# Caching for BitTorrent-like P2P Systems: A Simple Fluid Model and its Implications

Frank Lehrieder*, György Dán‡, Tobias Hoßfeld*, Simon Oechsner*, Vlad Singeorzan*
*University of Würzburg, Institute of Computer Science, Würzburg, Germany
‡ KTH Royal Institute of Technology, School of Electrical Engineering, ACCESS Linnaeus Centre, Stockholm, Sweden

***Abstract*—Peer-to-peer file-sharing systems are responsible for a significant share of the traffic between Internet service providers (ISPs) in the Internet. In order to decrease their peer-to-peer related transit traffic costs, many ISPs have deployed caches for peer-to-peer traffic in recent years. We consider how the different types of peer-to-peer caches – caches already available on the market and caches expected to become available in the future – can possibly affect the amount of inter-ISP traffic. We develop a fluid model that captures the effects of the caches on the system dynamics of peer-to-peer networks, and show that caches can have adverse effects on the system dynamics depending on the system parameters. We combine the fluid model with a simple model of inter-ISP traffic and show that the impact of caches cannot be accurately assessed without considering the effects of the caches on the system dynamics. We identify scenarios when caching actually leads to increased transit traffic. Motivated by our findings, we propose a proximity-aware peer selection mechanism that avoids the increase of the transit traffic and improves the cache efficiency. We support the analytical results by extensive simulations and experiments with real BitTorrent clients.**

***Keywords*-Peer-to-peer, caching, fluid model**

## I. Introduction

Peer-to-peer (P2P) file-sharing systems are one of the major sources of Internet traffic. They generate an estimated 40 to 70% of the total traffic depending on geographic location [1], and are expected to remain a significant source of traffic in the future [2]. For the users P2P file-sharing systems provide access to a large variety of content, and for content providers they provide a means to distribute data to a large population of users without the need for big investments in server and network resources. The costs of the content distribution are shared among the end users and their Internet service providers (ISPs). The protocols of the most popular P2P file-sharing systems were not designed to be aware of the network topology, and consequently P2P applications generate a large amount of inter-ISP traffic.

Increased inter-ISP traffic is a potential source of revenues for ISPs at the top of the ISP hierarchy (called tier-1 ISPs). Their main concern is to keep the traffic to their peering tier-1 ISPs balanced. Nevertheless, for ISPs in the lower levels of the ISP hierarchy (tier-2 and tier-3 ISPs), which are usually charged by their transit traffic providers, transit traffic is a source of costs, and hence is something to be kept low.

The research community has been trying to address the issue of inter-ISP traffic caused by proximity-unaware protocols in two ways. First, by introducing proximity-awareness in the most popular file-sharing protocols, and by trying to understand its effects on the application performance [3], [4]. Second, by proposing localization services for P2P protocols that would make proximity-aware protocols more efficient from the ISPs' point of view [5], [6]. While these approaches could yield a significant decrease of the inter-ISP traffic, there is no evidence yet of the widespread use of proximity-awareness in deployed systems.

ISPs have been addressing the issue of increased transit traffic by deploying commercially available caches for P2P traffic [7], [8]. P2P caches decrease the transit traffic by storing popular contents locally in the ISP so that they do not have to be downloaded from remote peers [9]. The caches provided by the different vendors, e.g., PeerApp's UltraBand and OverSi's OverCache P2P, follow fundamentally different design principles, yet all of them promise substantial savings in terms of inter-ISP traffic.

The question we address in this paper 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. In order 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 classes of peers, 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 classes of peers is sufficient to model multiple classes of peers. 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

This work was partially funded by the EU FP7 Network of Excellence Euro-NF through the Specific Joint Research Project "ISPeer" and by the Federal Ministry of Education and Research of the Federal Republic of Germany (Förderkennzeichen 01 BK 0800, GLab). The authors alone are responsible for the content of the paper.

support our results.

The rest of the paper is organized as follows. We discuss the related work in Sect. II. Sect. III briefly describes the relevant details of BitTorrent-like systems and the different P2P cache designs. We develop the fluid model of the effects of caches on the system dynamics in Sect. IV, and illustrate its importance in predicting the ISP transit traffic in Sect. V. We describe and evaluate a scheme to improve the cache efficiency in Sect. VI. In Sect. VII we conclude the paper.

## II. Related Work

There has been a significant amount of work on caching of P2P contents. The focus of those works was on the achievable cache hit ratios [10], [11], and on the efficiency of various caching algorithms [9], [11], [12]. However, inferring the amount of saved inter-ISP traffic directly from cache hit ratios is only possible if (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. We show in this paper that these two effects can have a major impact on the inter-ISP traffic in current BitTorrent-like P2P networks. To this end, we model the impact of caches on the population of a swarm and derive a model of the resulting inter-ISP traffic. Our model focuses on a single swarm and does therefore not account for the disk space of the cache and cache hit ratios. These questions are complementary and were already discussed in the literature [9]–[12].

Most closely related to our work are the analytical models of the system dynamics of BitTorrent-like systems. In [13] the authors described the system dynamics with a Markov process and showed that the service capacity of P2P systems grows exponentially with the offered load. In [14] the authors described a deterministic fluid model for BitTorrent-Like P2P networks and validated it by simulations and data from real BitTorrent traces. The focus of [14] was on the scalability, performance and the efficiency of a P2P network independent of the network topology, and showed that the number of peers is finite under arbitrary load conditions. These observations were reaffirmed in [15] based on a probabilistic model. In [16] the fluid model of [14] was extended to two classes of peers in order to evaluate how the allocation of the peers' upload rates between classes affects the system performance. A model of the effect of churn rate and download completion ratio on the performance was presented in [17]. In [18] a fluid model was described to assist the dimensioning of server assisted hybrid P2P content distribution.

Our model is inspired by the fluid model of the service capacity and the number of peers in [14] and extends the model in two ways. First, we derive a model to capture the effects of caching on the system dynamics. Second, we provide a simple means to analyze the amount of inter-ISP traffic in scenarios with multiple ISPs. To our knowledge, our work is the first to derive a model that provides insights into the effects of caches on the system dynamics and on the inter-ISP traffic in BitTorrent-like P2P systems.

## III. Background and System Description

In this section we give a brief overview of the relevant details of BitTorrent-like file-sharing protocols and present the different types of P2P caches. Finally, we describe our system model of BitTorrent and the ISP level network topology.

### A. BitTorrent-like Protocols

In BitTorrent-like file-sharing protocols, content is divided into a large number of pieces, and the peers exchange the pieces with each other. This way peers that do not have the entire content, called leechers, can also utilize their upload capacity to distribute the content. Peers that already own the entire content are referred to as seeds. All peers that distribute the same content are usually called a swarm.

A peer can get to know other peers interested in the same content via a centralized tracker (in BitTorrent), via a DHT (in BitTorrent) or via an unstructured overlay (in Gnutella which uses the partial file sharing protocol PFSP). Typically, a peer knows about a subset of the peers in the swarm, its neighbors, and exchanges data with a subset of these neighbors. The set of peers with which data is exchanged is dynamically determined by the choking mechanism in BitTorrent [19], but is fixed in Gnutella. For a detailed description of BitTorrent we refer the reader to [19] and to [20].

### B. Taxonomy of P2P Caches

Caches for P2P traffic can be grouped into three main categories.

*1) 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.

*2) 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.

*3) 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 [21], and resource discovery (e.g., cache discovery) is considered for standardization in the IETF Application Layer Traffic Optimization (ALTO) [22] and DECoupled Application Data Enroute (DECADE) [23] 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.

### C. 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 $\kappa_i$ 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 $\lambda_i$. 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 [24], as it can be considered the superposition of a large number of renewal processes [25]. Leechers abort the download at rate $\theta$, 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 $\gamma$, 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., [14], [26]).

Peers have upload capacity $\mu$ 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 [14]. In the mathematical model we assume without loss of generality that the file size is 1, so that $\mu$,$c$ and $\kappa_i$ are normalized to the file size. For the sake of simplicity, we assume homogeneous peer capacities. Table I summarizes the notation used in the paper.

TABLE I
FREQUENTLY USED NOTATION.

| Parameter | Definition |
|---|---|
| $\mathcal{I}$,$I$ | Set and number of ISPs, respectively |
| $\kappa_i$ | Cache upload capacity in ISP $i$ |
| $\lambda_i$ | Arrival rate in ISP $i$ |
| $\theta$ | Abort rate of leechers |
| $\gamma$ | Departure rate of seeds |
| $\eta$ | Effectiveness of file sharing |
| $\mu$ | 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 $t$ |
| $\rho_i^I$ | Incoming transit traffic in ISP $i$ |
| $\rho_i^O$ | Outgoing transit traffic in ISP $i$ |

## IV. 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 [14], 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 $U_i(\mathbf{x},\mathbf{y},\boldsymbol{\kappa})$ 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 $\boldsymbol{\kappa} = (\kappa_1,\ldots,\kappa_I)$. 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 $min(cx_i, U_i(\mathbf{x},\mathbf{y},\boldsymbol{\kappa}))$. Following the assumptions used in [14] 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

$$\frac{dx_i(t)}{dt} = \lambda_i - \theta x_i(t) - min\{cx_i(t), U_i(\mathbf{x},\mathbf{y},\boldsymbol{\kappa})\} \quad (1)$$

$$\frac{dy_i(t)}{dt} = min\{cx_i(t), U_i(\mathbf{x},\mathbf{y},\boldsymbol{\kappa})\} - \gamma y_i(t). \quad (2)$$

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

$$\frac{dx_i(t)}{dt} = \frac{dy_i(t)}{dt} = 0 \qquad i = 1,\ldots,I. \quad (3)$$

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 V when estimating the transit traffic between the ISPs.

### A. 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 [14] 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

$$U(x,y,\kappa) = \mu(\eta x + y) + \kappa. \tag{4}$$

Substituting this into (1) and (2) we get for the steady-state

$$0 = \lambda - \theta\bar{x} - min\{c\bar{x}, \mu(\eta\bar{x}+\bar{y}) + \kappa\} \tag{5}$$

$$0 = min\{c\bar{x}, \mu(\eta\bar{x}+\bar{y}) + \kappa\} - \gamma\bar{y}. \tag{6}$$

Let us first consider the download rate limited case, when the available upload rate exceeds the maximum download rate of the leechers, i.e., $c\bar{x} \leq \mu(\eta\bar{x}+\bar{y}) + \kappa$. 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 [14]

$$\bar{x} = \frac{\lambda}{c(1+\frac{\theta}{c})} \tag{7}$$

$$\bar{y} = \frac{\lambda}{\gamma(1+\frac{\theta}{c})}. \tag{8}$$

The condition under which the download rate is the limit is however different from that in [14]. Given the expressions for the steady-state number of leechers (7) and seeds (8) it is

$$\kappa \geq \frac{\lambda\{c(\gamma-\mu) - \gamma\eta\mu\}}{\gamma(\theta+c)}. \tag{9}$$

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

$$\bar{x} = \frac{\lambda}{\nu\left(1+\frac{\theta}{\nu}\right)} - \frac{\kappa}{\mu\eta\left(1+\frac{\theta}{\nu}\right)} \tag{10}$$

$$\bar{y} = \frac{\lambda}{\gamma\left(1+\frac{\theta}{\nu}\right)} + \frac{\kappa\theta}{\mu\eta\gamma\left(1+\frac{\theta}{\nu}\right)}, \tag{11}$$

where $\frac{1}{\nu} = \frac{1}{\eta}(\frac{1}{\mu} - \frac{1}{\gamma})$. Again, given the steady-state number of leechers (10) and seeds (11) we can express the condition under which the upload rate is the limit

$$\kappa \leq \frac{\lambda\{c(\gamma-\mu) - \gamma\eta\mu\}}{\gamma(\theta+c)}. \tag{12}$$

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 (10) and (11) we draw the following conclusions.

- For $\kappa = 0$ the results coincide with those in [14], 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 $\kappa$. 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 $\theta/\eta\gamma(1+\frac{\theta}{\nu})$. 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.

### B. 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 $\kappa_i$. 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

$$U_i(\mathbf{x},\mathbf{y},\kappa) = \mu(\eta x_i + \sum_{j\in\mathcal{I}} y_j \frac{x_i}{\sum_{j\in\mathcal{I}} x_j}) + \kappa_i. \tag{13}$$

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 (1) and (2) for every $i \in \mathcal{I}$ and solve the system of equations to get the steady-state number of leechers and seeds

$$\bar{x}_i = \frac{\lambda_i}{\nu\left(1+\frac{\theta}{\nu}\right)} - \frac{\kappa_i}{\mu\eta\left(1+\frac{\theta}{\nu}\right)} - \Delta_i(\mathbf{x},\mathbf{y},\kappa) \tag{14}$$

$$\bar{y}_i = \frac{\lambda_i}{\gamma\left(1+\frac{\theta}{\nu}\right)} + \frac{\kappa_i\theta}{\mu\eta\gamma\left(1+\frac{\theta}{\nu}\right)} + \frac{\theta}{\gamma}\Delta_i(\mathbf{x},\mathbf{y},\kappa), \tag{15}$$

where

$$\Delta_i(\mathbf{x},\mathbf{y},\kappa) = \frac{\sum_{j\in\mathcal{I}}(\lambda_i\kappa_j - \kappa_i\lambda_j)}{\eta\gamma\left(1+\frac{\theta}{\nu}\right)\left(\sum_{j\in\mathcal{I}}(\lambda_j - \kappa_j)\right)}. \tag{16}$$

From (14) and (15) we can obtain the following insights:

- Increasing the cache upload rate $\kappa_i$ 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 V. To verify that (14) is a monotonically decreasing function of $\kappa_i$ in an upload rate limited system, we evaluate the first and second derivatives of (14) w.r.t. $\kappa_i$. Eq. (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 $\bar{x}_i<0$, 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 $\kappa_i$, the number of leechers decreases until we reach the download rate limited regime. A similar reasoning holds for (15).
- $\Delta_i(\mathbf{x},\mathbf{y},\boldsymbol{\kappa})$ given in (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 $\Delta_i(\mathbf{x},\mathbf{y},\boldsymbol{\kappa})$ and consequently $\bar{y}_i$ and $\bar{x}_i$ 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 $\sum_{i\in\mathcal{I}}\Delta_i(\mathbf{x},\mathbf{y},\boldsymbol{\kappa})=0$, we have that $\sum_{i\in\mathcal{I}}\bar{x}_i=\bar{x}$ as given in (10) and $\sum_{i\in\mathcal{I}}\bar{y}_i=\bar{y}$ as given in (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 IV-C 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., $cx_i\leq U_i(\mathbf{x},\mathbf{y},\boldsymbol{\kappa})$). Then the steady-state number of leechers and seeds in ISP $i$ is given by

$$\bar{x}_i = \frac{\lambda_i}{c(1+\frac{\theta}{c})} \quad (17)$$

$$\bar{y}_i = \frac{\lambda_i}{\gamma(1+\frac{\theta}{c})}. \quad (18)$$

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

### C. 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 IV-C3, we briefly describe our simulation methodology and our experiment methodology.

*1) Simulation Methodology:* We implemented the BitTorrent protocol in the ProtoPeer [27] 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 [28].

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 [29].

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 $\text{min}^{-1}$, 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 [14].

*2) Experiment Methodology:* All measurements are performed in the experimental facility of the German-Lab (G-Lab) project [30]. This experimental facility is distributed over 5 universities in Germany. It consists of 152 nodes running Planet-Lab [31] 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. [32] 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.

*3) 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 $\kappa^*$. 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 $\bar{x}_1$ and in the whole swarm $\bar{x}$ in Fig. 1. The figure shows that for a given aggregate cache capacity $\kappa^*$ 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. This is because 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 $\lambda^*$, 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 $\kappa^*$ and the aggregate arrival intensity $\lambda^*$ of the rest of the ISPs. Therefore, in the rest of the paper we focus on a scenario with

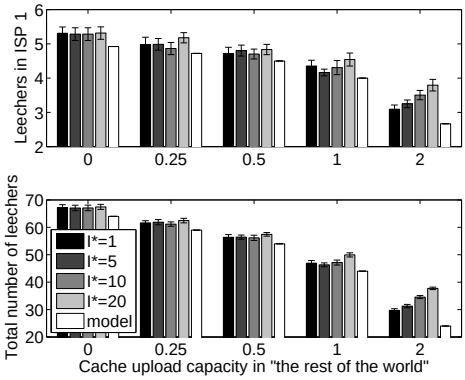


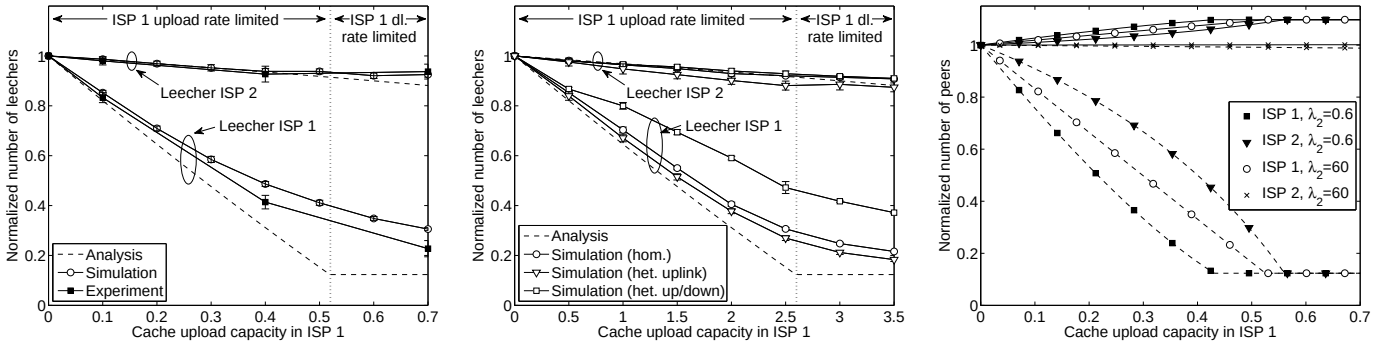

Fig. 1. Average number of leechers $\bar{x}_1$ in ISP 1 (top) and in the whole swarm $\bar{x}$ (bottom) for different numbers of other ISPs $I^*$ and aggregate cache capacities $\kappa^*$ in "the rest of the world". $\lambda_1 = 0.6$, $\kappa_1 = 0.1$, $\lambda^* = 6$.

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 $\kappa_1$ of ISP 1. We performed simulations and experiments with different values of $\kappa_1$, and measured the number of leechers and seeds.

In Fig. 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 $\bar{x}_i$ in ISP $i$ as a function of the cache upload capapcity $\kappa_1$ in ISP 1 normalized by the number of leechers $\bar{x}_i|_{\kappa_1=0}$ 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 $\bar{x}_1$ 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.

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: homogeneous peer upload and downloads speeds; heterogeneous peer upload speeds, and heterogeneous peer upload and download speeds. For the homogeneous scenario, we keep the default upload and download capacities (Sect. IV-C1) 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 sce-




(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).

Fig. 2. Normalized number of leechers $\frac{\bar{x}_i}{\bar{x}_i|_{\kappa_1=0}}$ and seeds $\frac{\bar{y}_i}{\bar{y}_i|_{\kappa_1=0}}$ as a function of the cache upload capacity $\kappa_1$ of ISP 1. The figures show the number of leechers $\bar{x}_i$ and seeds $\bar{y}_i$ in ISP $i$ divided by the corresponding values for the case without caching ($\bar{x}_i|_{\kappa_1=0}$ and $\bar{y}_i|_{\kappa_1=0}$).

nario. 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 Fig. 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 Fig. 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 $\kappa_1$, and a further increase of $\kappa_1$ does not reduce their download time and their number in the system.

### D. Numerical Results

The validation presented above allows us to consider two ISPs when evaluating the effects of the cache upload rate $\kappa_i$ 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.

Fig. 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 Figs. 2(a) and 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.

## V. The Importance of Fluid Modeling

In order 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 $\kappa_i$ ISP $i$ can decrease its incoming transit traffic $\rho_i^I$ by at least $\kappa_i$. 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 $\rho_i^I$ if ISP $i$ installed a transparent cache. The transparent cache serves requests that would generate incoming transit traffic, hence a cache upload rate of $\kappa_i$ decreases the amount of incoming traffic $\rho_i^I$ by $\kappa_i$. 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 $\rho_i^O$ is minimal. An alternative expectation can be that if ISP $i$ installs cache upload rate $\kappa_i$ then it decreases its total transit traffic $\rho_i^I + \rho_i^O$ by at least $\kappa_i$.

### A. 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

$$D_i^d = cx_i \left(1 - \frac{u_i^L}{\sum_j u_j^P + u_i^L}\right). \tag{19}$$

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 $\kappa_i$. 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

$$D_i^d = max(0, cx_i - \kappa_i)\left(1 - \frac{u_i^L - \kappa_i}{\sum_j u_j^P + u_i^L - \kappa_i}\right). \tag{20}$$

Since $u_i^L = \kappa_i$, whatever is not demanded from the *ImC* is demanded from the publicly available upload rate.

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

$$D_i^r = D_i^d min\left(1, \frac{\sum_j u_j^P}{\sum_j D_j^d}\right). \tag{21}$$

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*

$$\rho_i^I = D_i^r \left(1 - \frac{u_i^P}{\sum_j u_j^P}\right). \tag{22}$$

*where $D_i^r$ is defined in (19)-(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*

$$\rho_i^O = \sum_{j \neq i} \rho_j^I \frac{u_i^P}{\sum_{k \neq j} u_k^P}. \tag{23}$$

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.

### *B. Asymptotic Results of Cache Efficiency*

Motivated by the results of Section IV-B 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 $\kappa_1$ 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 $\lambda_2$ sufficiently large if $\mu\gamma\eta < c(\gamma - \mu)$ then the system is upload rate limited (see Eq. (12)).

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

$$\lim_{\lambda_2 \to \infty} (\rho_1^I|_{\kappa_1 = 0} - \rho_1^I) = \frac{\kappa_1}{\left(1 + \frac{\theta}{\nu}\right)} \tag{24}$$

$$\lim_{\lambda_2 \to \infty} (\rho_1^O|_{\kappa_1 = 0} - \rho_1^O) = \frac{\kappa_1}{\left(1 + \frac{\theta}{\nu}\right)} - \frac{\mu\kappa_1}{\gamma}. \tag{25}$$

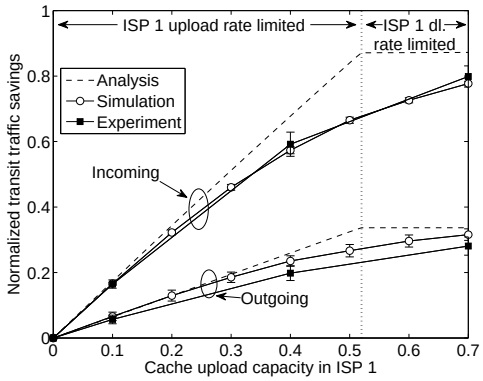
*Proof:* For an upload rate limited system and small cache upload rates $\kappa_i$ we can give an upper bound on the incoming transit traffic in ISP $i$ as $\frac{x_i}{\sum_{j \in \mathcal{I}} x_j}$ share of the total upload rate from leechers and seeds in all other ISPs $j \neq i$, i.e., $\sum_{j \neq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

$$\rho_i^I = \frac{x_i}{\sum_{j \in \mathcal{I}} x_j} \sum_{j \neq i} u_j^P. \tag{26}$$

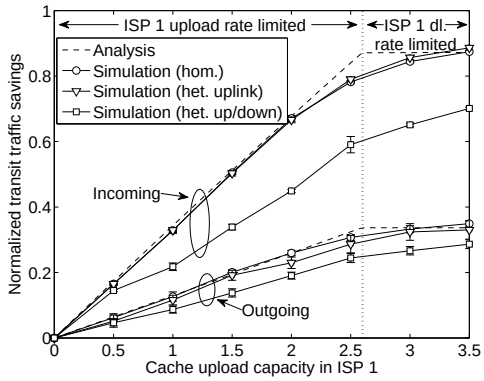
Substituting this expression into (23) we get an upper bound on the outgoing transit traffic intensity

$$\rho_i^O = \left(1 - \frac{x_i}{\sum_{j \in \mathcal{I}} x_j}\right) u_i^P. \tag{27}$$

Let us now substitute (14) and (15) into (26) and (27). By increasing the peer arrival rate in ISP 2 to infinity we get (24) and (25). ■




(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).

Fig. 3. Normalized transit traffic savings for ISP 1 vs. its cache upload capacity $\kappa_1$. 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., $(\rho_1^O|_{\kappa_1=0} - \rho_1^O)/\rho_1^O|_{\kappa_1=0}$.

Both expressions (24) and (25) are independent of the cache upload rate $\kappa_2$ 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*

$$\lim_{\lambda_2 \to \infty} (\rho_1^I|_{\kappa_1=0} - \rho_1^I) = \frac{\kappa_1}{\left(1+\frac{\theta}{\nu}\right)} + \frac{\kappa_1 \mu \eta}{c} \frac{\gamma}{(\gamma - \mu)} \quad (28)$$

$$\lim_{\lambda_2 \to \infty} (\rho_1^O|_{\kappa_1=0} - \rho_1^O) = \frac{\kappa_1}{\left(1+\frac{\theta}{\nu}\right)} - \frac{\mu \kappa_1}{\gamma}. \quad (29)$$

*Proof:* Consider the upper bound for the incoming transit traffic

$$\rho_i^I = \frac{cx_i - \kappa_i}{\sum_{j \in \mathcal{I}} cx_j - \kappa_j} \sum_{j \neq i} u_j^P, \quad (30)$$

and substitute this into (23) to get the upper bound on the outgoing transit traffic

$$\rho_i^O = \left(1 - \frac{cx_i - \kappa_i}{\sum_{j \in \mathcal{I}} cx_j - \kappa_j}\right) u_i^P. \quad (31)$$

We substitute (14) and (15) into (30) and (31) and increase the arrival rate in ISP 2 to infinity to get (28) and (29). ■

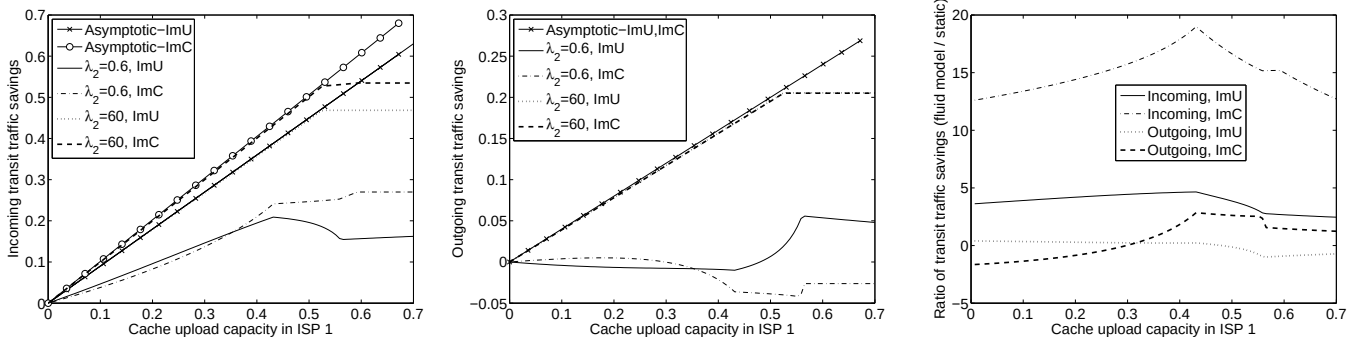
Again, the expressions are independent of $\kappa_2$, and the arrival intensity in ISP 1. Depending on the value of the rightmost term of (28) the efficiency of the cache upload rate for *ImC* can exceed 1. Consequently, an *ImC* can outperform a transparent cache in terms of the decrease of the incoming transit traffic. Comparing (24) to (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 (25) to (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*.

### C. 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. Sect. IV-C) 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 $\rho_1^O|_{\kappa_1=0} - \rho_1^O$. Figs. 3(a) and 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 Figs. 3(a) and 3(b) can also be interpreted as the fraction of incoming and outgoing transit traffic that can be saved by installing a cache with upload capacity $\kappa_1$.

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 (Fig. 3(a)). However, for values of $\kappa_1$ 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 $\kappa_1$ 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 Sect. IV-C, we perform 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 Fig. 3(b). Again, we conclude that the model is more accurate for larger peer populations and that there is hardly any difference between the




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

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

(c) Ratio of the estimated transit traffic savings using the fluid model and without using it. $\lambda_1 = \lambda_2 = 0.6$.

Fig. 4. Analytical results for transit traffic savings of ISP 1 vs. its cache upload capacity $\kappa_1$.

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. Fig. 2(b)), which has a big impact on the incoming transit traffic.

### D. 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.

*1) Numerical Results:* Fig. 4(a) shows the savings in terms of incoming transit traffic as a function of the cache upload rate $\kappa_1$ for ISP 1. The parameters are the same as the ones used for Fig. 2(c). For *ImU* the decrease of the incoming transit traffic is 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 $\lambda_2$ 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 our results the optimal cache capacity allocation would prioritize the swarms with the lowest ratio of peers inside the ISP. Nevertheless, in practice it might be difficult and resource intensive to estimate the ratio of leechers and seeds that are within the ISP for all swarms for which there are peers in the ISP. For this reason the optimal policy might be hard to implement. A detailed investigation of the optimal cache capacity allocation policy and its practical feasibility is beyond the scope of this study.

Fig. 4(b) shows the savings in terms of outgoing transit traffic as a function of the cache upload rate $\kappa_1$. The parameters are the same as the ones used for Fig. 2(c). Surprisingly, we observe that the outgoing transit traffic increases slightly for low values of $\kappa_1$. 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.

*2) 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.

Fig. 4(c) shows the mismatch of the estimate of the transit traffic savings if one did not use the fluid model described in Section IV 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 (22) and (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.

## VI. 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.

### A. 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 is consequently that a seed prefers local leechers in its choke algorithm 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 [22]. Another source for this information are public databases such as [33].

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-domain 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 [6]. This scheme ensures that the data delivery from the seeds to the leechers is kept as local as possible.

### B. 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 IV-B. 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 $\kappa_i$. The upload rate from the local seeds is $\mu y_i$. 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 $U_i(\mathbf{x},\mathbf{y},\boldsymbol{\kappa}) = \mu(\eta x_i + y_i) + \kappa_i$. Note that this expression of $U_i(\mathbf{x},\mathbf{y},\boldsymbol{\kappa})$ is the same as that for the single system studied in Section IV-A. 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., $cx_i \geq U_i(\mathbf{x},\mathbf{y},\boldsymbol{\kappa})$) we have

$$\bar{x}_i = \frac{\lambda_i}{\nu\left(1+\frac{\theta}{\nu}\right)} - \frac{\kappa_i}{\mu\eta\left(1+\frac{\theta}{\nu}\right)} \tag{32}$$

$$\bar{y}_i = \frac{\lambda_i}{\gamma\left(1+\frac{\theta}{\nu}\right)} + \frac{\kappa_i\theta}{\mu_i\eta\gamma\left(1+\frac{\theta}{\nu}\right)}. \tag{33}$$

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 (17) and (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

$$\kappa_i \leq \frac{\lambda_i\{c(\gamma-\mu)-\gamma\eta\mu\}}{\gamma(\theta+c)}, \tag{34}$$

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

### C. 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 (19)-(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 V-B. Following the same steps, but substituting (32) and (33) into (27), for the case of the *ImU* we get

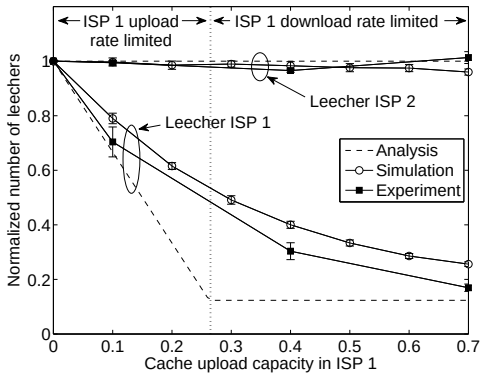
$$\lim_{\lambda_2\to\infty}(\rho_1^O|_{\kappa_1=0} - \rho_1^O) = \frac{\kappa_1}{\left(1+\frac{\theta}{\nu}\right)}. \tag{35}$$

Comparing (35) to (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.

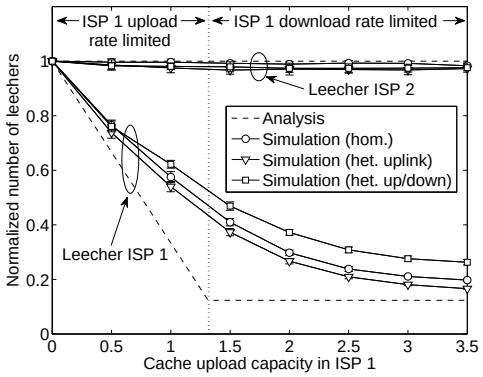
For the case of the *ImC* we substitute (32) and (33) into (31), and get

$$\lim_{\lambda_2\to\infty}(\rho_1^O|_{\kappa_1=0} - \rho_1^O) = \frac{\kappa_1}{\left(1+\frac{\theta}{\nu}\right)} \tag{36}$$

Comparing (29) to (36) we observe that the rightmost term disappears, and hence the upper bound of the outgoing transit traffic gain is higher.

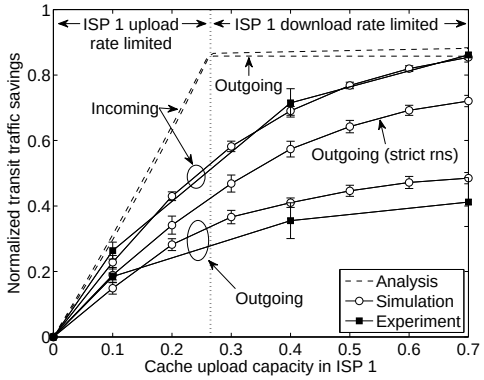



(a) Comparison of analysis, simulations, and experiments.

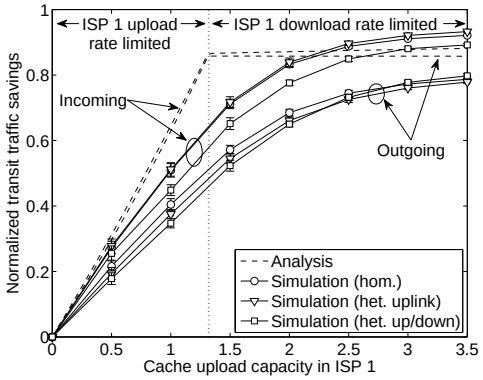



(b) Comparison of analysis and simulations ($\lambda_1 = 3$, $\lambda_2 = 30$, homogeneous and heterogeneous access speeds of the peers).

Fig. 5. Normalized number of leechers $\frac{\bar{x}_i}{\bar{x}_i|_{\kappa_1=0}}$ and seeds $\frac{\bar{y}_i}{\bar{y}_i|_{\kappa_1=0}}$ as a function of the cache upload capacity $\kappa_1$ of ISP 1 for the restricted neighbor selection.




(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).

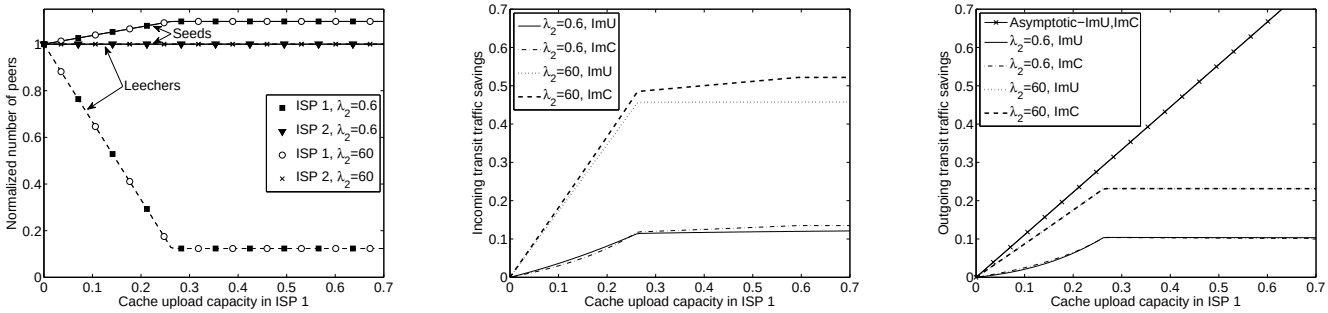
Fig. 6. Normalized transit traffic savings for ISP 1 vs. its cache upload capacity $\kappa_1$ in case of restricted neighbor selection. The incoming transit traffic savings $(\rho_1^I|_{\kappa_1=0} - \rho_1^I)$ are normalized by the incoming transit traffic without caching, $\rho_1^I|_{\kappa_1=0}$. The values for the outgoing transit traffic savings are calculated similarly, i.e., $(\rho_1^O|_{\kappa_1=0} - \rho_1^O)/\rho_1^O|_{\kappa_1=0}$.

### *D. Model Validation*

In order to validate the model for RNS, we use the same scenarios as for the unrestricted neighbor selection (cf. Sect. IV-C). The change of the number of leechers is shown in Figs. 5(a) and 5(b). Fig. 5(a) is analogue to Fig. 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 $\kappa_1 \in [0.2, 0.4]$ 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 $\kappa_1$ 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.

Fig. 5(b) corresponds to Fig. 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 Sect. IV-C3. 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 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-domain traffic, we consider the normalized transit traffic savings in analogy to Sect. V-C. In Fig. 6(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




(a) Normalized number of peers. (b) Incoming transit traffic savings $\rho_1^I|_{\kappa_1=0} - \rho_1^I$. (c) Outgoing transit traffic savings $\rho_1^O|_{\kappa_1=0} - \rho_1^O$.

Fig. 7. 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 $\kappa_1$ in case of restricted neighbor selection.

seeds never upload to remote leechers. The corresponding savings in outgoing traffic (labeled "Outgoing (strict rns)" in Fig. 6(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. Fig. 6(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.

### *E. Numerical Results and Insights*

We first consider the system dynamics with RNS. In Fig. 7(a), the number of leechers and seeds for RNS is shown as a function of the cache upload capacity $\kappa_1$. 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 $\kappa_1$ 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 $\kappa_1$ is required to reach the download rate limited state due to the RNS policy.

Fig. 7(b) shows the incoming transit traffic savings of ISP 1 for RNS obtained from the model. Comparing the figure to Fig. 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 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 $\kappa_1$. The outgoing transit traffic savings of ISP 1 obtained from the model are shown in Fig. 7(c) for the case with RNS. Again, we present non-normalized transit traffic savings to show the asymptotic limits. Comparing the figure to 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*.

## VII. Conclusion

We considered the impact of caches on the inter-ISP traffic due to BitTorrent-like peer-to-peer systems. We developed a simple fluid model of the effects of caches on the system dynamics and showed using the model how the caches installed in an ISP affect the system-wide and the local peer-dynamics. We described a simple model of inter-ISP traffic and used 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 provided asymptotic bounds on the efficiency of caches, and gave a comparison of the efficiency of caches under our modeling assumptions. We showed that caches can sometimes lead to increased outgoing transit traffic, depending on the portion of the peers within the ISP. We described a restricted neighbor selection policy, extended the fluid model to capture its effect on the system dynamics, and showed 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 validated 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. It will be subject of our future work to extend the analytical model of transit traffic to more complex network scenarios.

## REFERENCES


[1] H. Schulze and K. Mochalski, "Internet Study 2008/2009," 2009. [Online]. Available: http://www.ipoque.com/resources/internet-studies/

[2] "Cisco visual networking index: Forecast and methodology, 2009–2014," White paper, Jun. 2010. [Online]. Available: http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-481360.pdf

[3] 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 *Proc. IEEE Int'l Conf. Distributed Computing Systems (ICDCS)*, Jul. 2006, p. 66.

[4] D. R. Choffnes and F. E. Bustamante, "Taming the torrent: a practical approach to reducing cross-ISP traffic in peer-to-peer systems," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 4, pp. 363–374, Oct. 2008.

[5] V. Aggarwal, A. Feldmann, and C. Scheideler, "Can ISPs and P2P systems co-operate for improved performance?" *SIGCOMM Comput. Commun. Rev.*, vol. 37, no. 3, pp. 29–40, Jul. 2007.

[6] H. Xie, Y. R. Yang, A. Krishnamurthy, Y. G. Liu, and A. Silberschatz, "P4P: Provider portal for applications," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 4, pp. 351–362, Oct. 2008.

[7] PeerApp UltraBand, "http://www.peerapp.com."

[8] OverCache P2P, "http://www.oversi.com."

[9] M. Hefeeda and O. Saleh, "Traffic modeling and proportional partial caching for peer-to-peer systems," *IEEE/ACM Trans. Netw.*, vol. 16, no. 6, pp. 1447–1460, Dec. 2008.

[10] N. Leibowitz, A. Bergman, R. Ben-Shaul, and A. Shavit, "Are file swapping networks cacheable? Characterizing P2P traffic," in *Proc. Int'l Workshop on Web Content Caching and Distribution (WCW)*, Aug. 2002.

[11] T. Karagiannis, P. Rodriguez, and K. Papagiannaki, "Should Internet service providers fear peer-assisted content distribution?" in *Proc. ACM Internet Measurement Conf. (IMC)*, Oct. 2005, pp. 63–76.

[12] A. Wierzbicki, N. Leibowitz, M. Ripeanu, and R. Woźniak, "Cache replacement policies for P2P file sharing protocols," *Euro. Trans. on Telecomms.*, vol. 15, pp. 559–569, Nov. 2004.

[13] X. Yang and G. de Veciana, "Service capacity of peer to peer networks," in *Proc. IEEE INFOCOM*, Mar. 2004, pp. 2242–2252.

[14] D. Qiu and R. Srikant, "Modeling and performance analysis of BitTorrent-like peer-to-peer networks," in *Proc. ACM SIGCOMM*, Aug. 2004, pp. 367–378.

[15] L. Massoulié and M. Vojnović, "Coupon replication systems," *ACM SIGMETRICS Perf. Eval. Rev.*, vol. 33, no. 1, pp. 2–13, Jun. 2005.

[16] 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, pp. 32–49, Oct. 2005.

[17] Y. Tian, D. Wu, and K. W. Ng, "Modeling, analysis and improvement for BitTorrent-like file sharing networks," in *Proc. IEEE INFOCOM*, Apr. 2006, pp. 1–11.

[18] I. Rimac, A. Elwalid, and S. Borst, "On server dimensioning for hybrid P2P content distribution networks," in *Proc. IEEE Int'l Conf. Peer-to-Peer Computing (P2P)*, Sep. 2008, pp. 321–330.

[19] B. Cohen, "Incentives build robustness in BitTorrent," in *Proc. Workshop on Economics of Peer-to-Peer Systems*, Jun. 2003.

[20] "BitTorrent specification." [Online]. Available: http://wiki.theory.org/BitTorrentSpecification

[21] "BitTorrent local tracker discovery protocol." [Online]. Available: http://bittorrent.org/beps/bep_0022.html

[22] "Application-layer traffic optimization (alto)." [Online]. Available: http://www.ietf.org/html.charters/alto-charter.html

[23] "Decoupled application data enroute (decade)." [Online]. Available: https://datatracker.ietf.org/wg/decade/charter/

[24] L. Guo, S. Chen, Z. Xiao, E. Tan, X. Ding, and X. Zhang, "Measurement, analysis, and modeling of BitTorrent-like systems," in *Proc. ACM Internet Measurement Conf. (IMC)*, Oct. 2005, pp. 35–48.

[25] D. P. Heyman and M. J. Sobel, *Stochastic Models in Operations Research, Volume I*. McGraw-Hill, New York, 1982.

[26] K. Leibnitz, T. Hoßfeld, N. Wakamiya, and M. Murata, "Peer-to-peer vs. client/server: Reliability and efficiency of a content distribution service," in *Proc. Int'l Teletraffic Congress (ITC)*, Jun. 2007.

[27] "Protopeer," http://protopeer.epfl.ch/index.html.

[28] "BitTorrent Library for Protopeer." [Online]. Available: http://protopeer.epfl.ch/wiki/BitTorrent

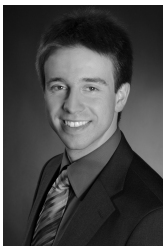
[29] T. Hoßfeld, F. Lehrieder, D. Hock, S. Oechsner, Z. Despotovic, W. Kellerer, and M. Michel, "Characterization of BitTorrent swarms and their distribution in the Internet," *Computer Networks*, vol. 55, no. 5, Apr. 2011.

[30] German-Lab project, "http://www.german-lab.de/."

[31] Planet-Lab, "http://www.planet-lab.org/."

[32] A. Rao, A. Legout, and W. Dabbous, "Can realistic BitTorrent experiments be performed on clusters?" in *Proc. IEEE Int'l Conf. Peer-to-Peer Computing (P2P)*, Aug. 2010.

[33] Team Cymru, "IP to ASN Mapping." [Online]. Available: http://www.team-cymru.org/Services/ip-to-asn.html

**Frank Lehrieder** studied computer science and business administration at the University of Würzburg/Germany and University Antonio de Nebrija in Madrid/Spain. In May 2008, he received his Master degree in computer science from University of Würzburg. Now, he is a researcher at the Chair of Communication Networks in Würzburg and pursuing his PhD. His special interests are performance evaluation and resource management for peer-to-peer networks and Future Internet infrastructures.

**György Dán** received the M.Sc. degree in computer engineering from the Budapest University of Technology and Economics, Hungary in 1999 and the M.Sc. degree in business administration from the Corvinus University of Budapest, Hungary in 2003. He worked as a consultant in the field of access networks, streaming media and videoconferencing 1999-2001. He received his Ph.D. in Telecommunications in 2006 from KTH, Royal Institute of Technology, Stockholm, Sweden, where he currently works as an assistant professor. He was a visiting researcher at the Swedish Institute of Computer Science in 2008. His research interests include the design and analysis of distributed and peer-to-peer systems.

**Tobias Hoßfeld** studied computer science and mathematics at the University of Würzburg, Germany. He finished his PhD on "Performance Evaluation of Future Internet Applications and Emerging User Behavior" in 2009. Currently, he is heading the research group "Future Internet Applications and Overlays" at the Chair of Communication Networks in Würzburg. His main research interests cover network virtualization, social networks and Internet dynamics, self-organization and traffic management mechanisms in overlay networks and P2P systems, as well as investigations on Quality of Experience (QoE) for Internet applications like Skype or YouTube.

**Simon Oechsner** studied computer science and received his Master degree from the University of Würzburg, Germany in 2004. He received his Ph.D. in 2010 from the same university, and is now a visiting professor at Universitat Pompeu Fabra (UPF), Barcelona, Spain. His research focuses on analytical and simulative performance evaluation of overlays. His interests include search and content distribution overlays, covering DHTs as well as currently popular filesharing and video-streaming networks.

**Vlad Singeorzan** studies computer science at the University of Wuerzburg/Germany. Currently, he is working on the optimization of BitTorrent-based peer-to-peer networks and pursuing his Master degree at the Chair of Communication Networks.