhungel, D. Wu, and K. W. Ross, "Unraveling the BitTorrent Ecosystem", *IEEE Transactions on Parallel and Distributed Systems*, Vol. 22, No. 7, 2010.

[79] "Gtk-Gnutella." Web page, 2011. http://gtk-gnutella.sourceforge.net.

[80] S. L. Blond, A. Legout, and W. Dabbous, "Pushing BitTorrent Locality to the Limit", *Computer Networks*, Vol. 55, No. 3, 2011.

[81] R. Cuevas, N. Laoutaris, X. Yang, G. Siganos, and P. Rodriguez, "Deep Diving into BitTorrent Locality", in *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM)*, 2011.

[82] G. Dán, "Cache-to-Cache: Could ISPs Cooperate to Decrease Peer-to-peer Content Distribution Costs?", *IEEE Transactions on Parallel and Distributed Systems*, Vol. 22, 2011.

[83] M. Kryczka, R. Cuevas, A. Cuevas, C. Guerrero, and A. Azcorra, "Measuring BitTorrent Ecosystem: Techniques, Tips and Tricks", *IEEE Communications Magazine*, Vol. 49, No. 9, 2011.

[84] "Akamai." Web page, 2012. http://www.akamai.com.

[85] R. Alimi, R. Penno, and Y. Yang, "ALTO Protocol v12." IETF Internet Draft, 2012. http://tools.ietf.org/html/draft-ietf-alto-protocol-12.

[86] N. Carlsson, G. Dán, A. Mahanti, and M. Arlitt, "A Longitudinal Characterization of Local and Global BitTorrent Workload Dynamics", in *Proceedings of the of Passive and Active Measurement Conference (PAM)*, 2012.

[87] I. Cisco Systems, "Cisco Visual Networking Index: Forecast and Methodology, 2011–2016." White paper, 2012.

[88] S. Kim, J. Han, T. Chung, H. chul Kim, T. Kwon, and Y. Choi, "Content Publishing and Downloading Practice in BitTorrent", in *Proceedings of the IFIP Networking*, 2012.

[89] MaxMind, "Geolocation Service." Web page, 2012. http://www.maxmind.com.

[90] PeerApp, "UltraBand Series." Web page, 2012. http://www.peerapp.com/Products/UltraBand.aspx.

[91] Team Cymru, "IP to ASN Mapping." Web page, 2012. http://www.team-cymru.org/Services/ip-to-asn.html.

[92] The Cooperative Association for Internet Data Analysis (CAIDA), "AS Ranking." Web page, 2012. http://as-rank.caida.org.

ISSN 1432-8801