Using Proximity Information between BitTorrent Peers: An Extensive Study of Effects on Internet Traffic Distribution

Peter Danielis, Jan Skodzik, Jens Rohrbeck, Vlado Altmann, 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@uni-rostock.de

Thomas Bahls
Ernst-Moritz-Arndt-University of Greifswald
Institute for Community Medicine
17487 Greifswald, Germany
Tel: +49 (3834) 86-7524
Email: Thomas.Bahls@uni-greifswald.de

Daniel Duchow
Nokia Siemens Networks GmbH & Co. KG
Broadband Access Division
17489 Greifswald, Germany
Tel: +49 (89) 5159-18237
Email: daniel.duchow@nsn.com

***Abstract*—Peer-to-Peer 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 regarding traffic engineering. Because Peer-to-Peer routing is usually agnostic of the underlying topology, traffic engineering abilities of Internet Service Providers are inhibited and their core networks are overburdened with Peer-to-Peer data. To disburden Internet Service Providers' core networks, a new algorithm for the BitTorrent protocol is proposed in order to improve peer selection. BitTorrent users are provided with accurate information on the hop counts to other BitTorrent users to select physically proximate users. Thereby, the initial Time-To-Live value of outgoing IP packets is copied and inserted as part of the BitTorrent payload. At the packet's destination, the hop count is calculated as the difference between the copied Time-To-Live value and the Time-To-Live value of the IP header. Simulation results for standard and modified BitTorrent implementation are presented highlighting beneficial effects on both the traffic of Internet Service Providers' core networks and BitTorrent users' download performance. The extensive exploration of impacts on Internet traffic includes the observation of traffic loads and peaks in the core networks as well as the number of dropped packets due to congestions. Moreover, as realistic simulations require the consideration of the BitTorrent users' behavior of continuously leaving and entering the BitTorrent network, this article elaborates on the dynamic nature of peers.**

Keywords-Peer-to-Peer; Topology Awareness; BitTorrent; Dynamic User Behavior; Traffic Analysis.

I. Introduction

During the last years, several new flavors of private and commercial use of the Internet have developed. One of these flavors are Peer-to-Peer (P2P) networks, clients, tools, and communities [1]. Today, Internet traffic is dominated by 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 [2]. However, there are numerous other application areas for P2P, e.g., P2P-based IPTV like Joost or Zattoo [3], [4].

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 [5]. But, on the other hand, the high P2P data volumes pose a significant traffic engineering challenge. Traffic engineering denotes the process of managing traffic flows through the network [6]. 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 [7]. Usually, an unstructured P2P network overlay—on which this article focuses—is constructed by choosing random peers [8]. 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 [9]. 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 [10]. 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 [6]. Congestions are neither good for ISPs nor for users. In contrast, communication with proximate peers consumes less bandwidth and traffic in the core network diminishes. ISPs benefit from a traffic reduction in their core networks as more bandwidth is available for other applications. Moreover, traffic congestions can be reduced as well.

Consequently, this contribution was motivated by the idea of disburdening ISPs' core networks by reducing the physical path lengths, i.e., the hop count. A new algorithm for the BT protocol to use hop count as additional selection criterion for peers (besides their download performance) has been developed. However, the hop count for determining proximity is not part of packets. Therefore, a new approach to provide P2P users—*BT users in particular*—with information on physical hop counts to other BT users is proposed as well. The standard BT implementation is modified 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 speed. However, BT users are assumed to be cooperative by selecting close-by users unless they do not suffer from this decision regarding their performance. Performance is defined as the time required to get desired content and is denoted as the users' Quality of Experience (QoE) in the following. In the best case, it is beneficial for users and ISPs. In the worst case, the performance shall be equal to or not considerably lower than the standard case when not using the hop count.

Briefly summarized, the main contributions of this article 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.
- The integration of the BT protocol in the network simulator ns-2 is detailed as well as the incorporation of the dynamic nature of BT peers.
- Simulation results for standard and modified BT algorithm are presented and (dis-)advantages for both ISPs and users are discussed.

The remainder of this article 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 motivates the use of the network simulator ns-2, elaborates on the dynamic nature of BT peers, and shows impacts of improved peer selection on the traffic in an ISP's core network and BT users' performance. The article concludes in Section VI.

II. Comparison with Related Work

The mismatch between logical P2P overlay and the underlying physical topology becomes a serious obstacle for the development of P2P systems. Being aware of the mismatching problem, much scientific literature can be found, which addresses the locality problem in Distributed Hash Table (DHT)-based P2P networks, i.e., structured P2P networks [11], [12]. In structured P2P networks, all peers are organized into an identifier ring and an association between content and location where it is stored is given. Basically, there are three approaches, which have been proposed to exploit network proximity in DHTs. For a detailed discussion of the three approaches, the interested reader is referred to [13].

However, not only structured but also unstructured P2P networks suffer from the mismatching between the logical overlay and the underlying physical topology. Hence, this article focuses on selecting close-by peers in unstructured P2P networks. In contrast to structured P2P networks, in unstructured P2P networks like BT, peers are organized arbitrarily. Thus, neighborship on the P2P overlay does *not* implicate physical proximity on the underlying Internet topology at all.

Many approaches, e.g., [14] and [15], 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, the authors' approach does not intervene with the *construction* of unstructured P2P networks. Instead, the BT protocol is slightly modified 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" [16]. 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 [17]. 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 [18]. Thereby, a so-called "network-peer can perform actions to optimize the P2P overlay taking into account the status of the transport network". Liu et al.

[19] suggests to use autonomous system (AS) hop count to achieve locality-awareness in BT-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, the approach of this article 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 determination [20]. Either it can be actively measured or it can be calculated by using the TTL. 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. Thus, as measuring method in the envisaged P2P use case, it is not favorable to use active measurement. Contrary, the calculation of hop count simply means subtracting the final TTL of a received IP packet from its initial TTL. This is optimal 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 regarding hop count calculation 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 [21]. 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 [22], due to the heterogeneity of the Internet, there is *no* unique initial TTL. The initial TTL depends on the operating system (OS).

Consequently, the question is: How to provide the initial TTL? Therefore, a modified BT algorithm is proposed, which inserts the initial TTL into BT messages.

A. Standard (tracker-based) BT algorithm

BT is currently the most widespread P2P network for file sharing. It accounts for about 56 % percent of the overall Internet traffic [2]. Its popularity is mostly due to its high download rates—the main interest of users. BT focuses on high speed rather than on search capabilities.

For each 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 *pieces*, which are further subdivided into sub-pieces). Thereby, all BT users, which are interested in this file, form a so-called *swarm*. Complete downloaders serving the whole files are called *seeds*. Incomplete downloaders are called *leechers*.

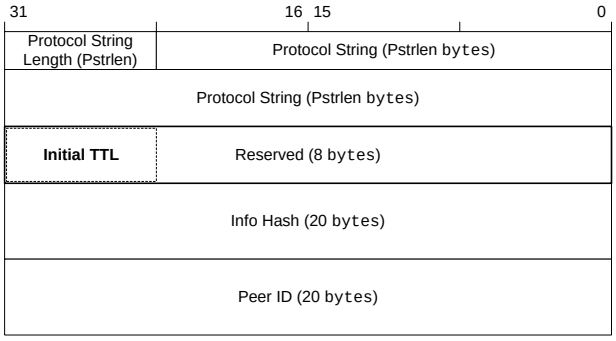

Figure 1. Composition of a BT Handshake message.

BT users start to download pieces in *random* order and change to *rarest first* order after the first piece is completed [23]. Thereby, they follow the *strict priority* of solely requesting sub-pieces of a particular piece before sub-pieces from the next piece. Furthermore, for a quick finish of a download, BT applies the so-called *endgame mode*, which is enabled when a download is almost complete. Then, a BT user simultaneously requests all missing sub-pieces from all BT users he knows.

For selecting a user who may download a piece, BT applies the so-called *choking algorithm* [23]. 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 (they are *unchoked*). The choking algorithm to determine a user that may download pieces is executed periodically because upload performance of users can change quickly. As an exception, each BT user has an *optimistic unchoke* available to unchoke one other user regardless of his upload performance. Furthermore, BT applies an *anti-snubbing* policy as part of the choking algorithm. Thereby, BT users, which have not given a single piece within one minute, are not unchoked except for optimistic unchokes.

Once a BT user has finished his download, he may decide to stay in the network for a while (lingering). During this time span, he only uploads pieces preferring users, to which he has the best upload rates.

B. Including the initial TTL in BT messages

To introduce as little 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) [24].

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) library is used *additionally*. Unix-like OS's apply the pcap implementation *libpcap*, whereby Windows OS's use a port of libpcap called *WinPcap*. libpcap provides a high degree of portability as it supports numerous OS's [25]. 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, raw sockets have turned out to be buggy, unportable, and may not be able to capture TCP packets at all (depending on the OS) 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 pieces 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 service rate. 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

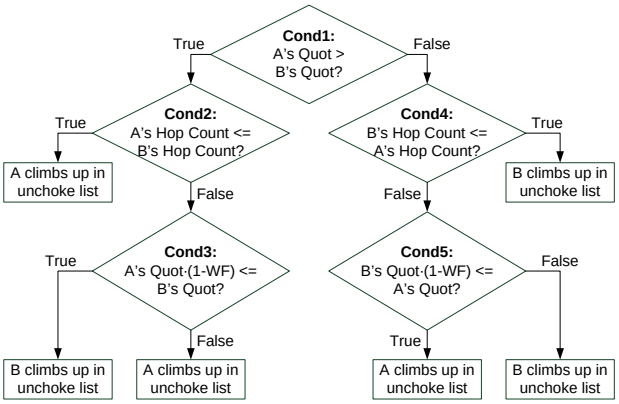

Figure 2. New BT choking algorithm for calculating a user's position in the unchoke list. The hop count is used as additional selection criterion.

(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 service rate. Otherwise (Cond3 = False), A climbs up because A's service rate is weighted more than B's hop count. In the else-branch of the algorithm, the analog conditions for B's quotient being greater than A's are stated.

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 of 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 appropriate, 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 service rate and his hop count. A high WF value results in BT users with low hop counts on top of the unchoke list almost regardless of their service rate. Vice versa, a low WF value leads to a higher weight of a user's service rate. Thus, users with high service rate 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 followed in this article is to integrate an automatic selection mechanism into the BT algorithm.

V. Evaluation of Standard and Modified BT Algorithm

In this section, the choice of the network simulator ns-2 for the evaluation of the authors' proposed approach is motivated. Then, the integration of the BT protocol into ns-2 is described including an elaboration of the dynamic nature of BT users. Finally, 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, the number of dropped packets, the maximum throughput in the whole network, and users' QoE.

A. Determination of an appropriate simulator

It is essential to choose an appropriate simulator to achieve realistic and trustful simulation results. There is a wide range of network simulators, especially in the field of P2P simulations. Table I lists the most significant open-source simulators, which could be used for the evaluation of standard and modified BT algorithm and compares them regarding scalability, license, and OS. Most of the simulators run under Linux/Unix or are implemented in Java and are thus platform independent. Node numbers within the range of 10^4 up to 10^6 are supported and hence, simulations of large P2P networks are possible.

Table I
Simulators and their properties [26], [27], [28], [29], [30].

Simulator	Scalability	License	OS
ns-2	$200 - 10^4$	GNU GPL	Linux/Unix
OverSim	10^5	GNU GPL	Linux, Mac, Windows
PlanetSim	10^5	GNU LGPL	Platform independent
PeerfactSim.KOM	$10^5 - 10^6$	GNU GPL	Platform independent
PeerSim	10^6	GNU LGPL	Platform independent

In the following, the choice of ns-2 is motivated by giving an overview of its advantages regarding the consideration of the physical underlay.

1) Advantages of ns-2: Most simulators do not accurately take into account the physical underlay, e.g., to speed up simulations [31].

OverSim supports three different types of underlays, which are the simple, single host, and INET underlay [27]. The INET framework, for example, offers the opportunity of modeling access and backbone networks based on a complete IP protocol stack. However, the accuracy of the simulator is not known and thus, realistic conditions cannot be assumed without further ado. Planetsim is an overlay network simulation framework, which solely provides simple networks like RingNetwork or CircularNetwork that do not consider latency costs [28]. Therefore, it does not provide sufficiently realistic network conditions. The same applies to PeerfactSim.KOM, which uses mathematical and stochastic models to emulate the behavior of a network and does not take into account the physical underlay [29]. PeerSim contains, among other things, an event-based engine, which supports transport layer simulations [30]. Thereby, the transport layer is represented by a special protocol that provides a message sending service and therefore, realistic conditions cannot be assumed without further ado as well.

On the contrary, ns-2 is a complex simulator attempting to model the whole network stack using real network components like network nodes and routers to construct the network [32]. Therefore, an explicit correlation between the logical P2P overlay and physical underlay is given. Thus, it is possible to examine the impact of improved peer selection on the underlay, which is necessary for the evaluation of the authors' proposed approach.

Furthermore, ns-2 contains full implementations of standard protocols like TCP, UDP, and FTP. These protocols are verified and run through continuous validations [33]. Therefore, these protocols are available for the implementation of the BT protocol to the full extent. This is an essential aspect as BT uses TCP for data transmission through the network.

Last but not least, ns-2 is well documented, which alleviates the development and evaluation of the modified BT algorithm [26].

In the authors' opinion, ns-2 is best suited for the envisaged simulations due to the given reasons and has therefore been chosen.

2) ns-2's principle of operation: There are two possibilities to set up and influence simulation runs in ns-2.

First of all, complex networks with hundreds to thousands of nodes and their connections can be defined in a TCL script. Thereby, the physical underlay (network topology) for the BT network is created. Following basic parameters for the simulation can be adjusted:

- Number of nodes
- Topology of network
- Bandwidth of the connection between nodes
- File size to be distributed

Furthermore, trace files can be declared, which record the whole network traffic during the simulation.

Table II
Recorded data in a trace file [26]

Event	Description
+	Packet has been prepared and put into the waiting queue to be sent
-	Packet has been transmitted to destination
h	Packet traveled a hop by passing a node
d	Packet has been dropped

The second and most powerful possibility is to modify the protocols to be simulated directly in their C implementation. Additionally, new protocols can be created and

basic elements like nodes, waiting queues, or links may be redefined. The BT protocol modifications concerns BT's choking algorithm as described in Section IV. When the TCL script starts running, trace files are generated and record the network communication between nodes. Recorded data consists of events (important events are listed in Table II) and serves as the basis of the statistics creation.

An example for traced data is shown in Table III. Thereby, T denotes simulated time in seconds. SOURCE gives the source node from where the data packet is sent and SINK specifies the destination node of the packet. TYPE provides information on the packet type; SIZE states the packet size in bytes.

Table III
EXAMPLE DATA IN A TRACE FILE [26]

EVENT	T [s]	SOURCE	SINK	TYPE	SIZE [Byte]
+	0.000233	12	6	tcp	40
-	0.000234	12	6	tcp	40
h	0.000312	12	10	tcp	40
d	0.000333	3	7	tcp	40

Traced data is finally processed by Perl scripts and sorting algorithms to create desired statistics. The processing has been automated to be able to handle the large amounts of simulation data (several tens of GB per simulation run) efficiently.

B. BitTorrent in ns-2

A BT patch from [34] has been used to implement the BT algorithm in ns-2. This implementation contains almost the full functionality of the BT protocol (see Section III-A) as follows:

- Partitioning of a file into pieces
- Strict priority download strategy
- Random and rarest first download strategy
- Choking algorithm
- Optimistic unchokes
- Upload only: Leave options of peers, which have completed their file download

The endgame mode and the anti-snubbing strategy have not been implemented. However, as the focus is on data transmission efficiency, this simplification is tolerable.

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 [35], BT users follow a Weibull distribution when arriving at the BT network (inter-arrival time) (see Figure 3). A Weibull distribution follows the cumulative distribution function $Weibull(x, \lambda, k) = 1 - e^{-(x/\lambda)^k}$ with scale parameter λ and shape parameter k. For better comparability with [35], the complementary cumulative distribution function (CCDF) $1 - Weibull(x, \lambda, k)$ of the respective distribution is used.

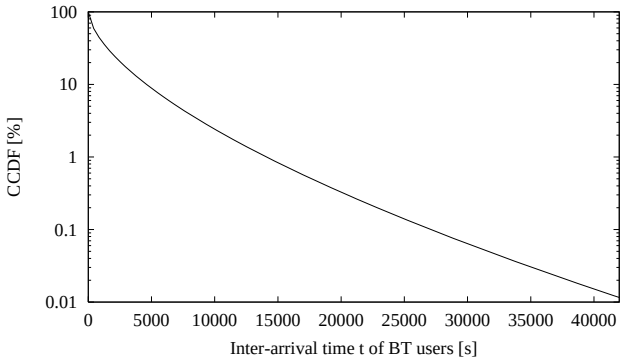

Figure 3. Complementary cumulative Weibull(t, 1200, 0.62) distribution function for the inter-arrival time of BT users. For the y-axis, a base 10 logarithmic scale is used.

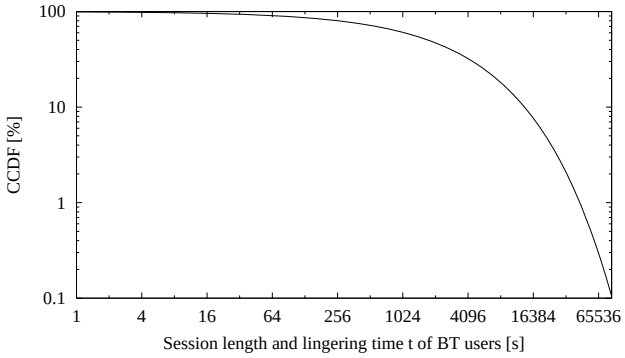

Figure 4. Complementary cumulative Weibull(t, 3302.977, 0.59) distribution function for the session length and lingering time of BT users. For the x-axis, a base 2 logarithmic scale is used; for the y-axis, a base 10 logarithmic scale is used.

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 (see Figure 4). When BT users leave the BT network after a session, they return after a uniformly distributed time (downtime). For clarity reasons, the uniform distribution is not depicted. Finally, BT users stay in the BT network for a while after the completion of a download (lingering time). The lingering time is modeled with the same Weibull distribution like the session lengths (see Figure 4).

Table IV
ASPECTS OF THE DYNAMIC NATURE OF BT USERS [35]

BT USER BEHAVIOR	DISTRIBUTION	λ	K	T_{max}
Inter-arrival time	Weibull	1200	0.62	700 min
Session length	Weibull	3302.977	0.59	-
Downtime	Uniform	-	-	0.5 d
Lingering time	Weibull	3302.977	0.59	-

Parameter values of the distribution functions are listed in Table IV. Thereby, scale parameter λ and shape parameter k have been set to values, which are reasonable for the

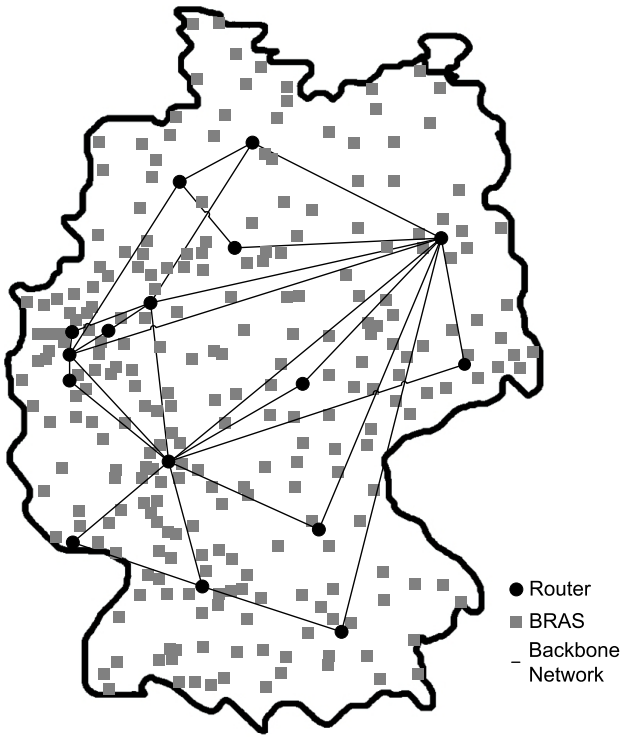

Figure 5. Telefonica's network infrastructure in Germany [36]

simulation of downloading a file of 100 MB in size by 200 users. Moreover, T_{max} has been introduced for the distribution functions of inter-arrival time and downtime to properly restrict their codomains for the simulation.

C. Simulation setup

For the simulation, a topology has been developed, which maps Telefonica's backbone network in Germany [36]. Telefonica possesses one of the most capacious network infrastructures in Germany. The schematic layout of the developed topology is depicted in Figure 5 and consists of

- a backbone (core) network of routers,
- Broadband Remote Access Servers (BRAS's) 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's is set to 4 per router (resulting in 64 BRAS's) 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 5. BRAS's are uniformly distributed around routers and in the same way, DSLAMs are uniformly distributed around BRAS's. Users are randomly connected to DSLAMs. The number of BRAS's, DSLAMs, and users is fixed for all simulations. The bandwidth between routers and between routers and BRAS's has been set to 20 Gbit/s. The bandwidth between DSLAMs and BRAS's 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.

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.

D. Simulation results

Both standard and modified BT algorithm (BTA) have been simulated on the developed topology. In the 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.
- Number of packet drops: Absolute number of packet drops due to congestions.
- Maximum throughput in the whole network: Traffic peak in the whole network and its point in time.
- 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, 10 measurements have been taken. In the diagrams, the mean value of those 10 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 in the given figures:

- CI for the number of physical hops: 950842
- CI for the data volume in the core network: 214 Mbit
- CI for the number of packet drops: 35785
- CI for the maximum throughput in the whole network: 2 Mbit/s and for its point in time: 29 s
- CI for the time necessary for the last user to finish a file download: 46.5 min

Number of physical hops: As apparent from Figure 6, the number of hops decreases when applying the modified BTA. For WF = 0, the simulation results show a minor increase of the number of hops by 0.02 % compared to the standard BTA. This is due to the fact that hop count is considered by the algorithm in Figure 2 but the users' service rate

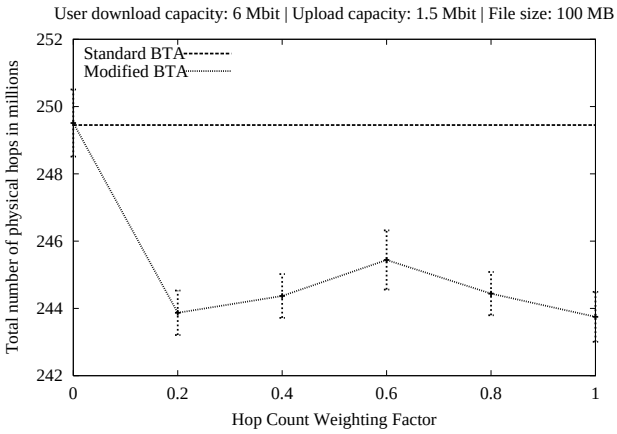

Figure 6. Number of hops for varying WF values. The result for standard BT is independent from WF values and therefore constant.

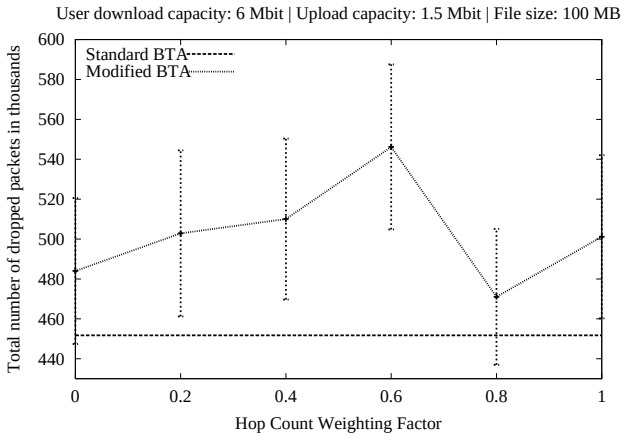

Figure 7. Number of dropped packets for varying WF values. The result for standard BT is independent from WF values and therefore constant.

Table V
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	14.43	14.22	1.5	240
0.2		12.13	15.9	175
0.4		12.29	14.8	168
0.6		12.23	15.2	179
0.8		13.01	9.8	291
1.0		11.98	17.0	142

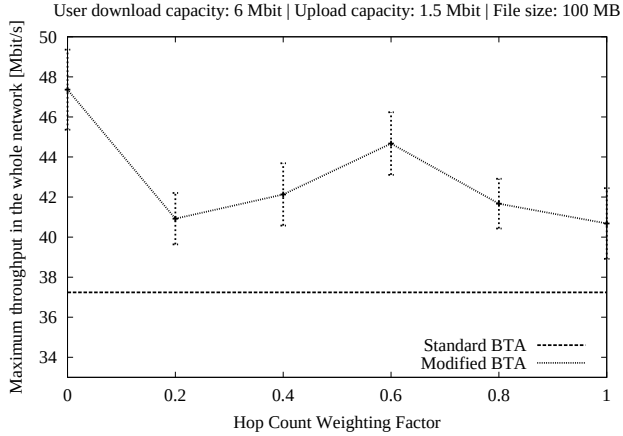

Figure 8. Maximum throughput in the whole network for varying WF values. The result for standard BT is independent from WF values and therefore constant.

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. For each other WF value except WF = 0, the total number of hops decreases. In fact, for WF = 1, the highest reduction of 2.3 % has been achieved.

Data volume in the core network: 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 V illustrates this fact, showing a reduction of the core load by 17 % for WF = 1.

Number of packet drops: In Figure 7, the number of dropped packets is shown. The number of dropped packets increases in case of the modified BT algorithm with an increase ranging from 7 to 21 %. However, the load in the core network is lower as traffic is pushed away from the core network to the network's edges. Thereby, packets may be dropped at BRAS's due to high data volumes exchanged by close-by users.

Maximum throughput in the whole network: In Figure 8, the maximum throughput in the whole network is depicted. It is considerably higher in case of the modified BT algorithm with an increase ranging from 9 % for WF = 1 to 27 % for WF = 0. Thereby, the users' upload service rate dominates as peer selection criterion for WF = 0 allowing for the highest increase in terms of maximum throughput still benefiting from traffic localization compared to the standard BTA. Thus, the modified BTA enables an increased maximum throughput allowing for a higher utilization of available bandwidth and a better exploitation of the network infrastructure's capabilities, respectively.

Thereby, the point in time of the traffic peak is between minute 15 and 17 after simulation start and shows only minimal differences (between 2 and 7 %) between both algorithms (see Figure 9). At this point in time, the majority of BT users (approximately 60 %) has entered the network as the inter-arrival time has been modeled by a Weibull(t, 1200, 0.62) distribution function (see Figure 3 and Table IV).

Time necessary for the last user to finish a file download: Furthermore, the simulations show that the time

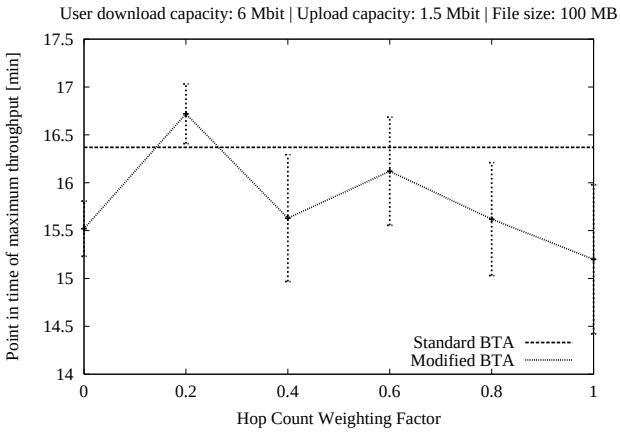

Figure 9. Point in time of maximum throughput for varying WF values. The result for standard BT is independent from WF values and therefore constant.

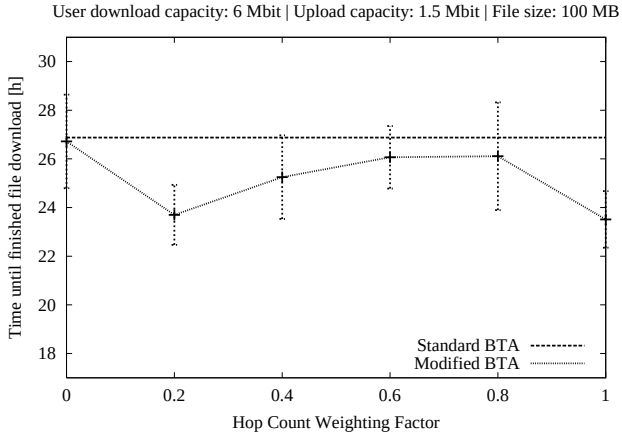

Figure 10. Time until the last BT user has downloaded the file for varying WF values. The result for standard BT is independent from WF values and therefore constant.

necessary until the last BT user has downloaded the complete file is considerably lower if the modified BTA is applied (see Figure 10). In fact, time is even decreased by up to 13 % for WF = 1. This decrease of time involves a reduction of the core load by 17 % and a decrease of the number of hops by 2.3 %. Thus, the most favorable WF value regarding BT users' QoE and the core network's load is obviously WF = 1, which weights the hop count highest.

VI. Conclusion

This article 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. Thereby, the modified BT protocol does neither require a modification of the construction algorithm for the BT network nor does it require sending additional packets to determine the hop count.

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 17 %. Thereby, traffic is localized (the number of hops is reduced by up to 2.3 %). By pushing traffic away from the core network to the network's edges, packets may be dropped there. Due to high data volumes exchanged by close-by users, this results in an increase of dropped packets ranging from 7 to 21 %. However, this is tolerable as the focus is on a decreased load in the core network to free bandwidth for traffic caused by other applications. The modified BTA enables an increased maximum throughput in the whole network of 9 to 27 % allowing for a higher utilization of available bandwidth. Moreover, the simulation show that users' QoE increases as time for the last BT user to finish a download is decreased by up to 13 %.

The utilization of hop count for the consideration of physical proximity is a generic approach. Future work will therefore focus on providing the hop count for further P2P file sharing protocols such as eMule's unstructured eDonkey2000 network. Currently, the mechanism has been implemented for IPv4 but IPv6 will be the dominating protocol in the prospective Internet. Thus, future work will be concerned with the adaptation of the new mechanism to IPv6 environments.

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] P. Danielis, J. Skodzik, D. Timmermann, T. Bahls, and D. Duchow, "Impacts of Improved Peer Selection on Internet Traffic in BitTorrent Networks," in *International Conference on Internet Monitoring and Protection (ICIMP 2011)*, 2011, pp. 8–13.

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

[3] "Joost Free Online TV," 2007. [Online]. Available: http://www.joost.com/

[4] "Zattoo TV to Go," 2007. [Online]. Available: http://zattoo.com/

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

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

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

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

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

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

[11] B. Y. Zhao, Y. Duan, L. Huang, A. D. Joseph, and J. D. Kubiatowicz, "Brocade: Landmark Routing on Overlay Networks," in *Proceedings of 1st International Workshop on Peer-to-Peer Systems (IPTPS)*, 2002, pp. 34–44.

[12] N. J. A. Harvey, M. B. Jones, S. Saroiu, M. Theimer, and A. Wolman, "SkipNet: A Scalable Overlay Network with Practical Locality Properties," in *Proceedings of the 4th conference on USENIX Symposium on Internet Technologies and Systems (USITS)*, 2003.

[13] M. Castro, P. Druschel, Y. Hu, and A. Rowstron, "Exploiting Network Proximity in Distributed Hash Tables," in *International Workshop on Future Directions in Distributed Computing (FuDiCo)*, 2002, pp. 52–55.

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

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

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

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

[18] 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.

[19] 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.

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

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

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

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

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

[25] T. Al-Harbash, "Raw IP Networking FAQ," 1999. [Online]. Available: http://www.ntua.gr/rin/rawfaq.html

[26] K. Fall and K. Varadhan, "The ns manual (the vint project)," 2008. [Online]. Available: http://www.isi.edu/nsnam/ns/doc/ns_doc.pdf

[27] I. Baumgart, B. Heep, and S. Krause, "OverSim: A flexible overlay network simulation framework," in *Proceedings of 10th IEEE Global Internet Symposium (GI '07) in conjunction with IEEE INFOCOM 2007*, 2007, pp. 79–84.

[28] P. García, C. Pairot, R. Mondéjar, J. Pujol, H. Tejedor, and R. Rallo, "Planetsim: A new overlay network simulation framework," in *Proceedings of the 19th IEEE International Conference on Automated Software Engineering (ASE). Workshop on Software Engineering and Middleware (SEM)*, 2004, pp. 123–136.

[29] D. Stingl, C. Gro, J. Rueckert, L. Nobach, A. Kovacevic, and R. Steinmetz, "Peerfactsim.kom: A simulation framework for peer-to-peer systems," in *Proceedings of the 2011 International Conference on High Performance Computing & Simulation (HPCS 2011)*, 2011, pp. 577–584.

[30] A. Montresor and M. Jelasity, "PeerSim: A scalable P2P simulator," in *Proceedings of the 9th Int. Conference on Peer-to-Peer (P2P'09)*, 2009, pp. 99–100.

[31] M. Baker and R. Lakhoo, "Peer-to-Peer Simulators," ACET, University of Reading, Tech. Rep., 2007.

[32] "The network simulator - ns-2," 2012. [Online]. Available: http://www.isi.edu/nsnam/ns/

[33] "The network simulator ns-2: Validation tests," 2008. [Online]. Available: http://www.isi.edu/nsnam/ns/ns-tests.html

[34] 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.

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

[36] O2/Telefonica, "Unser Netz," 2011. [Online]. Available: http://www.o2online.de/business/unternehmen/hosting/rechenzentren/netz/