Unrevealing the structure of live BitTorrent Swarms: methodology and analysis

Michal Kryczka*†, Ruben Cuevas†, Carmen Guerrero† and Arturo Azcorra†
*Institute IMDEA Networks
†University Carlos III Madrid

***Abstract*—BitTorrent is one of the most popular application in the current Internet. However, we still have little knowledge about the topology of real BitTorrent swarms and how the traffic is actually exchanged among peers. This paper addresses fundamental questions regarding the topology of live BitTorrent swarms. For this purpose we have collected the evolution of the graph topology of 250 real torrents from its birth during a period of 15 days. Using this dataset we first demonstrate that real BitTorrent swarms are neither random graphs nor small world networks. Furthermore, we will see how some factors such as the torrent popularity affect the swarm topology. Secondly, the paper proposes a novel methodology in order to infer the clustered peers in real BitTorrent swarms, something that was not possible so far. Finally, we dedicate special effort to demonstrate that current BitTorrent swarms are experiencing a marked *locality* phenomenon at the overlay construction level (or connectivity graph). This locality effect is even more pronounced when we consider the exchange traffic relationships between peers. This suggests that an important portion of the BitTorrent traffic is currently confined within the ISPs. This opens a discussion regarding the relative gain of the locality solution proposed so far.**

I. Introduction

BitTorrent is one of the most used application in the current Internet and is responsible for an important portion of the upstream and downstream traffic as revealed by last Sandvine report from Fall 2010 [13]. This has attracted the attention of the research community that has thoroughly investigated the BitTorrent protocol and ecosystem. In spite of this big effort, we still have some gaps in the knowledge of some aspects of BitTorrent. For instance, we do not know much regarding the topological structure of real BitTorrent swarms. In this paper we address this issue. Some previous papers [1], [6], [14] have analysed the structure of BitTorrent swarms through simulation and controlled environments considering a reduced number of swarms. Although these studies help to obtain initial results regarding the composition of BitTorrent swarms, they cannot consider real effects (e.g. network congestion, ISPs policies, etc) that may severely affect the topology of live BitTorrent swarm. In this paper, instead, we rely on measurement techniques to infer the graph structure of real BitTorrent swarms. We collect data from 250 real swarms from its birth during a period of 15 days. Specifically, our measurement tool uses the Peer Exchange (PEX) extension of the BitTorrent protocol in order to collect the routing table (i.e. list of neighbours) of the peers. Using this technique we are able to obtain the routing table of 150k peers along the time. With this dataset we aim to answer the following critical but still unanswered questions: (i) *Does the connectivity graph of live BitTorrent correspond to a random graph?* Previous emulation studies have contradictory conclusions. Dale et al. [6] conclude that BitTorrent swarms are random graphs whereas Al-Hamra et al. [1] state that they are not. Our results demonstrate that BitTorrent swarms are neither random graphs nor small-worlds. (ii) *Does the popularity of the content (i.e. the size of the swarm) affect the swarm topology?* Al-Hamra et al. [1] is the only work addressing this question. The authors conclude that the size of the swarm does not impact the swarm topology in their controlled experiments. Rather, we will demonstrate that in live swarms the size has a clear impact on the graph structure. (iii) *How dynamic is the graph structure of the swarms?* To the best of our knowledge no previous study has addressed this question. Our results demonstrate that although the peers' routing table are continuously changing, this is, there is a high level of *microscopic* dynamicity, the *macroscopic* parameters, such as the clustering coefficient or characteristic path length, remain stable leading to a low level of *macroscopic* dynamicity.

In this paper we also study the existence of a *clustering* phenomenon in BitTorrent swarms in the *wild*. Again, we found in the literature contradictory results based on emulation and simulation experiments considering few torrents. On the one hand Legout et al. [10] report the existence of a clustering phenomenon based on their emulation experiments. The authors suggest that peers with similar speed cluster together. On the other hand, Dale et al. [6] conclude the absence of clustering phenomenon (this is coherent with their statement of BitTorrent swarms being random graphs). Contrary to the simulation and emulation experiments used by previous works, we look at the phenomenon of clustering using data collected from real swarms. First, we demonstrate that the clustering coefficient of BitTorrent swarms is typically higher than that of a random graph of the same size, thus there exists a clustering phenomenon so that we further analyse this issue. BitTorrent clients use a set of different algorithms such as the unchoke algorithm, the optimistic connect (used in leecher state) and the optimistic disconnect (used in seeder state) that are fundamental to understand the clustering phenomenon. Based on these algorithms a peer would only keep in its routing table those nodes with which it interacts (i.e. exchanges traffic) systematically. We name these nodes *stable neighbours*. Therefore, in BitTorrent the clustering phenomenon is tied to the exchange of traffic between peers. We have analysed the routing table (i.e. neighbourhood) of 50k peers from our dataset in its leecher and seeder state. We observe a high percentage and absolute number of stable peers in the routing table of the peer during the leecher state whereas this percentage and absolute number gets dramatically reduced during the seeder state. This suggests that a peer clusters with other peers during the leecher state, however, current implementation of BitTorrent clients leads to not clustering during the seeder state to avoid the leechers with high download capacity getting all seeders upload bandwidth.

On the other side, some studies [2], [9] have highlighted that the lack of underlay topology awareness of peer-to-peer applications (p2p) is unnecessarily pushing an important amount of traffic to the Internet Service Providers (ISPs) transit links, producing an increment of their operational costs. BitTorrent, due to the huge amount of data that generates, has received an special attention and several solutions to confine BitTorrent traffic within the ISP has been proposed [3], [15]. We refer to them as BitTorrent *Locality* solutions. Furthermore, some previous studies have analysed the expected performance of these *Locality* solutions [5], [12] assuming that current BitTorrent swarms responds to a random graph structure. However, to the best of the authors knowledge there is no work that has studied whether a locality-biased composition exists or not in the current BitTorrent swarms. Then, in this paper we first study if the connectivity graph of live BitTorrent swarms shows any biased composition towards ISP-based locality or country-based locality. Secondly, we focus on studying the locality effect at the cluster level. For this purpose, for each one of the 50k peers considered in the clustering analysis, we analyse whether their stable neighbourhood presents an ISP or country biased composition. Our results reveal that there is a clear trend towards ISP-locality at both connectivity and cluster level in current BitTorrent Swarms. Therefore, the ISPs' policies (e.g. throttling policies) [8], the spread of new implementation of BitTorrent clients that favour the creation of locality-biased ISPs [3], [15] and some network phenomenon (e.g. congestion) lead to have ISP locality biased BitTorrent swarms.

In short, the main contributions of this paper are:

- This paper presents the first large scale study based on real data (250 torrents) to understand the overlay structure of live BitTorrent swarms.
- Live BitTorrent swarms are neither random graphs nor small worlds. However, the size of the swarm has a clear impact on its topology structure: the larger the swarm is the more different its structure is from a random graph.
- The paper presents a complete study of the dynamicity of the BitTorrent live swarm topology along the time. Churn effect as well as the algorithm implemented by the current BitTorrent clients makes the peers to continuously change an important portion of their neighbourhood. Therefore, BitTorrent swarms show a high *microscopic dynamicity*. However, when we analyse the macroscopic topological metrics, i.e. clustering coefficient and characteristic path length, they remain stable, thus resulting in a low *macroscopic dynamicity*.
- The obtained results demonstrate that peers tend to cluster with other peers during the leecher state, however this tendency disappears when the peer becomes a seeder.
- The current BitTorrent swarms show a marked effect towards an ISP-biased composition. This is, peers include in their neighbourhoods a larger number of neighbours from its own ISP than expected in a random selection process. Furthermore, this locality effect appears also at the cluster level, since the set of stable neighbours are also biased towards being from the same ISP of the peer.

The rest of the paper is organized as follows. Section II describes our measurement infrastructure and methodology as well as the dataset used along the paper. Section III presents our fundamental findings regarding the overlay structure of live BitTorrent swarms. Afterwards, Section IV discusses the clustering phenomenon existing in current BitTorrent swarms. Section V focuses on analyzing the *locality*-biased composition of current BitTorrent swarms. Finally, Section VI concludes the paper.

II. Measurement Methodology

The aim of our measurement study is to retrieve the graph topology of BitTorrent swarms. For this purpose we collect the routing table (i.e. neighbourhood list) of each peer in the swarm by using the Peer Exchange (PEX) extension of BitTorrent protocol. In the rest of the section we provide a detailed description of both, the used measurement infrastructure and methodology.

A. Measurement Infrastructure

Our measurement infrastructure is formed by 3 physical machines including 4 virtual machines (VMs) each. In total we have 12 VMs, each one with a single public IP address. One of the VMs acts as *Master* whereas the other 11 are *Slaves*.

One the one hand, the *Master* is responsible for learning new torrents from a BitTorrent Portal and contacting the tracker that manages the swarm associated to each torrent. Furthermore, the *Master* coordinates to which IP addresses (i.e. peers) each slave has to connect at any moment.

On the other hand, each *Slave* has a list of IP addresses (i.e. peers). The *Slave* tries to connect to each one of these peers and to retrieve the routing table of the peer among other information.

B. Measurement Methodology

The methodology of our measurments is similar to one presented in [4]. In order to learn new torrents we decided to use The Pirate Bay portal. This is the most important BitTorrent portal according to Alexa rank[1] and some research studies [16]. The Pirate Bay offers an RSS service where each

[1] http://www.alexa.com/topsites

new torrent is announced as soon as it is uploaded to the portal. Our *Master* is subscribed to this RSS service so that it is able to learn from a new torrent just after its birth. This guarantees that we will be able to crawl the full lifespan of a given torrent. The RSS service provides the *Master* with the .torrent file that includes the IP address of the tracker managing the swarm associated to the torrent along with other information not relevant to this paper. The *Master*, then, periodically queries the tracker with the maximum frequency allowed by this one (around 10 to 15 minutes) to avoid being blacklisted by the tracker. Each answer of the tracker includes: the number of seeders (i.e. peers with a complete copy of the file), the number of leechers (i.e. peers with an incomplete copy of the file) and a random set (typically 200) of IP addresses of peers participating in the swarm. Furthermore, the *Master* is responsible for coordinating the *Slaves* activity. The *Master* learns the IP addresses of peers within a swarm from the tracker and also from the *Slaves* as we will see later. The *Master* has to schedule the connection of the different *Slaves* to a given peer: the *Slaves* contribute no chunks to the other peers, thus, if a *Slave* connects few consecutive times to a given peer, the latter blocks the former. In order to avoid this, the *Master* schedules the connection to each individual learnt peer in a round robin fashion so that a given *Slave* only connects to the same peer once every 11 connections (around 2 hours). This prevents that any peer blacklists our *Slaves*.

Each *Slave* receives a list of IP addresses (i.e. peers) to connect to. The *Slave* connects to each of them that is not behind a NAT, and for those peers supporting the Peer Exchange extension (PEX)[2] the *Slave* retrieves their routing table (the list of IP addresses to which given peer is connected). It is worth noting that the most important BitTorrent clients such as uTorrent and Vuze support PEX, thus we are able to retrieve the routing table for almost every reachable peer. Furthermore, each *Slave* informs the *Master* regarding the IP addresses obtained through PEX. If any of these IP addresses is new, the *Master* adds it to the list of IP addresses to crawl.

As mentioned before, there are peers that are behind a NAT and are not reachable, therefore if we fail to connect to a given IP address 5 times we declare this peer as *unreachable*. Furthermore, due to churn phenomenon some nodes join and leave the swarm dynamically, so a reachable node may become unreachable, thus after 5 times failing to connect to a previously reachable node we consider that it left the swarm.

C. Dataset description

We have applied the described measurement methodology to 250 consecutively published torrents, learnt by RSS from the Pirate Bay portal between 20th December 2010 until 4th January 2011. From this set of torrents we were able to learn the routing tables of 150k peers. It is worth to note that we collect the routing table evolution for a given peer since we periodically connect to it as explained above. Furthermore we map the peers' IP addresses to their country and ISP using the MaxMind Database [11].

III. BitTorrent Live Swarms Graph Topology

In this Section we analyse the main characteristic of the *Connectivity Graph* of real BitTorrent swarms and answer important questions such as *Are BitTorrent Swarms random graphs or small world topologies? Does the size of the swarm impact the overlay topology? What is the level of dynamicity on the graph structure of BitTorrent swarms?* To the best of the authors knowledge, this is the first study that analyse the structure of BitTorrent swarms in the wild.

A. Methodology

We represent a swarm as a collection of vertices (V) and edges (E). Each peer within the swarm is represented as a vertex, thus peer i is represented by v_i. Therefore $V = [v_1, v_2, ..., v_n]$, where n is the torrent population. Furthermore, $e_{ij} = 1$ if peer i and j are connected and 0 otherwise. Hence, the Connectivity Graph (or Matrix) is the representation of E in the form of a matrix. Note that in BitTorrent the connections are bidirectional, thus the connectivity matrix is symmetric.

For each torrent in our dataset we have collected the routing tables (i.e. neighbourhood) of each reachable peer[3]. We connect to each peer every 10 minutes, hence we are able to present a swarm's connectivity matrix evolution along the time in 10 minutes intervals. Then, we analyse both static and dynamic properties of the connectivity matrix.

For each snapshot of the connectivity matrix we calculate the following standard parameters used in graph theory studies:

- Clustering coefficient -*C*- (also called transitivity measurement) shows the probability that the adjacent vertices of a given vertex are connected among them.
- Characteristic path length -*L*- is the average length of the shortest path between each pair of vertices in the graph.
- Diameter -*D*- is the shortest path of maximum length between a pair of nodes in the graph.

Based on these parameters we perform an exhaustive characterisation of the overlay structure of current live BitTorrent swarms. It is worth noting that due to space constraints and since the results obtained from the analysis of *L* and *D* lead to the same conclusions, in this paper we only discuss *L*.

B. Are BitTorrent Swarms Small World or Random Graphs?

The first part of our analysis aims to understand what type of graph structure, small world or random graph, represents better a BitTorrent swarm. For this purpose we apply a simple graph theory analysis. For each torrent in our dataset we calculate the clustering coefficient (C_k) and characteristic path length (L_k) for the different snapshots along the torrent lifespan. Furthermore, we calculate the clustering coefficient (C_r) and characteristic path length (L_r) for a random graph with the same number of edges and vertices as the different snapshots. For each snapshot we also calculate $R_c = C_k/C_r$

[2]PEX is an extension to the BitTorrent protocols that allow the peers supporting it to exchange their routing tables. This reduces the load from the tracker since peers are able to learn other peers without asking the tracker.

[3]We can collect the routing tables of those peers that are not behind a NAT and implement the PEX protocol.

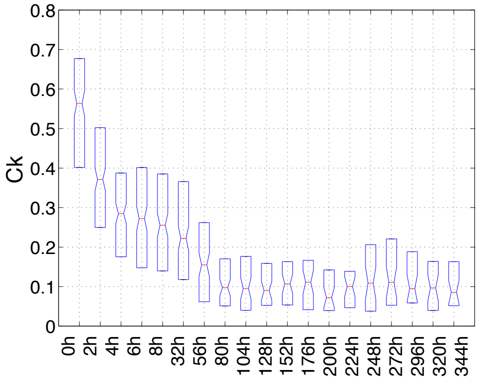


Fig. 1: Distribution of C_k after X hours from the torrent birth for the 250 torrents from our dataset

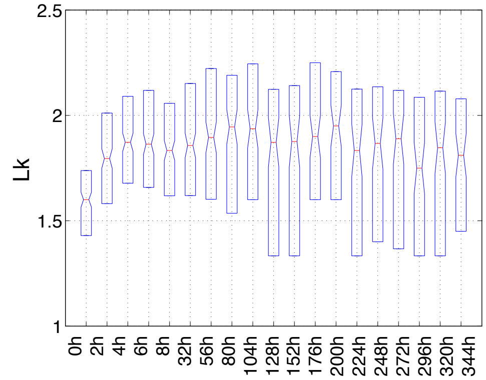


Fig. 2: Distribution of L_k after X hours from the torrent birth for the 250 torrents from our dataset

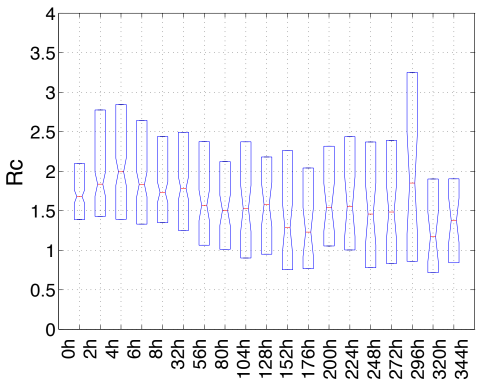


Fig. 3: Distribution of R_c after X hours from the torrent birth for the 250 torrents from our dataset

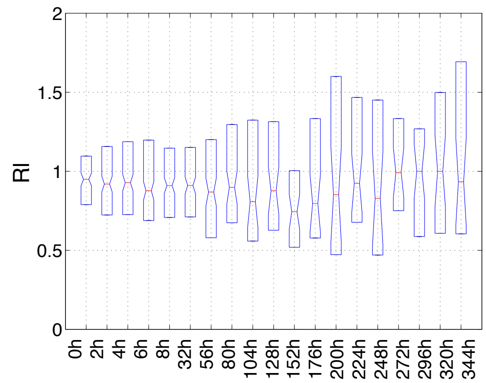


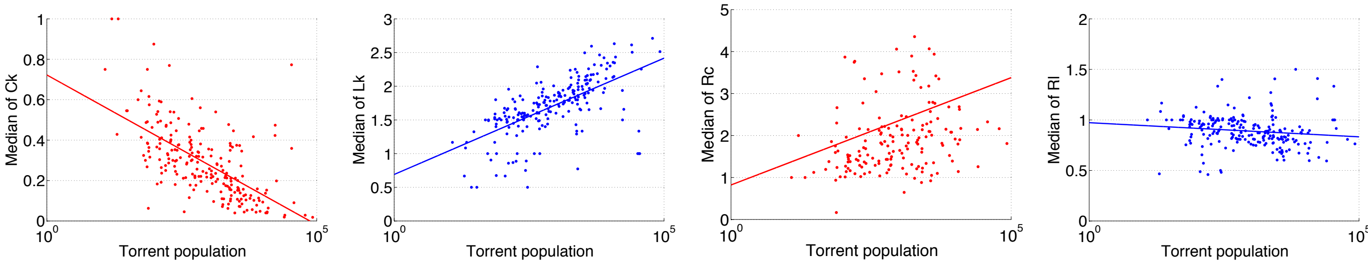
Fig. 4: Distribution of R_l after X hours from the torrent birth for the 250 torrents from our dataset

and $R_l = L_k/L_r$. These two ratios indicate how similar the actual swarm's graph is to either a random graph or a small world network. Those torrents having both R_c and R_l close to 1 are random graphs whereas those torrents having R_l close to 1 and $R_c >> 1$ have small world topology.

Figure 1 summarizes the evolution of C_k value for all the torrents in the dataset. In more detail, it shows C_k for the first 8 hours of the torrent lifespan with a 2 hours difference interval and with a step of 24 hours after this point. For each of these moments of the torrent lifespan we present a boxplot that indicates the 25%, 50% and 75% percentiles of the C_k considering all the torrents in the dataset. Figure 2 shows the same for L_k. The results help us understanding how the clustering coefficient and the characteristic path length evolve along the time in real BitTorrent swarms. On the one hand the clustering coefficient is higher in the birth phase of the torrent with values around 0.6 and continuously decrease until reaching stable state, at the 56-80 hours after the torrent birth phase, where the clustering coefficient is typically around 0.1 for most of the torrents. The explanation to this behaviour is the following: at the birth of the torrent the swarm size is small and most of the nodes are connected among them what leads to a high clustering coefficient. As the time passes, the torrent population grows, what leads to nodes being connected to just a portion of other nodes learnt from the tracker or through PEX. This produces a reduction on the clustering coefficient. The observed behaviour of the evolution of the clustering coefficient is consistent with previous emulation-based results [6], although the absolute values differ.

On the other hand, the characteristic path length is smaller in the birth of the torrent with a median value of 1.6 and experiences slightly grow to reach a stable phase after few (4 to 6) hours where the median of characteristic path length varies between 1.8 and 1.9. Again, in the birth phase we find a lower number of nodes that are well connected what leads to a lower characteristic path length. However, even in the stable phase the characteristic path length is small, what guarantees that pieces of the file can easily reach any part of the swarm in less than 3 hops for the vast majority of the torrents at any moment. This trend in the characteristic path length is consistent with previous emulation studies [6].

In order to understand if the actual BitTorrent swarms respond to a random graph topology we represent in Figure 3 and Figure 4 the evolution of the defined R_c and R_l ratios along the time for the torrents in our dataset. We use the same boxplot charts as for the clustering coefficient and characteristic path length evolution. On the one hand, we observe that $R_c > 1$ for most of the cases. This confirms that the clustering coefficient of the actual BitTorrent swarms is typically higher than the expected for random graph of the



(i) C_k *(ii)* L_k *(iii)* R_c *(iv)* R_l

Fig. 5: Median value of the different graph metrics (C_k,L_k,R_c,R_l) per torrent vs torrent size

same size. On the other hand the median for all torrents of $R_l < 1$ at any moment of the torrent lifespan. Therefore, in actual BitTorrent swarms we expect to have shorter paths between nodes than in random graphs of the same size. These observations are contradictory to previous studies [6], where the authors state that torrent swarms are random graphs. In a nutshell, the actual BitTorrent swarms show a slightly higher clustering coefficient and slightly lower characteristic path length than a random graph, so they are more clustered than an equivalent random graph. However, they neither fulfil the properties of small world networks.

C. Does the popularity of the content affects the swarm topology?

In this subsection we study how the popularity of the content, i.e. the size of the swarm, affects the overlay topology.

Figure 5 shows the median of clustering coefficient and the median of characteristic path length as a function of the overall popularity for the different torrents in our dataset. Moreover the figure presents trend lines calculated with polynomial curve fitting for each metric. We observe a clear trend making larger swarms having a lower clustering coefficient and a higher path length. This is because, as the torrent size grows the cluster effect gets reduced due to the high dynamicity of the connectivity graph reported in the next subsection.

However, the study of the absolute values of these metrics may lead to wrong conclusions such as the larger the swarm is the more random it is. Therefore we have also studied the dependency of R_c and R_l with the size of the torrent. These results are also shown in Figure 5 together with corresponding trend lines. We observe an increasing trend of R_c and a decreasing trend of R_l with the size of the swarm. Therefore, the trend is the opposite when considering relative metrics rather than absolute metrics. Finally, the reported trend of R_c and R_l suggests that *the larger the swarm is the less random typically it is*. This is an important observation that has not been reported before to the best of our knowledge.

D. What is the level of dynamicity on the graph topology of live BitTorrent swarm?

In this subsection we study how stable is the overlay topology of BitTorrent swarms. For this purpose we study the dynamicity of live BitTorrent swarms at two different levels of dynamicity: (i) *microscopic dynamicity*, we study how stable is the neighbourhood of a given peer along the time and (ii) *macroscopic dynamicity*, we study the evolution of the graph characteristics (clustering coefficient and characteristic path length) of a given torrent along the time.

a) Microscopic Dynamicity: As explained in Section II we have collected the routing table (i.e. neighbourhood) of each peer every 10 minutes approximately. In order to quantify the microscopic dynamicity we have calculated the percentage of neighbours that appear in two consecutive routing table snapshots for a given peer. We have perform a fine grain analysis and considered separately those periods in which the peer is a leecher and a seeder. The results presented in Figure 6 show that around half of the leechers change 50% of their neighbours every 10 minutes. This percentage dramatically increases up to 80% for seeders. This suggests that BitTorrent swarms suffer from an extremely high microscopic dynamicity that leads to a continuous change of the overlay graph. This high microscopic dynamicity is produced on the one hand by the churn effect (i.e. peers leaving and joining the swarm) and on the other hand by the combination of the different algorithm implemented in BitTorrent clients such as the unchoke algorithm, the optimistic connect algorithm (used in the leecher phase) and the optimistic disconnect algorithm (used in the seeder phase).

The unchoking algorithm makes a leecher to select N (typically 4) neighbours to upload chunks to every 10 seconds. These peers are then *unchoked* whereas the rest of the node's neighbours are *choked* and will not receive data from the peer. The BitTorrent node unchokes the N peers from whom it received more data in the last 20 seconds. Therefore, the unchoking algorithm tries to find *good* neighbours to exchange traffic with. Furthermore every 30 second the BitTorrent node performs an *optimistic unchoke*. This is, it chooses a random peer from its neighbourhood and uploads data to it. The optimistic unchoke allows to discover better peers to exchange traffic with.

Moreover, the most important BitTorrent clients such as Vuze or uTorrent utilizes the *optimistic connect* [7] during the leecher phase. This algorithm drops the connection to those neighbours that have uploaded few or no data to the leecher during some time. These neighbours are substituted by new ones. Therefore, the combination of the unchoking, optimistic unchoking and optimistic connect algorithms lead the leecher to identify and drop those peers from which the leecher is not

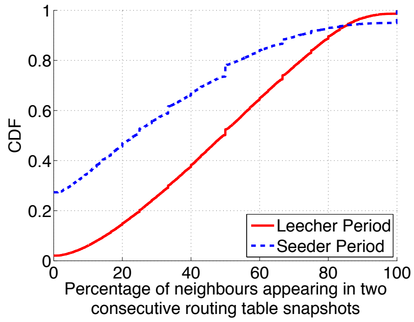


Fig. 6: CDF of the percentage of neighbours that appear in two consecutive snapshots (10 minutes apart) of a given peer

obtaining good enough performance, thus contributing to the high microscopic dynamicity reported before.

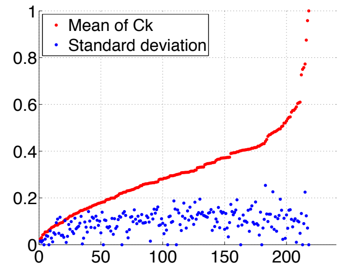
On the other hand, BitTorrent seeders apply different unchoke strategies depending on the implementation. The most extended strategies are: (i) *proportional*, the seeder unchokes every 10 seconds the N leechers that have downloaded more data from him in the last 20 seconds. (ii) *balanced*, the seeder unchokes the peers following a round robin policy. Furthermore, seeders use the *optimistic disconnect* algorithm [7]. Based on this a seeder closes the connection to those peers to which it has not sent data for a long period of time (around 5 minutes). The combination of these algorithms (specially the balanced unchoking and the optimistic disconnect) aim to make the seeder communicating with as many peers as possible, not necessarily looking for *good* neighbours as happened in the leecher state. The result is that seeders show an extremely high microscopic dynamicity.

b) Macroscopic Dynamicity: Now we focus on understanding the evolution of the main graph topological parameters (clustering coefficient and characteristic path length) along the time. For this purpose Figure 7 presents the mean and the standard deviation of the clustering coefficient and characteristic path length for each torrent from our dataset (sorted by the mean of C_k/L_k in ascending order). We observe that for the major portion of the torrents the standard deviation is relatively small compared to the mean value for both the clustering coefficient and the characteristic path length. Therefore, these torrents present a low macroscopic dynamicity.

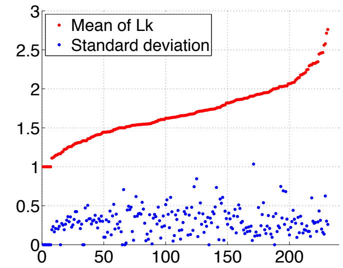
In a nutshell, the live BitTorrent swarms show a high microscopic dynamicity produced by peers varying a large portion of its neighbourhood even in short periods of time. However, this does not affect the macroscopic graph structure of the torrent that remains stable with a low variance in the showed clustering coefficient and characteristic path length.

IV. BitTorrent clustering phenomenon in live BitTorrent swarms

In the previous section we have performed a thorough analysis of the static and dynamic characteristics of the overlay topology of live BitTorrent swarms. We have demonstrated that current swarms do not have a random graph topology. This suggests that a clustering phenomenon is taking place in BitTorrent swarms. In this section we devote our effort to understand this clustering effect. We will present a novel methodology that allows the identification of the neighbours with which a peer clusters together. Furthermore we quantify the cluster effect, this is, we reveal with how many neighbours a given peer maintains a stable relationship.



(i) C_k



(ii) L_k

Fig. 7: Mean an standard deviation of the clustering coefficient (C_k) and characteristic path length (L_k) for each torrent

To the best of our knowledge this is the first paper that addresses the clustering phenomenon on BitTorrent swarms in the wild.

A. Methodology

First of all, we divide the lifespan of a peer in a BitTorrent swarm in two different phases: leecher and seeder phase. This is, the time the peer is leecher and seeder respectively.

On the one side, in Section III-D we have described the main algorithms applied by a BitTorrent leecher: unchoke, optimistic unchoke and optimistic connect. The objective of these algorithms is to find a set of *good* neighbours that provides the highest possible download rate to the peer and keep the interaction with them. Specifically, the optimistic unchoke algorithm tries to find new neighbours that increase the download rate of the peer; the unchoke algorithm is responsible to interact with the set of *good* neighbours (i.e. upload traffic to them) to assure they keep sending data to the peer; the optimistic connect substitutes the useless peers by new ones that are potential candidates to interact with. Our hypothesis is that these algorithms converge to a set of *stable* neighbours with which the peer systematically exchange traffic with, thus leading to a clustering phenomenon happening in BitTorrent swarms. On the other side, seeders apply different algorithms: *proportional* or *balanced* unchoke and optimistic disconnect. The final objective of the combination of these algorithms (specially, in the case of combining *balanced* unchoke and optimistic disconnect) is letting the seeder to distribute pieces of the content homogeneously among the leechers. Therefore, our hypothesis is that during the seeder phase the peer tends to have a few (or none) stable neighbours. This suggests that seeders do not (typically) participate as members of clusters.

We have collected all the routing table snapshots for almost 50k peers from our dataset. For each one of these peers we have calculated the percentage and absolute number of *stable* neighbours. These are neighbours that appear within all the collected routing tables for a given peer. It is also worth to note that we distinguish the leecher and the seeder phases since

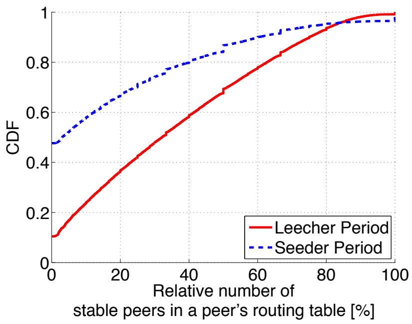


Fig. 8: CDF of the percentage of stable neighbours in a peer's routing table

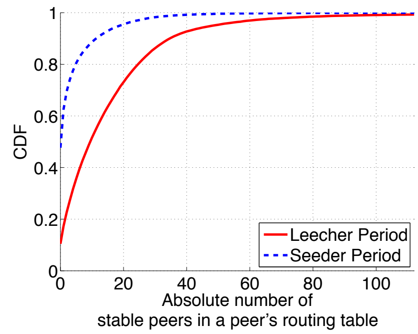


Fig. 9: CDF of the absolute number of stable neighbours in a peer's routing table

we expect different results for each of them.

B. Quantifying the clustering phenomenon happening in live BitTorrent swarms

One the one hand, Figure 8 shows the CDF of the percentage of stable neighbours for the 50k considered peers. The percentage is calculated as the number of stable peers divided by the median size of the peer's routing table along its lifespan. Note that we have differentiated the seeder and the leecher phases. On the other hand, Figure 9 presents the CDF of the absolute number of stable neighbours for the 50k considered peers, again distinguishing between seeder and leecher phases. The results validate our hypothesis: on the one side, leechers keep an important percentage (30% in median) of *stable* neighbours. These are neighbours with which the peer systematically exchange traffic and thus clusters together. On the other side, seeders have much lower percentage of stable peers, in fact half of the seeders do not have any *stable* neighbour. However, there are some seeders that keep an important percentage of stable neighbours. These seeders participate in small swarms where all the nodes knows each other (in these cases the stable neighbours do not reach typically the 100% because of the churn effect).

If we focus on the absolute number of stable peers, we observe that, on the one hand, leechers have in median 10 stable neighbours. This value is higher than the typical number of unchoke slots used by the leecher, i.e. 4. Therefore a BitTorrent leecher multiplexes its resources (i.e. unchoke slots) in time in order to attract a larger number of peers to obtain pieces from. On the other hand, 60% of the seeders presents a number of stable peers ≤ 1, supporting our hypothesis that seeders do not (typically) participate in the formed clusters.

In a nutshell, this section has demonstrated that: (i) clustering is happening in live BitTorrent swarms, (ii) clustering is linked to the data exchange procedure of BitTorrent and, (iii) leechers tend to cluster with a number of neighbours typically larger than the number of unchoke slots whereas seeders do not (typically) form clusters.

V. Does Locality happen in live BitTorrent Swarms?

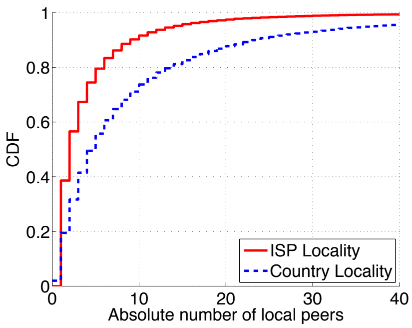
The random bootstrapping used in p2p applications, and more specifically in BitTorrent is unnecessarily pushing a lot of traffic to the transit links of ISPs increasing the operational cost of them [2], [9]. Some solutions [3], [15] have been proposed in order to make a BitTorrent node selects (when available) peers within its own ISP, thus keeping as much BitTorrent traffic as possible within the ISP. These techniques are named *Locality* techniques.

Furthermore, some other works have studied the performance of these Locality techniques [5] [12]. However, to the best of the author knowledge any of the previous works have analysed the level of locality that exist in live BitTorrent swarms. Most of the previous works have assumed that live BitTorrent swarms are random graphs in which the likelihood of having a *local* neighbour from the same ISP in the peer's neighbourhood is the same as having a *remote* neighbour from a different ISP. However, we have shown in Section IV that current BitTorrent swarms are not random graphs, so our hypothesis is that peers' neighbourhoods are typically biased towards locality, i.e. the probability of having a local neighbour is higher than the probability of having a remote neighbour. Understanding this is critical in order to understand what is the actual gain that we can obtain from applying the Locality techniques proposed so far.

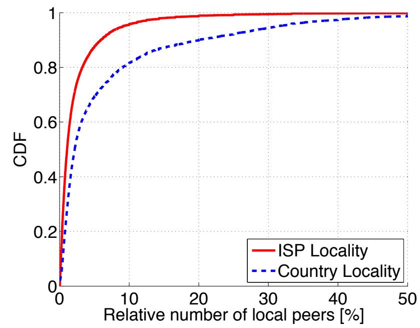
Moreover, it is interesting to understand the level of locality which exists at the traffic exchange level, this is, at the clustering level. Towards this end we analyse the location information (ISP and country) of the stable neighbours to see if there is a locality bias in the selection of those neighbours with which a peer exchanges traffic.

A. Locality-biased graph topologies

We first study whether live BitTorrent swarms presents an ISP- or country- biased overlay composition, this is whether the number of connections between peers from the same ISP (or country) is higher than expected from a pure random process. For this purpose we use the following methodology, that is based on [5]. Lets denote *V(T)* as all the peers participating in a torrent swarm *T*. We also define *V(I,T)* as a subset of *V(T)* which includes all the peers belonging to the same ISP *I* and *V(C,T)* as a subset of *V(T)* which contains all the peers belonging to the same country *C*. On the one hand we calculate the expected number of local nodes from the same ISP (E_i) and the same country (E_c) that a given peer should have under a pure random assignment procedure on

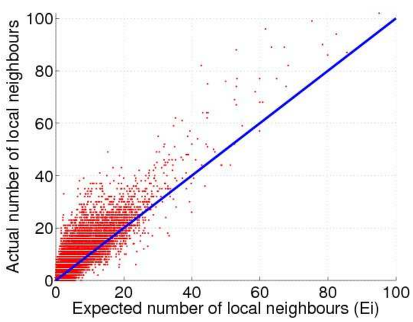


(i) Absolute

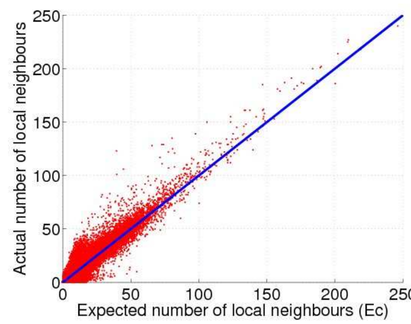


(ii) Relative

Fig. 10: Number of local available nodes for peers in unlocalised torrents: *(i)* absolute and *(ii)* relative [%]



(i) ISP



(ii) Country

Fig. 11: Expected number of local neighbours vs actual number of local neighbours: *(i)* ISP and *(ii)* Country locality

each of its routing table snapshots. This is given by the mean of the Hyper-Geometric distribution[4]. On the other hand, we calculate the actual number of local nodes from the same ISP (I) and from the same country (C) that appears in the peer's routing table. We define a simple metric named the *Locality Ratio* (LR) that captures whether the neighbourhood of a given peer is biased towards having more local nodes than expected from a random bootstrapping process. More specifically, we define LR_I (ISP Locality Ratio) as E_i/I and, LR_C (Country Locality Ratio) as E_c/C. In short, a peer with an $LR_I > 1$ and $LR_C > 1$ has a higher than expected number of peers in its neighbourhood from its ISP and country respectively. Then, if this happens for a majority of peers we can conclude that a Locality effect exists in live BitTorrent swarms.

Prior to present our results we perform a filtering to avoid biasing the obtained results. It has been reported in [5] the existence of *unlocalisable* torrents (snapshots of torrents) for a given peer. Locality is by definition (almost) impossible to happen for this peer, since the number of local nodes in particular torrent snapshot is 0 or very low. Therefore, we have removed from our dataset all those peers located in unlocalisable torrents, this is torrents with 0 local peers or with values of E_i or $E_c < 1$ (the expected number of local nodes to appear in the routing table of the peer is very low). To validate our filtering technique we have measured the absolute and relative (as a percentage of whole population) number of local nodes for those filtered peers. The results are shown in Fig 10. For the case of ISP locality we observe that the filtered peers have (in median) 1.5 local nodes to select as neighbours. Furthermore, these local nodes represents less than 2% of the torrent population. The results for country locality are similar. Therefore, we conclude that this filtering technique is removing peers associated to unlocalisable torrents.

Figure 11 shows E_i vs I and E_c vs C for each peer's routing table snapshot. We observe that most of the peers have a locality-ISP biased neighbourhood, whereas this bias is slightly lower when we consider the country criteria. Therefore we can conclude that **locality-ISP bias is happening in live BitTorrent swarms**. To the best of the authors knowledge this is the first study reporting this critical observation.

To gain more insight in this phenomenon we investigate how much biased are these neighbourhoods towards having peers from the same ISP or country. Figure 12 presents the median LR_I and LR_C of each peer across all its routing table snapshots.

We can observe that an important portion of peers (45%) have a surprisingly high LR_I over 1.3. This means that they have 30% more local neighbours from its own ISP than expected. This percentage gets reduced when looking at the locality at the country level, where only 27% of peers shows LR_C over 1.3. Therefore, we can conclude that there exists

[4]The probability of getting x "successes" (i.e., local nodes) when drawing randomly W samples from a pool of N items, out of which M are "successes" is given by the HyperGeo(x, N, M, W). In our case, for a given peer N is represented by the swarm size - 1 (itself), M is represented by the number of local nodes (from the ISP or Country) -1 (itself) and W is represented by the peer's routing table size.

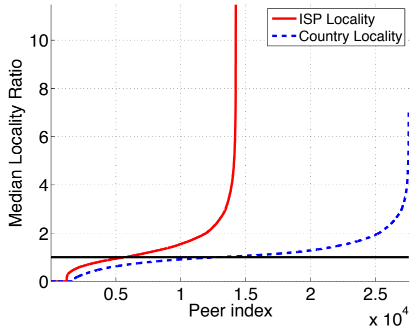


Fig. 12: Median locality ratio per peer (in ascended order)

a marked ISP-bias in the peers' neighbourhood composition. This suggests that network congestion, ISP throttling policies [8] and new Locality-biased BitTorrent implementations [3], [15] are contributing to create ISP-biased overlays. This is an important conclusion to take into consideration to accurately evaluate the actual gain that future BitTorrent locality solutions may achieve. Furthermore, the locality bias is much more pronounced at the ISP level than at the country level, thus, ISP-based factors such as ISP policies dominate in order to create the swarms' biased overlays.

Let's now analyse the demographic aspects of the observed locality phenomenon. Our main objective is to discover whether there are ISPs showing a high level of locality. For this purpose, for each ISP in our dataset we collect the absolute and relative (i.e. percentage) number of peers having a high LR_I that we name *high locality* peers. Specifically, we consider a peer as *high locality* peer if it has a $LR_I > 1.3$. Table I shows the 10 ISPs with the largest number of *high locality* peers. In addition, the table reports the percentage of *high locality* peers and the median of LR_I of the *high locality* peers for each one of the 10 ISPs. We observe the presence of major US and European ISP such as Comcast (US) or Virgin Media (UK) in the list. This suggests that some major ISPs are implementing policies in order to bias the overlay construction of the torrents in which their clients participates. Furthermore, it is worth to mention the presence of 4 different Indian ISPs in the list. Therefore, the usage of techniques to reduce the transit traffic generated by BitTorrent seems to be common among major Indian ISPs.

We repeat the same analysis at the country level for a peers with a $LR_C > 1.3$. The results are shown in Table II. India is the country with the highest number of *high locality* peers at the country level. US occupies the second position in the ranking and we also observe the presence of some European countries. It is also worth to note that more than 60% of users from Taiwan are *high locality* peers. These results are coherent with the conclusion obtained from the previous ISP analysis.

B. Locality at the traffic exchange level

The BitTorrent Locality-biased solutions proposed so far [3], [15] aim to bias the overlay structure of the BitTorrent swarms. However, the final objective of these solutions is reducing the amount of traffic going to the ISPs transit links. In the previous subsection we have shown that live BitTorrent swarms present an ISP-biased overlay topology, however we would like to go one step further and understand if there exists *locality* at the traffic exchange level. For this purpose we analyse the location of the stable neighbours for the 50k studied peers in Section IV. Remind that the stable neighbours are those node with which the peer systematically exchanges traffic. Using the same methodology explained in the previous subsection we calculate the Locality Ratio for each one of the 50k studied peers at both the ISP and Country levels.

Figure 13 shows a box plot showing the distribution of LR_I and LR_C at both the overlay construction and the exchange traffic levels. Specifically, the boxes represents the 25, 50 and 75 percentiles of the LR distribution considering locality at the ISP and country level. Firstly, we observe that the exchange traffic level also shows a locality bias that is slightly more pronounced than at the overlay construction level. The same effect is observed at the country level. Secondly, this figure confirms that the locality effect is more marked at the ISP level than at the country level. The results suggest that ISP policies (e.g. throttling transit BitTorrent traffic) along with other secondary effects such as network conditions (e.g. network congestion) and the presence of new BitTorrent clients favouring the locality effect lead to confine an important portion of the exchanged BitTorrent traffic within the ISP, thus reducing the BitTorrent transit traffic.

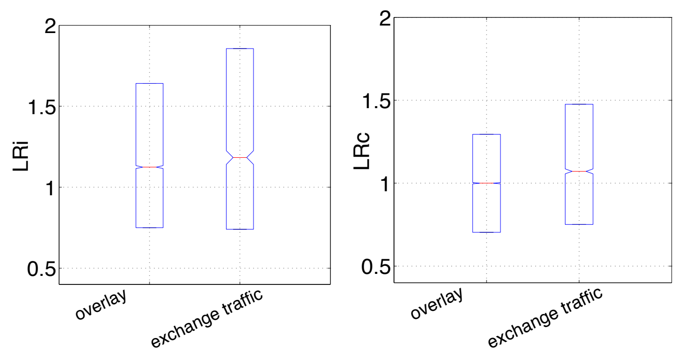


Fig. 13: Distribution of LR_I and LR_C at overlay construction and the exchange traffic levels

We have repeated the demographic analysis performed for the overlay construction level at the traffic exchange level. The results are presented in the Table III. We conclude that the observations done at the overlay construction level are also valid at the exchange traffic level: the presence of several major ISPs from India and several major US (Comcast) and European providers (Telecom Italia, Telefonica Espana) among the top 10 ISPs with a larger number of *high locality* peers. We have also performed the analysis at the country level (Table IV) and obtained similar results at the traffic exchange level as at the overlay construction level.

In a nutshell, our results confirm that existing locality policies applied by the ISPs are relatively effective. They lead to the presence of local nodes in neighbourhood composition, but what is more important, these policies help to confine the

ISP	Median	%
Bharti Broadband	2.22	79.75
NIB (National Internet Backbone)	1.77	42.40
Comcast Cable	1.65	36.80
PTCL Triple Play Project	1.89	55.44
CHTD, Chunghwa Telecom Co., Ltd.	1.89	72.19
Road Runner	1.63	36.27
Mahanagar Telephone Nigam Ltd.	1.97	42.86
RELIANCE COMMUNICATIONS	1.71	38.06
Virgin Media	1.72	38.16
SBC Internet Services	1.74	41.34

TABLE I: ISPs with the highest number of *high-locality* peers at the overlay construction level

Country	Median	%
IN	1.89	25.38
US	1.58	29.51
GB	1.66	32.71
RU	1.80	23.15
PK	1.85	47.40
CA	1.60	31.95
TW	1.85	60.96
PL	1.71	36.94
FR	1.63	37.71
SE	1.58	15.83

TABLE II: Countries with the highest number of *high-locality* peers at the overlay construction level

BitTorrent traffic within the ISP.

VI. Conclusions

This paper presents the first comprehensive study of the topology structure of live BitTorrent swarms based on large scale real measurements including data from 250 torrents. Furthermore, based on the real data collected, the paper analyses two critical aspects of BitTorrent swarms structure: the clustering and locality phenomena.

The obtained results demonstrate that the real BitTorrent swarms are neither random graphs nor small worlds. A more detailed analysis have revealed that on the one hand the size affects the swarm structure making large swarms more clustered. On the other side, the swarm structure remains stable along the time if we consider macroscopic metrics such as the clustering coefficient and the characteristic path length. This is a surprising result due to the high dynamicity observed at the microscopic level where the peers change in median 50% of its neighbours every 10 minutes. Furthermore, the paper shows that peers tend to form clusters when they are leechers whereas the seeders typically do not do it. Finally, our results reveal the existence of ISP locality biased overlays in BitTorrent swarms and what is more relevant, even more marked ISP locality effect at the exchange traffic level. Moreover, we observe an important presence of Indian and major European and American ISPs among those showing a higher level of locality. This locality effect may be produced by different reasons such as the network conditions (e.g. congestion), the ISPs policies (e.g throttling) or the implementation of locality-biased algorithms at both BitTorrent clients and trackers side.

ISP	Median	%
NIB (National Internet Backbone)	1.93	51.84
Bharti Broadband	2.53	76.52
CHTD, Chunghwa Telecom Co., Ltd.	1.69	61.11
Comcast Cable	1.63	31.34
Telecom Italia	1.65	30.91
Bredbandsbolaget AB	1.55	45.83
Telefonica de Espana	1.57	25.64
Road Runner	1.77	36.00
PTCL Triple Play Project	2.28	41.18
Mahanagar Telephone Nigam Ltd.	2.47	66.67

TABLE III: ISPs with the highest number of *high-locality* peers at the traffic exchange level

Country	Median	%
IN	1.93	50.83
US	1.70	30.15
PL	1.83	59.32
RU	1.76	20.43
GB	1.83	36.44
SE	1.55	13.00
TW	1.71	48.00
DE	1.56	21.65
ES	1.58	9.94
CN	1.68	30.91

TABLE IV: Countries with the highest number of *high-locality* peers at the traffic exchange level

VII. Acknowledgement

The authors would like to thank Sebastian Kaune for his help on developing the crawler application. The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) through the TREND NoE project (grant agreement n. 25774) and the Regional Government of Madrid through the MEDIANET project (S-2009/TIC- 1468).

References

[1] A. Al-Hamra, A. Legout, and C. Barakat. “understanding the properties of the bittorrent overlay”. *Technical Report, INRIA, Sophia Antipolis*, July 2007.

[2] 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 *ICDCS'06*.

[3] D. R. Choffnes and F. E. Bustamante. Taming the torrent: a practical approach to reducing cross-isp traffic in peer-to-peer systems. *ACM SIGCOMM 2008*.

[4] R. Cuevas, M. Kryczka, A. Cuevas, S. Kaune, C. Guerrero, and R. Rejaie. Is content publishing in bittorrent altruistic or profit driven? In *The 6th International Conference on emerging Networking EXperiments and Technologies (CoNEXT)*, Philadelphia, USA, 2010.

[5] R. Cuevas, N. Laoutaris, X. Yang, G. Siganos, and P. Rodriguez. Deep diving into bittorrent locality. *IEEE INFOCOM 2011, Shanghai, China*.

[6] C. Dale, J. Liu, J. Peters, and B. Li. Evolution and enhancement of bittorrent network topologies. *IEEE IWQoS'08, Enschede, The Netherlands*, June 2-4, 2008.

[7] P. Dhungel, X. Hei, D. Wu, and K.W. Ross. A measurement study of attacks on bittorrent seeds. *ICC, Kyoto, Japan*, June 2011.

[8] M. Dischinger, A. Mislove, A. Haeberlen, and K. P. Gummadi. Detecting bittorrent blocking. *ACM IMC 2008*.

[9] T. Karagiannis, P. Rodriguez, and K. Papagiannaki. Should internet service providers fear peer-assisted content distribution? In *ACM IMC'05*.

[10] A. Legout, N. Liogkas, E. Kohler, and L. Zhang. Clustering and sharing incentives in bittorrent systems. *ACM SIGMETRICS 2007, San Diego*, June 12-16, 2007.

[11] MaxMind. http://www.maxmind.com.

[12] M. Piatek, H. V. Madhyastha, J. P. John, A. Krishnamurthy, and T. Anderson. Pitfalls for isp-friendly p2p design. *HotNets-VIII*.

[13] Sandvine. Fall 2010 Global Internet Phenomena Report. Available at: http://www.sandvine.com/news/global_broadband_trends.asp.

[14] G. Urvoy-Keller and P. Michiardi. Impact of inner parameters and overlay structure on the performance of bittorrent. *9th IEEE Global Internet Symposium 2006 in conjunction with Infocom 2006*.

[15] H. Xie, Y. R. Yang, A. Krishnamurthy, Y. Liu, and A. Silberschatz. P4p: Provider portal for applications. *ACM SIGCOMM 2008*.

[16] C. Zhang, P. Dhungel, D. Wu, and K.W. Ross. Unraveling the bittorrent ecosystem. *IEEE Transactions on Parallel and Distributed Systems*, 2010.