Impacts of Improved Peer Selection on Internet Traffic in BitTorrent Networks

Peter Danielis, Jan Skodzik, Dirk Timmermann
University of Rostock
Institute of Applied Microelectronics and Computer Engineering
18051 Rostock, Germany
Tel./Fax: +49 (381) 498-7272 / -1187251
Email: {peter.danielis;jan.skodzik;dirk.timmermann}@uni-rostock.de
Web: http://www.imd.uni-rostock.de/networking

Thomas Bahls, Daniel Duchow
Nokia Siemens Networks GmbH & Co. KG
Broadband Access Division
17489 Greifswald, Germany
Tel./Fax: +49 (89) 5159-22771 / -18237
Email: {thomas.bahls;daniel.duchow}@nsn.com

***Abstract*—Peer-to-Peer (P2P) file sharing generates by far the most Internet traffic reaching up to 70 % in some regions of the world. These data volumes pose a significant challenge to Internet Service Providers (ISPs) regarding traffic engineering. Because P2P routing is usually agnostic of the underlying topology, traffic engineering abilities of ISPs are inhibited and their core networks are overburdened with P2P data. To disburden ISPs' core networks, we propose a new algorithm for the BitTorrent (BT) protocol in order to improve peer selection. BT users are provided with accurate information on the hop counts to other BT users to select physically proximate users. Thereby, the initial Time-To-Live value (TTL) of outgoing IP packets is copied and inserted as part of the BT payload. At the packet's destination, the hop count is calculated as the difference between the copied TTL and the TTL of the IP header. We present simulation results for standard and modified BT implementation and discuss impacts on both the load of ISPs' core networks and BT users' download performance.**

***Keywords*-BitTorrent, Peer-to-Peer, Topology Awareness**

I. Introduction

Today, Internet traffic is dominated by Peer-to-Peer (P2P) data ranging from 43 % up to 70 % in different regions of the world. This is mainly caused by file sharing applications such as eMule or BitTorrent (BT). Particularly, BT traffic accounts for up to 80 % of P2P traffic, i.e., 56 % of overall Internet traffic [1].

On the one hand, Internet Service Providers (ISPs) benefit from the P2P hype through an increase of their operating income. This is because P2P applications are one of the main reasons for Internet users to subscribe for a broadband connection [2]. But on the other hand, the high P2P data volumes pose a significant traffic engineering challenge. This discrepancy between operating income and traffic engineering challenge puts network operators and ISPs into a difficult situation. Other traffic like HTTP is choked down because the network infrastructure is overburdened with P2P data. The main reason is that routing within logical P2P networks does not take the underlying physical Internet topology into account [3]. Usually, an unstructured P2P network overlay—on which we focus here—is constructed by choosing random peers [4]. Due to this arbitrary procedure, neighborship on the P2P overlay does *not* implicate proximity on the underlying Internet topology at all. This problem is usually denoted as topology mismatching problem between P2P overlays and physical network infrastructures [5]. Thus, two communicating P2P neighbors may be physically far away from each other although the desired content is often available on a physically more proximate peer as well [6]. Communication with physically distant peers uses long data paths, e.g., regarding the hop count. This consumes more bandwidth, which is highly inefficient when the load of the network is already high. It can therefore cause traffic congestions [7]. In contrast, communication with proximate peers consumes less bandwidth and traffic in the core network diminishes. ISPs benefit from traffic reduction in their core networks as more bandwidth is available for other applications. Moreover, traffic congestions can be reduced as well.

Consequently, our contribution was motivated by the idea of disburdening ISPs' core networks by reducing the physical path lengths, i.e., the hop count. We developed a new algorithm for the BT protocol to use hop count as additional selection criterion for peers (besides their download performance). However, the hop count for determining proximity is not part of packets. Therefore, we also propose a new approach to provide P2P users—*BT users in particular*—with information on physical hop counts to other BT users. We modified the standard BT implementation such that the initial Time-To-Live value (TTL) of outgoing IP packets is inserted as part of the BT payload. At the packet's destination, the modified BT algorithm calculates the hop count from the inserted initial TTL and the received IP packet's TTL.

However, a BT user primarily wants to download desired content as fast as possible. He is usually not aware of or even not interested in the underlying transport mechanism. Thus, a user would not select the most proximate BT user among all users, which provide the desired content with nearly the same upload capability. However, we assume BT users to be cooperative by selecting close-by users unless they do

Copyright (c) IARIA, 2011. ISBN: 978-1-61208-125-0

not suffer from this decision regarding their performance. We define performance as the time required to get desired content, which we call the users' Quality of Experience (QoE). In the best case, it is beneficial for users and ISPs. In the worst case, we want the performance to be equal to or not considerably lower than the standard case when not using the hop count.

Briefly summarized, the main contributions of this paper are the following:

- Investigations are carried out on how to calculate the hop count and provide it for BT users.
- Improved peer selection for BT is described, where hop count information is used by a modified BT algorithm to select close-by users.
- Simulation results for standard and modified BT algorithm are presented and (dis-)advantages for both ISPs and BT users are discussed.

The remainder of this paper is organized as follows: Section II contains a comparison of the proposed approach with related work. Section III explains the computation of the hop count from the TTL and how to provide the initial TTL. Section IV addresses improved peer selection for BT. Section V shows impacts of improved peer selection on the load of an ISP's core network and BT users' performance. The paper concludes in Section VI.

II. Comparison with Related Work

Many approaches, e.g., [8] and [9] to *construct* unstructured topology-aware overlay networks do exist. These approaches improve performance significantly and avoid unnecessary traffic by exploiting network proximity. However, they require adding structure to unstructured P2P networks following physical network characteristics. Furthermore, traffic overhead is created for maintaining this structure. In contrast to these approaches, we do not intervene with the *construction* of unstructured P2P networks. Instead, we slightly modify the BT protocol to provide the hop count and select proximate peers by means of a new BT choking algorithm. Thereby, no modification of the construction algorithm is necessary and no additional packets have to be sent to determine the distance, i.e., the hop count between peers.

Also, there are approaches to *shape* P2P traffic in a more efficient way *with* network support. The IETF has planned to develop a protocol for Application-Layer Traffic Optimization (ALTO). By means of an ALTO server, peers can obtain information "to perform better-than-random initial peer selection" [10]. The Portal for P2P Applications (P4P) project aims at allowing more effective cooperate traffic control between P2P applications and ISPs via dedicated trackers to localize P2P traffic [11]. Moreover, in the course of the Network-Aware P2P-TV Application over Wise Networks (NAPA-WINE) project, the impacts on the underlying transport network shall be minimized when distributing P2P-IPTV datastreams [12]. Thereby, a so-called "network-peer can perform actions to optimize the P2P overlay taking into account the status of the transport network". [13] suggests to use autonomous system (AS) hop count to achieve locality-awareness in BitTorrent-like P2P applications. However, the tracker is required to know the Internet topology and peers have to obtain dynamic distance information by P4P or content distribution networks. As opposed to these approaches, our approach is completely self-sufficient and does not require network support but does the necessary modifications solely in the BT application.

III. Computing Hop Count for BT

Since hop count information is neither stored in the IP header nor elsewhere, it is necessary to compute it. There are two methods for hop count calculation [14]. One is the so-called active measurement. The other is denoted as passive measurement. For active measurement, ICMP ECHO packets are used. Although this method mostly results in an accurate hop count, applying it to many hosts in current and prospective P2P scenarios is impractical since enormous traffic overhead is created. Contrary, passive measurement simply means subtracting the final TTL of a received IP packet from its initial TTL. This is ideal for computing hop counts of many hosts as no extra packets have to be sent. Consequently, this approach is chosen for calculating the hop count. However, the problem with passive measurement is that the initial TTL is not available in IP packets. Thus, it must be made available.

The TTL is part of the IPv4 (further referred to as IP) header [15]. It is used as hop counter. Thus, each router processing a packet decrements the TTL by one. To calculate the hop count from TTL, the initial TTL of an outgoing IP packet is required. Then, this value can be subtracted from the final TTL of the IP header at the packet's destination to get the hop count. As shown in [16], due to the heterogeneity of the Internet, there is *no* unique initial TTL. The initial TTL depends on the operating system.

Consequently, the question is: How to provide the initial TTL? Therefore, we propose a modified BT algorithm to insert the initial TTL into BT messages.

A. Standard (tracker-based) BT algorithm

For every shared file in BT, an own P2P net is created. To search for a file, usually a web site is contacted to get a *.torrent* file. This file contains, among other things, the address of a *tracker* and information about the file to be downloaded. The tracker is contacted to get a list of BT users holding the file (or parts of it—so-called *chunks*). Thereby, all BT users, which are interested in this file, form a so-called *swarm*. Complete downloader serving the whole files are called *seeds*. Incomplete downloaders are called *leechers*.

Copyright (c) IARIA, 2011. ISBN: 978-1-61208-125-0

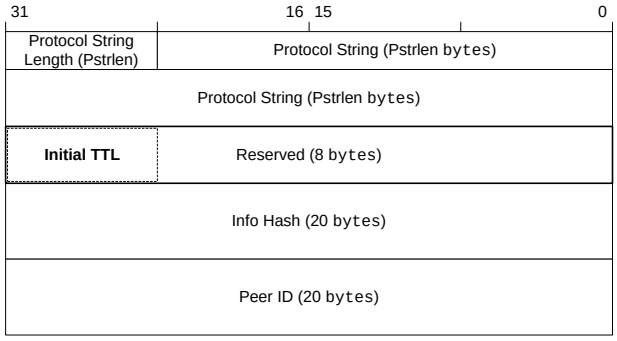

Figure 1. Composition of a BT Handshake message.

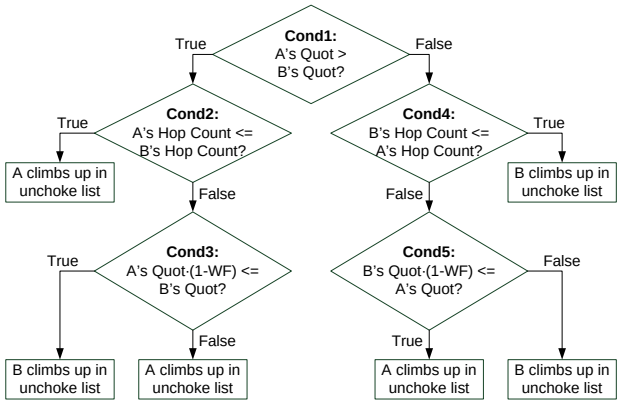

Figure 2. New BT choking algorithm for calculating a user's position in the unchoke list. The hop count is used as additional selection criterion.

For selecting a user who may download a chunk, BT applies the so-called *choking algorithm* [17]. Put in a nutshell, this is a variant of the tit-for-tat strategy. Only users offering sufficient upload performance are given download in return. The choking algorithm to determine a user that may download chunks is executed periodically because upload performance of users can change quickly.

B. Including the Initial TTL in BT messages

To introduce as few overhead as possible into the BT algorithm and BT traffic, the initial TTL is only included into necessary BT messages. Two types of messages can be distinguished in BT flows: the tracker requests and responses and the messages between BT users. Thereby, messages between BT users are solely exchanged via TCP sockets. One message necessary for BT user interaction is the *Handshake* message (see Figure 1) [18]. Per BT specification 1.0, which is widely used for BT protocol implementations, Pstrlen is set to 19 and Protocol String = "BitTorrent protocol".

A BT Handshake is sent by the initiator of a connection between two users of a swarm. In return, the recipient of the Handshake message has to respond with a Handshake message himself. In both the Handshake message and the response to it, the modified BT algorithm directly inserts the initial TTL. The authors suggest to use the first byte of the eight *Reserved* bytes since these bytes can be used to change the BT protocol behavior.

C. Providing the TTL in the BT application

As TCP sockets are used for communication, IP header fields like the TTL are not available in applications per se. Therefore, following two questions are answered to clarify how to make it available:

1) How to get the initial TTL of outgoing BT Handshakes? The initial TTL is retrieved via the BSD sockets compatible function *getsockopt*, which is both Windows and Unix-compatible. Thereby, the specific socket option *IP_TTL* is used, which is supported by TCP sockets for outgoing packets.

2) How to get the final TTL of incoming BT Handshakes? Since TCP sockets do not offer support for providing the TTL of incoming packets, the pcap (packet capture) interface is used *additionally*. Unix-like OS apply the pcap implementation *libpcap*, whereby Windows OS use a port of libpcap called *WinPcap*. Using filters like the Berkeley Packet Filter, already the kernel can be instructed to copy only packets, which match the composition of a BT Handshake, to the BT application. Thus, the kernel buffer is not overfilled with packets, which could lead to high packet loss. Possible alternatives are *raw sockets* offering direct access to the network layer as well. However, they do either forward each packet to the application or do not forward TCP packets at all (depending on the OS implementation) and are thus not feasible.

IV. Improved Peer Selection for BitTorrent

The modification of BT for improved peer selection concerns BT's choking algorithm. The standard choking algorithm selects BT users that may download chunks solely depending on their offered upload performance (except *optimistic unchokes*). Users are ranked in an unchoke list such that the user with the highest upload performance (i.e., the highest *service rate*) is on top. Usually, a fixed number of users on top of the list (e.g., 4) may download concurrently. In the modified version, users are no longer ranked only depending on their upload performance. Instead, for two users A and B *in another user's unchoke list*, quotients for those users A and B are calculated as

$$Quotient = Service\ Rate / Hop\ Count.$$

The quotient (denoted as *Quot* in the algorithm depicted in Figure 2) determines a user's rank in the unchoke list. If A's quotient is greater than B's, i.e., condition 1 = true (denoted as *Cond1* in Figure 2) and A's hop count is smaller than or equal to B's (Cond2 = True), A climbs up in the unchoke list.

However, if A's hop count is greater than B's (Cond2 = False), A's quotient is multiplied (i.e., weighted) by a

variable factor (*1 - hop count weighting factor* (WF)) with WF values ranging from 0 to 1. If A's weighted quotient is smaller than or equal to B's quotient (Cond3 = True), B climbs up in the unchoke list because B's hop count is weighted more than A's upload performance. Otherwise (Cond3 = False), A climbs up because A's upload performance is weighted more than B's hop count. In the else-branch of the algorithm, we have stated the analog conditions for B's quotient being greater than A's.

As an exemplification for the algorithm, let us assume a user A with high service rate and high hop count and a user B with moderate service rate but very low hop count. Following the calculation rule for a user's quotient, A is assigned a relatively low quotient compared to B's quotient regardless A's high service rate. Still, A's quotient be greater than B's quotient in this example (Cond1 = True) although B's hop count be significantly smaller than A's hop count (Cond2 = False). However, B's hop count can be given an even higher weight by assigning an approriate, i.e., high WF value to let B climb up in the unchoke list (Cond3 = True).

To summarize, WF is used to make a compromise regarding the weighting of a user's upload performance and his hop count. A high WF value results in BT users with low hop counts on top of the unchoke list almost regardless their upload performance. Vice versa, a low WF value leads to a higher weight of a user's upload performance. Thus, users with high upload performance are put more probably on top of the unchoke list nearly irrespective of their hop count.

One approach for the selection of close-by BT users is to let the user decide directly. The alternative approach, which we follow in this paper, is to integrate an automatic selection mechanism into the BT algorithm.

V. Evaluation of Standard and Modified BT Algorithm

In this section, simulation results are shown using the network simulator ns-2. Results include a comparison of standard with modified BT algorithm in terms of the number of hops between users, the load of the core network, and users' QoE.

A. Simulation Setup

To implement the BT algorithm in ns-2, a BT patch from [19] has been used. The BT algorithm has been complemented by the dynamic nature of peers, i.e., the BT users' behavior of continuously leaving and entering the BT network. According to [20], BT users follow a Weibull distribution when arriving at the BT network (inter-arrival time). Session lengths of BT users, i.e., the time how long they stay in the BT network each time they appear are implemented to follow a Weibull distribution as well. When BT users leave the BT network after a session, they return after a uniformly distributed time (downtime). Finally, BT users stay in the BT network for a while after the completion of a download (lingering). The lingering time is modeled with a Weibull distribution.

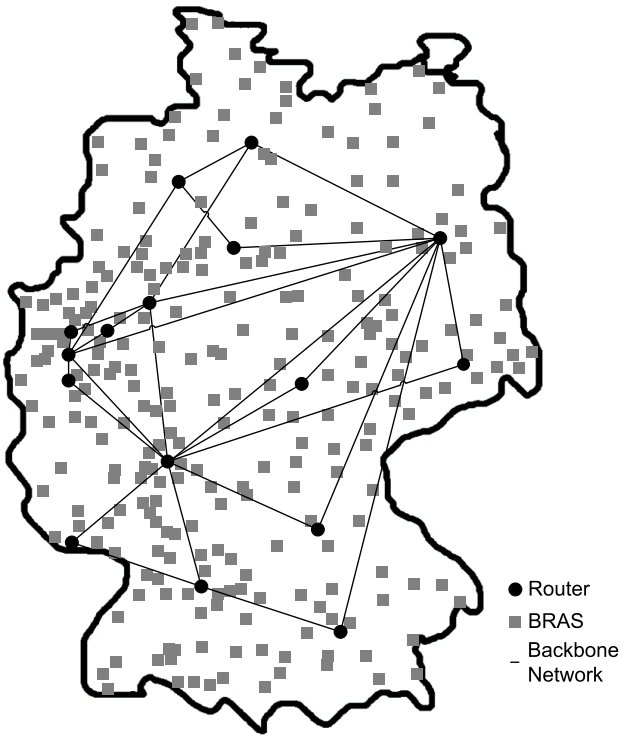

Figure 3. Telefonica's network infrastructure in Germany [21]

At the start of each simulation run, one BT user is a seeder. This seeder stays in the network until each BT user of the swarm has finished his download of a file of 100 MB in size. The maximum number of users that may download concurrently from another user is set to 4.

For the simulation, a topology has been developed, which maps Telefonica's backbone network in Germany [21]. Telefonica possesses one of the most capacious network infrastructures in Germany. The schematic layout of the developed topology is depicted in Figure 3 and consists of

- a backbone network of routers,
- Broadband Remote Access Servers (BRAS) that are connected to routers, which are linked to DSL Access Multiplexers (DSLAMs),
- and BT users, which are connected to DSLAMs.

In accordance with Telefonica's network infrastructure in Germany, the developed topology comprises 16 routers. The number of BRAS is set to 4 per router (resulting in 64 BRAS) and there are 6 DSLAMs per BRAS (resulting in 384 DSLAMs). The number of BT users is set to 200 for the simulations. The routers form a static structure like the one apparent in Figure 3. BRAS are uniformly distributed around routers and in the same way, DSLAMs are uniformly distributed around BRAS. Users are randomly connected to

Copyright (c) IARIA, 2011. ISBN: 978-1-61208-125-0

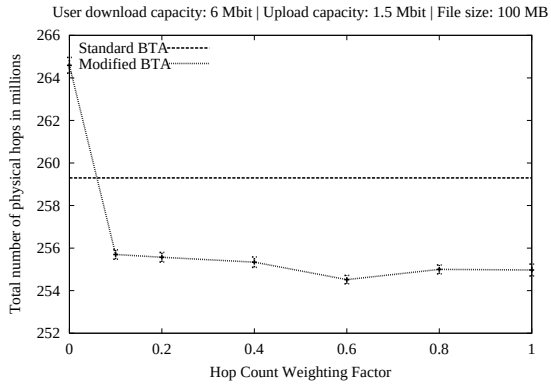

Figure 4. Number of hops for varying WF values. The result for standard BT is independent from WF values and therefore constant.

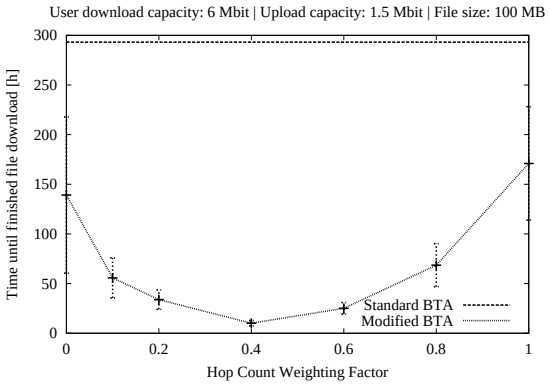

Figure 5. Time until the last BT user has downloaded the file for varying WF values. The result for standard BT is independent from WF values and therefore constant.

DSLAMs. The number of BRAS, DSLAMs, and users is fixed for all simulations. The bandwidth between routers and between routers and BRAS has been set to 20 Gbit/s. The bandwidth between DSLAMs and BRAS is set to 1 Gbit/s. These bandwidths values are common values in practice. Each BT user is assigned a download capacity of 6 Mbit and an upload capacity of 1.5 Mbit, which are reasonable values for an asymmetric Internet access.

B. Simulation Results

Both standard and modified BT algorithm (BTA) have been simulated on the developed topology. In our simulations, the following values have been determined for varying WF values to compare both algorithms:

- Number of physical hops: Summed up number of physical hops that data has to travel through the network during the simulation.
- Data volume in the core network: Summed up data volume passing the routers of the core network during the simulation.
- Time necessary for the last user to finish a file download: Time needed until the last BT user has finished the file download during the simulation.

As users are connected to DSLAMs randomly, for each WF value on the x-axis, 38 measurements have been taken. In the diagrams, the mean value of those 38 measurements is depicted on the y-axis for each WF value. For the modified BT, the 95 % confidence interval (CI) is depicted to demonstrate that the measurements' precision is sufficient to draw conclusions.

The calculated mean value for standard BT is independent from WF values and thus constant. Therefore, the CI is not charted for standard BT (CI for the data volume in the core network: 73 Mbit, CI for the number of physical hops: 234344, CI for the time necessary for the last user to finish a file download: 157 h).

As apparent from Figure 4, the number of hops decreases when applying the modified BTA except for WF = 0. For WF = 0, the simulation results show a minor increase of the number of hops by 2 % compared to the standard BTA. This is due to the fact that hop count is considered by the algorithm in Figure 2 but users' upload performance dominates as peer selection criterion. Thereby, the degrees of freedom are limited and the number of hops increases. Please remember, that hop count is *always* considered by the modified BTA and an increasing WF value solely *boosts* the hop count's influence. Any other WF value decreases the total number of hops. In fact, for WF = 40, 60, 80, and 100 %, we achieved the highest reduction of approximately 2 %.

This relatively slight decrease in the number of hops results in a significant lower load of the core network for the modified BT variant. Table I illustrates this fact, showing a reduction of the core load by 11 % for WF = 40 and 60. The slight increase of the core load for WF = 0 % has the same reasons that apply for the increased number of hops for WF = 0.

Furthermore, our simulations show that the time necessary until the last BT user has downloaded the complete file is considerably lower if the modified BTA is applied (see Figure 5). In fact, time is even decreased by up to 97 % for WF = 40 %. This tremendous decrease of time involves a reduction of the core load by 11 % and a decrease of the number of hops by 2 %. Thus, choosing WF values between 40 and 60 % is obviously benefical for both BT users and the core network as these values offer the highest degrees of freedom regarding peer selection.

Copyright (c) IARIA, 2011. ISBN: 978-1-61208-125-0

Table I
DATA VOLUME IN THE CORE NETWORK FOR VARYING WF VALUES. THE RESULT FOR STANDARD BT IS INDEPENDENT FROM WF VALUES AND THEREFORE CONSTANT.

WF	Standard BTA	Modified BTA		
	Data volume [Gbit]	Data volume [Gbit]	Data volume reduction [%]	CI [Mbit]
0	29.10	29.48	-1	79
0.1		26.24	10	82
0.2		26.20	10	84
0.4		26.03	11	75
0.6		25.99	11	68
0.8		26.16	10	80
1.0		26.19	10	90

VI. CONCLUSION

This paper proposes a new choking algorithm for the BT protocol to preferably select physically close-by BT users in order to disburden ISPs' core networks. The selection criterion is the hop count. It is calculated from the difference of the initial TTL value of a packet's IP header and the TTL value at the packet's destination. As the initial TTL is not directly available, it is inserted into BT Handshake messages by the modified BT algorithm.

The simulations carried out for the BT algorithm clearly show that ISPs benefit from a modified BT using the hop count as additional selection criterion for download partners. The load of the ISP's core network is alleviated by up to 11 %. Thereby, traffic is localized (the number of hops is reduced by up to 2 %). Moreover, our simulation show that users' QoE tremendously increases as time for the last BT user to finish a download is decreased by up to 97 %.

Future work will focus on providing the hop count for further P2P file sharing protocols such as eMule's unstructured eDonkey2000 and its impacts on Internet traffic.

ACKNOWLEDGEMENT

The authors would like to thank the Broadband Access Division of Nokia Siemens Networks GmbH & Co. KG in Greifswald, Germany for their inspiration and continued support in this project. This work is partly granted by Nokia Siemens Networks.

REFERENCES

[1] H. Schulze, K. Mochalski (ipoque), "Internet Study 2008/2009," 2009.

[2] T. Mennecke, "DSL Broadband Providers Perform Balancing Act," 2005.

[3] V. Aggarwal, S. Bender, A. Feldmann, and A. Wichmann, "Methodology for Estimating Network Distances of Gnutella Neighbors." GI Jahrestagung (2), 2004, pp. 219–223.

[4] R. Steinmetz and K. Wehrle, *P2P Systems and Applications, Springer Lecture Notes in Computer Science*. Springer-Verlag Berlin Heidelberg, 2005.

[5] H. Wan, N. Ishikawa, and J. Hjelm, "Autonomous Topology Optimization for Unstructured Peer-to-Peer Networks." ICPADS, 2005, pp. 488–494.

[6] A. Rasti, D. Stutzbach, and R. Rejaie, "On the Long-term Evolution of the Two-Tier Gnutella Overlay," no. 4146697. INFOCOM, 2006.

[7] X. Xiao and L. Ni, "Internet QoS: A Big Picture," vol. 13. IEEE Network Magazine, 1999, pp. 8–18.

[8] S. Merugu and E. Zegura, "Adding Structure to Unstructured Peer-to-Peer Networks: The Use of Small-World Graphs." JPDC, 2005, pp. 142–153.

[9] Y. Liu, X. Liu, L. Xiao, L.M.Ni, and X. Zhang, "Location-Aware Topology Matching in P2P Systems." INFOCOM, 2004, pp. 2220–2230.

[10] IETF, "Application-Layer Traffic Optimization (alto)," 2009. [Online]. Available: http://datatracker.ietf.org/wg/alto/charter/

[11] H. Xie, Y. R. Yang, A. Krishnamurthy, and Y. L. A. Silberschatz, "P4P: Provider Portal for Applications." ACM SIGCOMM, 2008, pp. 351–362.

[12] E. Leonardi, M. Mellia, A. Horvath, L. Muscariello, S. Niccolini, and D. Rossi, "Building a cooperative P2P-TV application over a wise network: The approach of the European FP-7 strep NAPA-WINE." IEEE Communications Magazine 46 (4), 2008, pp. 20+22.

[13] B. Liu, Y. Cao, Y. Cui, Y. Lu, and Y. Xue, "Locality Analysis of BitTorrent-Like Peer-to-Peer Systems." 7th IEEE CCNC, 2010, pp. 1–5.

[14] K. Fujii and S. Goto, "Correlation between Hop Count and Packet Transfer Time." APAN/IWS, 2000.

[15] Information Sciences Institute, University of Southern California, "Internet Protocol Specification," RFC 791, September 1981.

[16] Swiss Education & Research Network (SWITCH), "Default TTL Values in TCP/IP," 2002.

[17] B. Cohen, "Incentives Build Robustness in BitTorrent." First Workshop on the Economics of Peer-to-Peer Systems, June 2003.

[18] " Bittorrent Protocol Specification v1.0," 2009. [Online]. Available: http://wiki.theory.org/BitTorrentSpecification

[19] K. Eger, T. Hofeld, A. Binzenhofer, and G. Kunzmann, "Efficient Simulation of Large-Scale P2P Networks: Packet-level vs. Flow-level Simulations." UPGRADE-CN'07, 2007, pp. 9–16.

[20] D. Stutzbach and R. Rejaie, "Understanding CHurn in Peer-to-Peer Networks." ACM SIGCOMM Internet Measurement Conference, 2006, pp. 189–202.

[21] Telefonica, "Unser Netz," 2009. [Online]. Available: http://www.telefonica.de/wholesale/unser-netz.html

Copyright (c) IARIA, 2011. ISBN: 978-1-61208-125-0