Connectivity properties of real BitTorrent swarms

Ruben Cuevas[1], Michal Kryczka[12],
Angel Cuevas[1], Carmen Guerrero[1], Arturo Azcorra[12]
[1] Universidad Carlos III de Madrid
Madrid, Spain
[e-mail: {rcuevas,acrumin,carmen.guerrero,azcorra}@it.uc3m.es]
[2] Institute IMDEA Networks
Madrid, Spain
[e-mail: rcuevas @it.uc3m.es]
*Corresponding author: Rubén Cuevas

Received November 17, 2012; revised August 9, 2013; accepted September 9, 2013; published September 30, 2013

Abstract

BitTorrent is one of the most important applications in the current Internet. Despite of its interest, we still have little knowledge regarding the connectivity properties of real BitTorrent swarms. In this paper we leverage a dataset including the connectivity information of 250 real torrents and more than 150k peers to carefully study the connectivity properties of peers. The main topology parameters of the studied swarms suggest that they are significantly less resilient than random graphs. The analysis of the peer level connectivity properties reveals that peers continuously change more than half of their neighbours. Furthermore, we also find that a leecher typically keeps stable connections with a handful of neighbours with which it exchanges most of its traffic whereas seeders do not establish long-term connections with any peer so that they can homogeneously distribute chunks among leechers. Finally, we have discovered that a significant portion of the studied peers (45%) have an important locality-biased neighbourhood composition.

Keywords: P2P, BitTorrent, Internet Measurements

A preliminary version of this paper appeared in IEEE P2P 2011, August 31 – September 2, Kyoto, Japan. The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) through the eCOUSIN project (grant agreement n. 318398), the Spanish Ministry of Economy and Competitiveness through the eeCONTENT project (TEC2011-29688-C02-02) and the Regional Government of Madrid through the MEDIANET project (S-2009/TIC- 1468).

http://dx.doi.org/10.3837/tiis.2013.09.010

1. Introduction

BitTorrent is one of the most used applications in the current Internet and is responsible for an important portion of the upstream and downstream traffic as revealed by recent reports [1]. The significant footprint of BitTorrent in the Internet has motivated researchers and practitioners to dedicate an important amount of effort to understanding and improving BitTorrent. However, despite this effort, we still have little knowledge regarding the connectivity properties exhibited by *real* BitTorrent swarms at both swarm and peer level. Due to the difficulty in collecting the required information from real swarms, most of the existing works that analyse connectivity properties are based on simulations [2] or experiments in controlled environments [3], [4]. As a result, they are likely to miss some of the effects affecting BitTorrent swarms in the wild. To the best of the authors' knowledge, there is just a previous study that evaluates few properties of the overlay topology (i.e., swarm level connectivity) of real BitTorrent swarms [5].

The analysis of the connectivity properties at the swarm level (i.e., overlay topology) and at the peer level (i.e., peers' neighbourhood composition) in real BitTorrent swarms can reveal important information such as: (*i*) the resilience of a swarm to different events such as being partitioned [3], (*ii*) the efficiency of swarming and neighbour selection algorithms and (*iii*) the locality-bias exhibited by current BitTorrent swarms. Furthermore, the characterization of these properties is of interest to: (*i*) researchers and practitioners designing (improving) new (existing) BitTorrent clients or related algorithms; (*ii*) companies using BitTorrent for critical functions such as software release, backups distribution or content replication [6][7][8][9] and (*iii*) Internet Service Providers (ISPs) carrying BitTorrent traffic.

In this paper, we first present a methodology to collect the connectivity information at both the swarm and the peer level for the entire lifespan of a real torrent. Specifically, we discover new torrents just after their birth by using the RSS service of the most important BitTorrent portal, namely The Pirate Bay. Afterwards, we exploit the Peer Exchange (PEX) extension of the BitTorrent protocol to gather the set of neighbours for each peer. PEX is a gossiping technique whose main goal is to allow peers to exchange their list of neighbours so that they can learn about other participants in the swarm without contacting the tracker. Note that PEX has been implemented by most of the existing BitTorrent clients and in particular by the most popular ones such as uTorrent or Azureus [10]. The information collected from PEX (i.e., a peer's neighbourhood) provides the connectivity information at the peer level. Furthermore, by aggregating the neighbourhood information collected from every peer in a swarm we are able to build the overlay topology of that swarm (i.e., swarm level connectivity). We retrieve the information from each active peer every 10 minutes and then study the dynamic evolution of both the overlay topology of the swarm and the composition of each peer's neighbourhood.

We have applied the described methodology to collect the connectivity information of 250 real torrents, including more than 150k peers, since their birth during a period of 15 days. This datasets constitutes the basis for our analysis, which is divided into two parts. In the first part we analyse one of the most important features of a graph: its resilience. Specifically, in our study we evaluate the resilience of real BitTorrent swarms to be partitioned. To this end we consider two types of events: churn and an eventual attack represented by a random node removal and a selective node removal processes, respectively.

In the second part of the analysis we focus on the connectivity properties at the peer level. First, we study the variability in the composition of peers' neighbourhoods along time, paying special attention to the presence of stable neighbours. Second, we perform a thorough study in order to understand whether the level of locality observed in the composition of the peers' neighbourhood is higher/lower than the expected from the random neighbour selection process implemented by default in BitTorrent. Furthermore, we group those peers presenting a significant positive deviation in the exhibited locality by their neighbourhoods into ISPs and countries. This simple technique allows us to discover ISPs and countries whose peers are more likely to show locality-biased neighbuorhoods:

Our main contributions and findings can be summarized as follows:

- Real BitTorrent swarms are fully resilient to be partitioned under a random removal process (e.g., churn). However, they are significantly less resilient than random graphs to a highest-degree node removal process (e.g., an attack). This result is of interest to researchers and practitioners that work on the enhancement of the different aspects (including resilience) of BitTorrent. Furthermore, our observation is also useful for those companies using BitTorrent for critical functions such as software release or content replication.
- Both leechers and seeders change a significant portion of their neighbours continuously. This is a consequence of the combination of the different neighbour selection algorithms implemented by current BitTorrent clients.
- Leechers keep stable connections with a handful of other peers with which they exchange most of the traffic. This number of users is typically larger than the commonly used number of unchoke slots (4). Therefore, leechers tend to multiplex their resources (i.e. unchoke slots) to optimize their download performance. Furthermore, seeders typically do not keep stable connections with any peer so that they homogeneously distribute pieces among the participants in the swarm. Again, this result is useful for the design and validation of improvements on existing BitTorrent algorithms or the design of new algorithms.
- 45% of the analysed peers present at least 30% more local neighbours than expected from a pure random neighbour selection process. Furthermore we observe an even more pronounced locality deviation in the set of stable neighbours (remember that the stable neighbours are those with which a peer exchange most of its traffic). This suggests that ISP locality enforcement policies (e.g. throttling), the proliferation of successful locality-aware P2P clients or plugins such as Ono [12] and other networking effects such as congestion are leading BitTorrent peers to exhibit a higher locality-bias in the composition of their neighbourhoods. This result is of interest for ISPs and to those researchers and practitioners working in the definition of locality-aware BitTorrent clients.

The rest of the paper is organized as follows. Section 2 briefly introduces the neighbor selection mechanisms implemented in BitTorrent. Section 3 describes our measurement infrastructure and methodology as well as the dataset used along the paper. Section 4 evaluates the resilience of BitTorrent swarms and compares it with that observed for random graphs. Afterwards, Section 5 analyses the stability in both the overlay topology and the peers neighbourhood composition along time. Section 6 studies the locality-biased composition of peers' neighbourhoods. Finally, Section 7 presents the related work and Section 8 concludes the paper.

2. Background of Neighbor Selection Algorithms in BitTorrent

1) Leecher phase algorithms: The *unchoking* algorithm makes a leecher select N (typically 4) neighbours to upload chunks to every 10 seconds. These neighbours are then *unchoked* whereas the rest of the node's neighbours are *choked* and will not receive data from the peer. The BitTorrent peer unchokes the N neighbours from whom it received most data in the last 20 seconds. Therefore, the unchoking algorithm tries to keep *good* neighbours for exchanging traffic with. Furthermore, every 30 seconds the BitTorrent peer performs an *optimistic unchoke*. That is, it chooses a random neighbour and uploads data to it. The optimistic unchoke allows nodes to discover better peers to exchange traffic with. Moreover, the most important BitTorrent clients such as Vuze or uTorrent utilize the *optimistic connect* [18] 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. The combination of the three described algorithms leads a leecher to identify and drop those peers from which the leecher is not obtaining enough good performance.

2) Seeder phase algorithms: BitTorrent seeders apply different unchoke strategies depending on the implementation. The most extended strategies are: (i) *proportional* in which the seeder unchokes every 10 seconds the N leechers that have downloaded more data from it in the last 20 seconds and (ii) *balanced* in which the seeder unchokes peers following a round robin policy. Furthermore, seeders use the *optimistic disconnect* algorithm [18]. Based on this algorithm 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 (especially the balanced unchoking and the optimistic disconnect) aims to make the seeder communicating with as many peers as possible, rather than looking for *good* neighbours as occurs in the leecher state.

3. Measurement Methodology

The aim of our measurement study is to retrieve the topology graph of real BitTorrent swarms. For this purpose, we collect the neighbours list (or neighbourhood)[1] of each peer in the swarm by using the Peer Exchange (PEX) extension of the BitTorrent protocol. In the rest of the section we provide a detailed description of both the measurement infrastructure and the methodology. For a full description of the BitTorrent ecosystem we refer the reader to [13] and [14].

3.1 Measurement Infrastructure

To achieve the described goal, we need to build an architecture that implements mechanisms to (i) identify the torrent (or swarm) to be monitored, (ii) identify the peers within the swarm and (iii) collect the connectivity information of those peers. To this end we use a *Master-Slave* scheme in which the *Master* is responsible for learning new torrents from a BitTorrent portal and contacting the tracker that manages the swarm associated with 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) to monitor. The *Slave* tries to connect to each one of these peers and to retrieve the peer's neighbours list among other information. In particular, our measurement infrastructure is formed by 12 virtual

[1] In the rest of the paper we will use neighbours list, neighbours set and neighbourhood undistinguishably.

machines, with a single public IP address each, installed in 3 different physical machines. One of the VMs acts as *Master* whereas the other 11 are *Slaves*.

3.2 Measurement Methodology

The methodology of our measurements is based on the one presented in [15] and has several similarities with the methodology used in [5] and [33]. In order to learn new torrents we decided to use The Pirate Bay portal. This is the most important BitTorrent portal according to Alexa ranking[2] and some research studies [14]. The Pirate Bay offers an RSS service where each new torrent is announced as soon as it is uploaded to the portal. Our *Master* is subscribed to this RSS service so that it can discover new torrents after their 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 with 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. Each reply from 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*. The *Master* has to schedule the connection of the different *Slaves* to a given peer. The *Slaves* contribute no chunks to other peers, thus, if a *Slave* connects a 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 any peer from blacklisting 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 reachable, and gathers the neighbours list for those peers supporting the Peer Exchange eXtension (PEX). PEX is an extension to the BitTorrent protocol that allows peers to exchange their neighbours lists. This reduces the load at the tracker since peers are able to learn about other peers without asking the tracker. In particular, a *Slave* retrieves the list of *connected* and *disconnected* neighbours from the PEX messages. Note that the list of connected neighbours includes those nodes with which the peer has currently an established connection, i.e., the peer's current neighbours. Hence, in our analysis we use exclusively the list of connected peers and refer to them as peer's neighbours list (or neighbourhood). It is worth noting that most BitTorrent clients and particularly the most popular ones such as uTorrent and Vuze support PEX [10], thus we are able to retrieve the neighbourhood 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 be crawled.

As mentioned earlier, there are peers that are not reachable. These are nodes behind a NAT that do not use manual or automatic (e.g., UPnP) techniques to open ports in the NAT. Therefore if we fail to connect to a given IP address 5 times we declare this peer as *unreachable*. Furthermore, due to the 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.

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

3.3 Dataset Description

We have applied the described measurement methodology to 250 consecutively published torrents, learnt from The Pirate Bay's RSS service from December 20th 2010 until January 4th 2011. From this set of torrents we were able to learn the neighbours list for more than 150k peers. Specifically, we are able to derive the evolution of the neighbourhood composition for each peer since, as explained above, we periodically retrieve every peer's neighbours list. Furthermore we map the peers' IP addresses to their country and ISP using the MaxMind Database [16]. Finally, it is worth mentioning that roughly 70% of peers are unreachable and thus, we cannot directly collect their neighbourhoods snapshots. However, these peers establish most of their connections to other reachable peers (i.e., it is unlikely that two unreachable peers establish a connection since it would require to use complex NAT-traversal mechanisms). Then, unreachable peers are members of the neighborhoods of reachable peers that our tool is able to collect. We identify the IP address of each node that has a given unreachable peer as neighbour. The identified IP addresses form the neighborhood of that unreachable peer. As indicated above, this technique only misses those connections to other unreachable peers (that are unlikely to be established).

3.4 Representing BitTorrent Swarm as a Graph

We represent a BitTorrent 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 neighbourhood of each reachable peer every 10 minutes; therefore, we are able to present a swarm's connectivity matrix evolution over time in 10 minutes intervals. For each snapshot of the connectivity matrix we calculate the following standard parameters used in graph theory studies: clustering coefficient C_K and characteristic path length L_K.

4. Resilience of Real BitTorrent Swarms

In this Section we investigate one of the most relevant performance aspects, namely the resilience of real BitTorrent swarms. Resilience can be measured in different manners, in this paper we specifically analyse the resilience of real BitTorrent swarms to be partitioned and compare it to the resilience shown by equivalent random graphs, that are known to be resilient topologies.

We explore two scenarios: nodes failures or churn modeled as a random removal process and an eventual attack modeled as the selective removal of the highest degree nodes. The implementation of BitTorrent for file-sharing over Internet includes some mechanisms that help to address the problem of overlay partitioning (e.g., PeerExchange, DHTs or querying the tracker when the size of the neighbourhood goes below a threshold). However, BitTorrent (or modified versions of it) is being used for multiple purposes in different contexts (e.g., distribution of data across datacenters or software distribution within a company) and will be eventually used in new contexts in the future. These specific implementations of BitTorrent may or may not include the previous mentioned mechanisms and may be susceptible to attacks or failures of the involved nodes. Therefore, understanding the resilience of BitTorrent overlays is of high interest in the future development of BitTorrent (or modified version of it)

in diverse contexts.

4.1 Methodology

We analyse the resilience of a real BitTorrent swarm snapshot and an equivalent (same number of vertices and edges) Erdos-Reni [17] random graph to two type of events: (*i*) a random removal process in which we sequentially remove nodes selected at random until the graph is partitioned into (at least) two separated components; and (*ii*) a selective removal process in which we sequentially remove the node having the highest degree until the graph is partitioned into (at least) two separated components. The first event can represent, for instance, a scenario in which several (random) peers fail or leave the system (i.e., churn) whereas the second event can represent an attack against the swarm.

In order to quantify the resilience of a given graph to be partitioned we use two different metrics:

- K, represents the total number of peers that need to be removed in order to partition the graph.
- k, represents the percentage of peers that need to be removed to partition the graph. Therefore $k = 100\ K/N$ where N is the number of peers forming the swarm snapshot.

Note that we will refer to K_{real} (k_{real}) and K_{rand} (k_{rand}) as the values of K (k) associated with a real swarm snapshot and its equivalent random graph, respectively. Due to the computational cost of these experiments we perform our study considering a random set of 400 snapshots in a stable phase (i.e., after the initial flash-crowd phase) from the 250 torrents forming our dataset. In addition, to minimize the impact of abnormal peers (e.g., new incomers or nodes behind a NAT) we consider only those peers having a degree ≥ d. We have perform the experiments with d = 2, 5, 10 and 15. Although the obtained results present some quantitative differences for different values of d (the higher d is, the higher the number of nodes to be removed to partition the graph is), they are qualitatively consistent. Therefore, due to space constrains in this paper we present results for d = 5.

4.2 Random Node Removal

Given the random nature of this removal process we perform 10 simulation runs for each real swarm snapshot. Furthermore, for each snapshot we generate 10 Erdos-Reni random graphs with the same number of vertices and edges and for each of them we simulate 10 runs of the random removal process. Hence, for each snapshot and its equivalent random graphs we can calculate the average values of K and k as well as their standard deviations.

Fig. 1(a) shows in a scatter plot the average value of k_{rand} vs the average value of k_{real} for all the analysed swarm snapshots. We observe that the value of k for most of the analysed snapshots is located in the point (100,100). This means that both real snapshots and equivalent random graphs are fully resilient to a random removal process. There are only a few (2%) real snapshots that are not fully resilient, but even in these ones more than 85% of the nodes must be removed to partition the graph. Finally, it is worth to note that the standard deviation for both k_{real} and k_{rand} is negligible for all the studied snapshots (in particular, the standard deviation for the number of removed nodes to partition the graph is lower than 1 for all studied snapshots).

Hence, the obtained results suggest that real BitTorrent swarms are in most of the cases fully resilient to a random removal process of peers that could be produced by different factors such as churn, peers' failures or an attack. This result is consistent with previous results

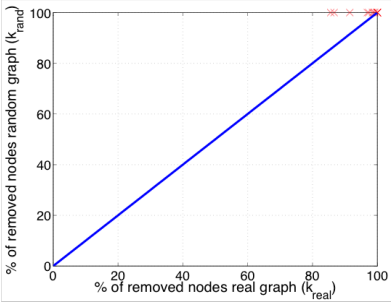
obtained in a controlled environment [3].

4.3 Highest Degree Node Removal

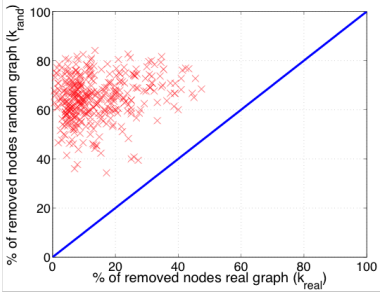
In this case, the removal process for a given real snapshot is deterministic, therefore we perform a single run for each real snapshot. Again, we generate 10 different equivalent random graphs for each analysed snapshot. For each one of these random graphs we perform a single run of the removal process since as said before it is a deterministic process. Therefore, in this case we calculate the standard deviation only for K_{rand} and k_{rand}. Again, the standard deviation for the number of removed nodes to partition the graph is lower than 1 for all studied snapshots.

Fig. 1(b) shows in a scatter plot the average value of k_{rand} vs the value of k_{real} for all the analysed swarm snapshots. We observe that random graphs are significantly more resilient to the considered removal process than real snapshots. In particular, k_{rand} ranges roughly between 40% and 80% whereas k_{real} ranges between 1% and 50%.

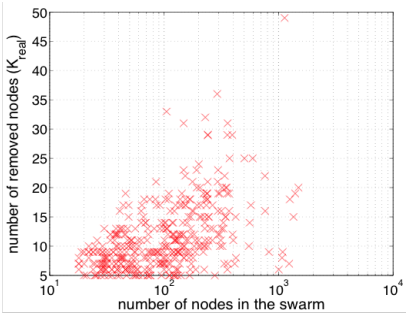
Moreover, we analyse whether the size of the torrent has any impact in its resilience. Towards this end, Fig. 1(c) and Fig. 1(d) present scatter plots of the value of K_{real} and k_{real} in front of the snapshot size for every analysed real swarm snapshot considering a highest-degree removal process, respectively. Note that Fig. 1(c) presents a log-scaled x-axis whereas Fig. 1(d) uses a log-log scale. On the one hand, there is a surprisingly small correlation between K_{real} and the size of the swarm that suggests that for large swarms the number of nodes to be removed to partition the graph is in the same order of magnitude than in the case of small swarms. This low correlation leads to the results observed in Fig. 1(d) in which we see that the relative number of peers to be removed decreases significantly with the size of the swarm. Therefore, if we consider that the selective node removal process introduced in this Section represents an attack to a BitTorrent swarm, we conclude that the attacker would need roughly the same resources to perform an attack independently of the size of the swarm under attack. This observation is consistent with the fact that larger BitTorrent swarms are typically less random than small ones as shown in [29]. Since random graphs are by definition resilient, the less similar the swarm structure is to a random graph the less resilient we expect it to be.



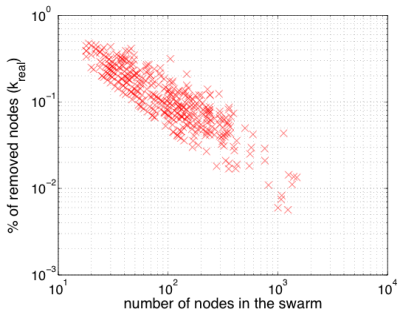
(a) Avg. k_{rand} vs Avg. k_{real} (random removal)



(b) Avg. k_{rand} vs k_{real} (highest-degree removal)



(c) K_{real} vs snapshot size (highest-degree removal)



(d) k_{real} vs snapshot size (highest-degree removal)

Fig. 1. Results of the resilience of real BitTorrent swarms and equivalent random graphs to random and highest-degree removal processes.

4.4 Summary and Discussion

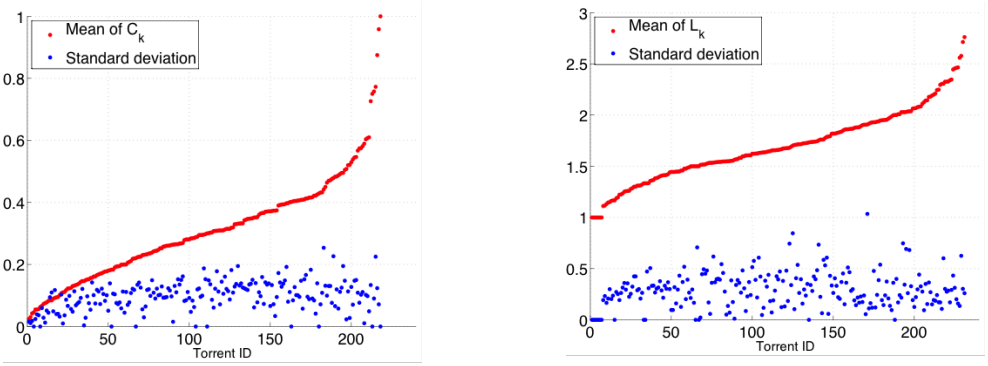
Our results suggest that BitTorrent swarms are fully resilient to a removal process of random nodes, however, their resilience to a selective removal process is significantly smaller than the one shown by random graphs.

5. Stability

In this Section we analyse the stability of BitTorrent swarms at two different levels: (*i*) *overlay topology stability*, we quantify the variability of the graph characteristics (clustering coefficient and characteristic path length) of a given torrent over time. (*ii*) *peer's neighbourhood stability*, we quantify how stable the neighbourhood of a given peer is over time and carefully analyse the subset of stable neighbours.

5.1 Overlay Topology Stability of Real Swarms

We study the stability of the main topological parameters (i.e., clustering coefficient and characteristic path length) along the time. If the variability of those parameters is low across time we can conclude that the topology is overall stable. Fig. 2 presents the mean and the standard deviation of the clustering coefficient (characteristic path length) for each torrent within our dataset sorted by the mean of C_k (L_k) in ascending order. We observe that for a 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. This suggests that overlay topologies of real BitTorrent swarms present a high stability. Note, that a more detailed studied of the overlay topology characteristics of BitTorrent swarms can be found in [29].



(a) C_K (b) L_K

Fig. 2. Mean an standard deviation of the clustering coefficient (C_k) and characteristic path length (L_k) for each torrent.

5.2 Peer's Neighbourhood Stability

As explained in Section 3 we have collected the neighbourhood (i.e., list of neighbours) for each peer every 10 minutes approximately. Then, to quantify the peer's neighbourhood stability we have computed the percentage of neighbours that appear in two consecutive neighbourhood snapshots for every peer in our dataset. In addition, we considered separately those periods in which a peer is a leecher and a seeder. Fig. 3 presents the CDF for the defined metric. The results show that around half of leechers change 50% of their neighbours between two consecutive snapshots (i.e. every 10 minutes). This percentage dramatically increases up to 80% for seeders. Hence, BitTorrent peers are continuously changing a significant portion of their neighbours. This high variability is due to two causes: (*i*) the churn effect (i.e., peers leaving and joining the swarm) and (*ii*) the combination of different neighbour selection algorithms 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) described in Section 2.

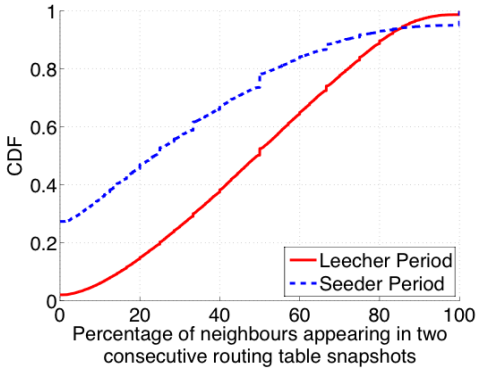


Fig. 3. CDF of the percentage of neighbours that appear in two consecutive snapshots (10 minutes apart) of a given peer

5.3 Stable Neighbours

In the previous subsection we have shown that both seeders and leechers are constantly changing a significant number of their neighbours. This leads to the reported high dynamism in the composition of peers' neighbourhoods. In this subsection, we change our focus and aim to analyse peers' *stable* neighbours. We define a stable neighbour as a neighbour that appears

in all the neighbourhood snapshots of a given peer. Note that similar to the previous subsection we consider separately the leecher and seeder phases for a peer. As we have seen the main algorithms applied by a BitTorrent leecher are 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 leecher and keep the interaction with them. Our hypothesis is that these algorithms converge to a set of *stable* neighbours with which the peer systematically exchange traffic with. Indeed, this hypothesis is supported by the results presented by Legout et al. [24] showing that peers of similar speed tend to cluster together and exchange traffic among them.

In addition, seeders apply different algorithms: *proportional* or *balanced* unchoke and optimistic disconnect. The final objective of the combination of these algorithms (especially, in the case of combining *balanced* unchoke and optimistic disconnect) is letting the seeder to distribute pieces of the content homogeneously among leechers. Then according to these algorithms, it is expected that during the seeder phase a peer tries to reach as many peers as possible and thus, tends to have a few (or none) stable neighbours.

In order to validate our hypothesis, we have collected all the neighbourhood snapshots for 50k peers within our dataset and analysed separately the leecher and seeder phases for each peer[3]. Fig. 4 shows the CDF of the time that each one of the analysed peers spend in their leecher and seeder phases, respectively. We observe that in median leechers stay in the system 70 mins whereas seeders stay 100 mins. On the one hand, the leecher phase time roughly represents the download time, although in some cases the leecher leaves the torrent before finishing the download. On the other hand, we observe that seeders typically stay in the system longer than leechers. This result is supported by previous works that shown that users dedicated to do professional seeding (thus presenting long seeding sessions) are responsible for a major portion of the content published in The Pirate Bay [15]. We refer the reader interested on the behaviour of professional seeders to [15].

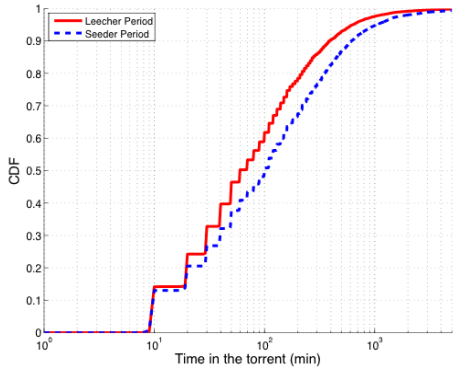


Fig. 4. CDF of time in the torrent

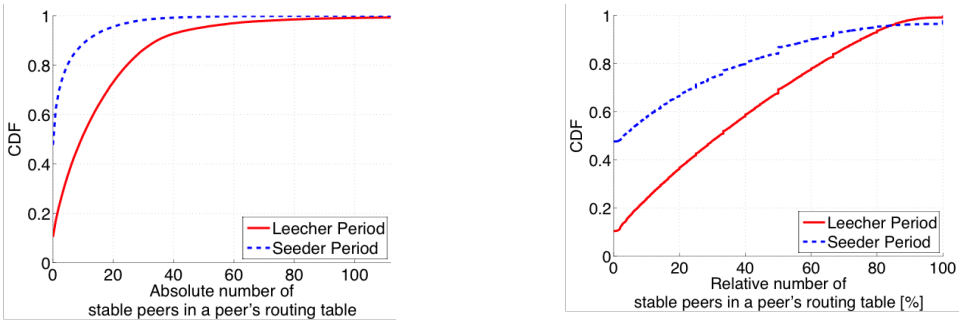
Moreover, for each peer (and phase) we have calculated two metrics: (*i*) the number of stable neighbours and (*ii*) the percentage of stable neighbours as the ratio between the number of stable neighbours and the median size of the peer's neighbourhood in the considered phase. Fig. 5(a) and Fig. 5(b) depict the CDF for these two metrics for the 50k analysed peers, respectively. The results validate our hypothesis.

Fig. 5(b) shows that leechers keep an important percentage (30% in median) of *stable* neighbours. These are neighbours with which the peer systematically exchange traffic.

[3] Note that some peers can present only one of the two phases.

Furthermore, seeders have a much lower percentage of stable neighbours, in fact half of the seeders do not have any stable neighbour. Note that few seeders present a high fraction of stable neighbors. These are seeders located in small swarms where they connect to the same set of peers a long time.

If we analyse now the number of stable peers (Fig. 5(a)), we observe that leechers have in median 10 stable neighbours. This value is higher than the typical number of unchoke slots used by the leecher (4). This observation suggests that current BitTorrent implementations lead leechers to multiplex their resources (i.e., unchoke slots) in time so that they are able to attract a larger number of peers to obtain pieces from. Moreover, 60% of seeders presents a number of stable peers ≤ 1. This finding suggests that current implementations of seeding algorithms lead seeders to communicate with as many peers as possible. By doing so, seeders aim to obtain an homogeneous dissemination of pieces among participants in the swarm (avoiding selfish behaviours by which a peer tries to retrieve all the pieces from the seeder without contributing pieces to other peers).



(a) Absolute (b) Relative

Fig. 5. CDF of the absolute number and percentage of stable neighbours in a peer's routing table

5.4 Summary and Discussion

First, we have shown that both leechers and seeders continuously change a significant portion (more than half) of their neighbours.. Surprisingly, our results reveal that the overlay topology properties (i.e., clustering coefficient and characteristic path length) of real BitTorrent swarms remain stable, despite the high dynamicity reported at the peer's neighbourhood level.

Second, our analysis of the set of stable neighbours leads to the following conclusions: (*i*) leechers tend to interact (i.e., exchange data) with a reduced number of neighbours, that is typically larger than the number of unchoke slots, in order to optimize their download rate, (*ii*) seeders present a very reduced number of stable neighbours, thus they interact with a large number of peers in order to guarantee the proper dissemination of pieces within the swarm.

The results reported in this Section reveal some interesting properties regarding the swarming efficiency driven by current BitTorrent client implementations and constitute a solid basis to design future BitTorrent implementations and compare their performance with the existing ones. Specifically, future developments can consider aspects such as: (*i*) how modifying the dynamism in the formation of peer's neighbourhood would affect the performance of the protocol in different environments (e.g., fixed vs mobile environments), and (*ii*) exploring the effects of reducing/increasing the number of stable neighbours for seeders and/or leechers.

6. Analysing Locality in Real BitTorrent Swarms

The random bootstrapping used in P2P applications, and more specifically in BitTorrent, is pushing a lot of traffic to the transit links of ISPs increasing their operational costs [19], [20]. Although the flattening of the Internet topology [32], as well as the falling marginal costs of transit traffic, may diminish the importance of the traffic locality debate in many scenarios, some ISPs have started to implement different policies (e.g., throttling) in order to minimize the impact of BitTorrent traffic in their networks [11, 30, 31]. Moreover, the research community have proposed promising solutions such as Ono [12] or P4P [21] to make a BitTorrent node select (when available) neighbours within its own ISP, thus confining as much BitTorrent traffic as possible within the ISP. In addition, other aspects such as network congestion can also affect the amount of BitTorrent traffic localized within an ISP. Note that all these factors are likely to affect just to a subset of peers in a swarm rather than the whole swarm. Therefore, in this Section we study the locality phenomenon at the peer level. Specifically we investigate whether the locality level exhibited by a peer's neighbourhood and/or set of stable neighbours is higher/lower than the expected from the random neighbours selection process implemented by default in BitTorrent.

6.1 Methodology

In this Section we consider the same 50k peers used for our analysis in Section 5C. We first study whether the neighbourhoods of BitTorrent peers present an ISP- or a country-biased composition. That is, whether the number of neighbours from the same ISP (or country) as the peer is higher than the expected compared to a purely random process. Then we repeat the analysis considering exclusively the peers' stable neighbours (i.e., neighbours with which the peer systematically exchange traffic with). Note that we use the Maxmind database [16] to map each peer to its ISP and country. Next we describe our methodology, which is based on the methodology presented in [22]:

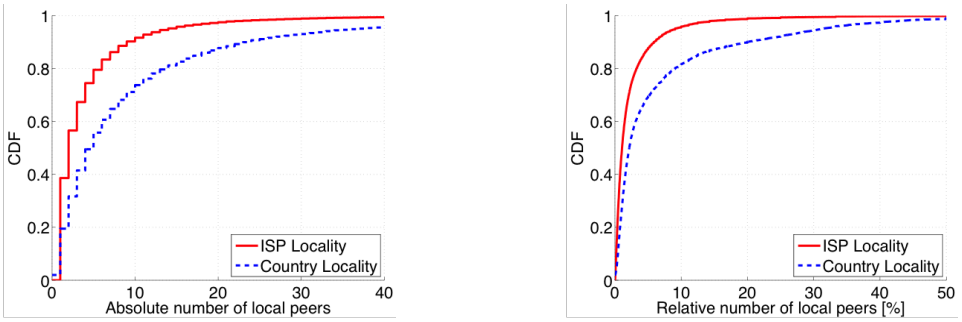
Let us consider a peer p belonging to a torrent swarm T. We denote $V(T)$ as all peers participating in T. We also define $V(I,T)$ as a subset of $V(T)$ which includes all peers belonging to p's ISP (I) and $V(C,T)$ as a subset of $V(T)$ which contains all peers belonging to the same country (C) as p.

On the one hand, we consider a random neighbours selection hypothesis that represent the expected functionality of the BitTorrent protocol. We refer to this hypothesis as H_0 in the rest of the section. In particular, we calculate the expected (i.e., average) number of local neighbours that p should have from its ISP (E_i) and its country (E_c) in each of its neighbourhood snapshots under H_0. This is given by the mean of the Hyper-Geometric distribution. 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).[4]. On the other hand, we calculate the actual number of local nodes from the same ISP (I_n) and from the same country (C_n) that appears in p's neighbourhood.

Finally, 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 selection process. More specifically, we define LR_I (ISP Locality Ratio) as I_n/E_i and, LR_C (Country Locality Ratio) as C_n/E_c. Hence, a peer with an $LR_I > 1$ and $LR_C > 1$ has a higher number of neighbours from its ISP and country than expected under H_0, respectively.

[4] 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 neighbourhood size.

Prior to presenting our results, we apply a filtering technique to avoid a bias in the obtained results. Previous works [22], [23] reported that a given peer p can be located in an unlocalizable torrent[5] (i.e., swarm snapshot). Locality is, by definition, (almost) impossible for this peer since the number of local nodes in the considered swarm snapshot is 0 or very low. Therefore, we have removed from our dataset all those peers located in unlocalizable swarm snapshots. Specifically, we consider that a swarm snapshot is unlocalizable for a peer p if: (i) there are no other local peers or (ii) p's E_i (E_c) < 1 (i.e., the expected number of local nodes in p's neighbourhood 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 6. For the case of ISP locality we observe that (in median) there are just 1.5 local nodes for the filtered peers. Furthermore, these local nodes represent less than 2% of their torrents population. The results for country locality are similar. Therefore, we can conclude that our filtering technique successfully removes peers located in unlocalizable torrents.



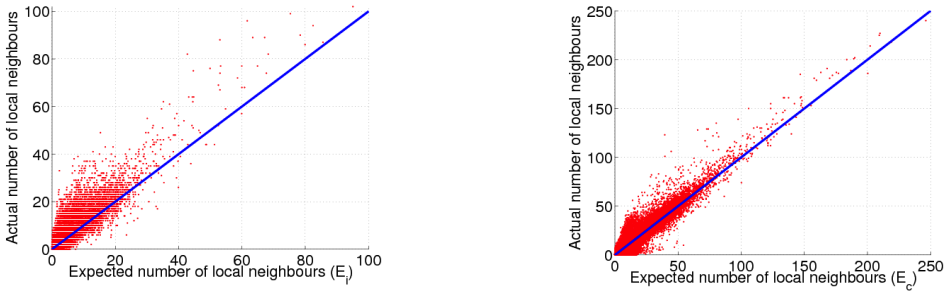
(a) Absolute (b) Relative

Fig. 6. Number of local available nodes for peers in unlocalised torrents.

6.2 Locality-biased Peer's Neighbourhood

In this subsection we apply the previously described methodology to every neighbourhood snapshot of the 50k analysed peers. Fig. 7 shows E_i vs I_n and E_c vs C_n for every considered neighbourhood snapshot. We observe that most peers have a locality-ISP biased neighbourhood ($I_n > E_i$) whereas this bias is slightly lower when we consider the country criteria. Therefore we can conclude that a significant portion of BitTorrent peers present a locality-bias neighbourhood composition. To gain more insight into this phenomenon we next quantify the reported bias. To this end, Fig. 8 presents the distribution of the median LR_I and the median LR_C of each peer across all its neighbourhood snapshots. We can observe that an important portion of peers (45%) have a surprisingly high $LR_I > 1.3$. This means that they have 30% more local neighbours from its own ISP than expected under H_0. This percentage gets reduced when looking at the locality at the country level where only 27% of peers shows $LR_C > 1.3$.

[5] We can have peers for which the torrent is unlocalizable and others for which it is not, within the same swarm.



(a) ISP (b) Country

Fig. 7. Expected number of local neighbours vs actual number of local neighbours.

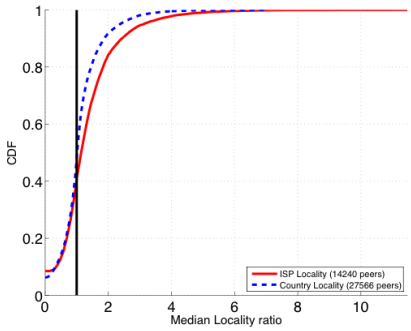


Fig. 8. CDF of locality ratio

Furthermore, it is interesting to analyse the demographics of the observed locality phenomenon in order to discover whether there are ISPs presenting a large number of peers with a high level of locality. Towards this end, for each ISP in our dataset we collect the absolute and relative number of peers having a high LR_l. We refer to these peers as *high locality* peers. Specifically, we consider a peer as a *high locality* peer if it has a $LR_l > 1.3$. Table 1 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 LR_l of the *high locality* peers for each one of these 10 ISPs. Interestingly, we observe the presence of major US and European ISP such as Comcast (US) or Virgin Media (UK) in the list. Furthermore, it is worth mentioning the presence of 4 different Indian ISPs in the list. Hence, the mentioned ISPs present specific conditions that lead to a locality biased neighborhood composition in their peers. Specifically, factors such as enforced locality policies, network congestion or the proliferation of locality-aware BitTorrent clients may be the cause of the observed results.

Table 1. ISPs with the highest number of *high-locality* peers at the overlay construction level (sorted by the number of *high-locality* peers)

ISP	Median	%
Bharti Broadband - IN	2.22	79.75
NIB (National Internet Backbone) - IN	1.77	42.40
Comcast Cable - US	1.65	36.80
PTCL Triple Play Project - PK	1.89	55.44
CHTD, Chunghwa Telecom Co., Ltd. - TW	1.89	72.19
Road Runner -US	1.63	36.27
Mahanagar Telephone Nigam Ltd. - IN	1.97	42.86

RELIANCE COMMUNICATIONS - IN	1.71	38.06
Virgin Media - UK	1.72	38.16
SBC Internet Services - US	1.74	41.34

We repeat the same analysis at the country level for peers with $LR_C > 1.3$. The results are shown in Table 2. India is the country with the highest number of *high locality* peers at the country level. The US occupies the second position in the ranking and we also observe the presence of some European countries. It is also worth noting that more than 60% of users from Taiwan are *high locality* peers. These results are consistent with the conclusion obtained at ISP level.

Table 2. Countries with the highest number of *high-locality* peers at the overlay construction level (sorted by the number of *high-locality* peers)

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

6.3 Locality-biased Peer's Stable Neighbours Set

The ultimate objective of the different policies implemented by ISPs [11] as well as researchers' proposals [12], [21] is reducing the amount of traffic crossing ISPs' transit links. Therefore, it is interesting to examine whether we observe any *locality* effect at the traffic-exchange level. For this purpose, we apply the described methodology to the stable neighbours of the 50k analysed peers. Remind that the stable neighbours are those nodes with which a peer systematically exchanges traffic with.

Fig. 9 shows box plots with the distribution of LR_I and LR_C at both the neighbourhood and the stable neighbours (i.e., exchange traffic) levels. Specifically, the boxes represent the 25, 50 and 75 percentiles of the different LR distributions. First, we observe that the set of stable neighbours shows a slightly higher locality-bias than the neighbourhood at both ISP and country levels. Second, the figure confirms that the locality effect is more marked at the ISP level than at the country level.

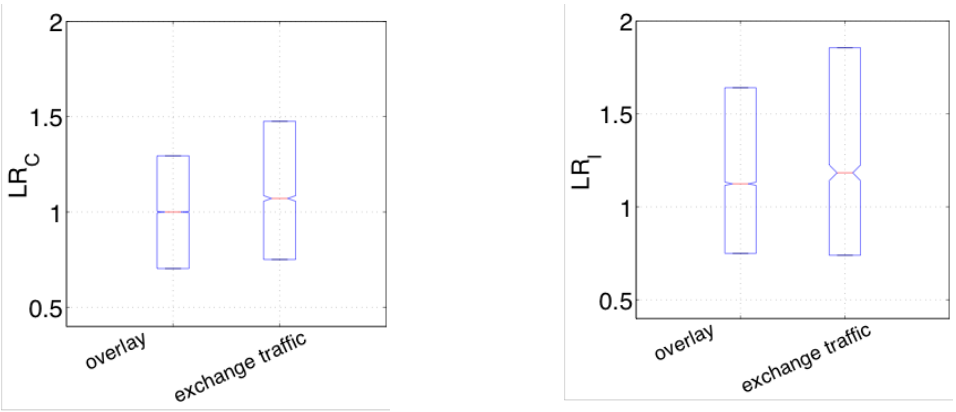


Fig. 9. Distribution of LR_I and LR_C at overlay construction and the exchange traffic levels

Finally, we have repeated the demographic analysis considering in this case the set of stable neighbours. The results are presented in Table 3 and Table 4 for ISP level and country level locality, respectively. We conclude that the observations done at the neighbourhood level are also valid at the stable neighbours level: the presence of several major ISPs from India and several major US (Comcast) and European (Telecom Italia, Telefonica Espana) providers among the top 10 ISPs with a larger number of *high locality* peers.

Table 3. ISPs with the highest number of *high-locality* peers at the traffic exchange level (sorted by the number of *high-locality* peers)

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

Table 4. Countries with the highest number of *high-locality* peers at the traffic exchange level (sorted by the number of *high-locality* peers)

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

6.4 Summary and Discussion

In this section, we have demonstrated that a significant fraction of the studied peers (45%) present at least 30% more local neighbours than those expected from a purely random neighbour selection process. Furthermore, this biased composition is even more marked when we consider the set of stable neighbours, which are those with which a peer exchange most of its traffic. The enforcement policies implemented by ISPs (e.g., throttling) along with the proliferation of successful locality-aware BitTorrent clients such as Ono [12] and other network effects such as congestion seem to be the cause for the observed results. Furthermore, our results reveal that Indian ISPs along with large American and European ISPs are those hosting a larger number of users presenting a higher locality-biased in their neighbourhood composition. Finally, our results show a higher locality-biased at the ISP than at the country level. This can be explained since it is likely that in a specific country just some ISPs implement locality-enforcing techniques, therefore when analysing the locality-biased at the individual ISPs we observe a larger bias than in the case of considering aggregately all the peers within the country.

7. Related Work

7.1 Stable Relationship of BitTorrent Clients

Few works in the literature have analysed the existence of stable connections among peers in BitTorrent swarms. First, Legout et al. [24] used a controlled environment and few torrents to analyse the interaction between peers in a swarm. The authors conclude that peers with similar speed tend to establish stable relationships among them. This observation partially supports our result across hundreds of real BitTorrent swarms in which we observe that peers tend to keep stable connections (i.e. exchange traffic) with few of its neighbours. Second, Choffness et al. [25] use traces from 10K peers to identify the existence of communities of BitTorrent users across torrents and time. Specifically, the authors reveal that BitTorrent users with similar interests tend to interact over time in multiple torrents leading to the creation of identifiable communities.

7.2 Resilience of BitTorrent Swarms

To the best of the author's knowledge the unique paper studying the resilience of BitTorrent swarms to be partitioned is [3]. The authors analyse similar scenarios to those considered in our study, the namely random node removal process and the highest-degree nodes removal process, in a controlled environment for a single torrent. Rather, we consider 400 swarms snapshots collected from 250 real BitTorrent swarms. In addition we compare the resilience of real BitTorrent swarms to that shown by equivalent random graphs.

The results for the random node removal process are similar in both studies, however, they significantly differ for the high-degree node removal process. The emulation results from [3] conclude that more than 80% of nodes must be removed in order to partition a BitTorrent

swarm whereas our experiments reveal that the number of peers to be removed is < 50% for any of the real BitTorrent swarms snapshots analysed. Therefore, results in controlled environment seem to overestimate the resilience of BitTorrent swarms.

7.3 BitTorrent Locality:

Since the seminal work by Karagiannis et al. [19] that demonstrated a substantial overlap in the torrents downloaded by users located within a campus network, substantial efforts have been dedicated to understanding and implementing BitTorrent locality solutions. Systems implementing locality solutions such as Ono [12] or P4P [21] have been proposed showing an enormous potential for locality techniques.

Some other works have performed thorough studies of the expected performance of these Locality solutions [22], [26], [27]. Typically these performance studies assume that current BitTorrent swarms present a random overlay structure. However, our analysis reveals that current BitTorrent swarms already show a locality-biased composition. Hence, the results of previous performance studies could be revisited using our conclusions.

In addition, some works [22], [23] have shown that the demographics of a torrent directly impact its inherent locality level and the theoretical capacity to localize traffic. For instance, a torrent for a local Japanese movie is expected to confine (even under a random overlay construction) most of the traffic within Japanese ISPs. Instead, a blockbuster popular movie is expected to be consumed by a large number of users across the world leading to a poor traffic locality but offering a large room for improvement using locality techniques. Furthermore, these cited works report the presence of unlocalizable torrents for a peer, i.e. torrents in which there are not other nodes from the same ISP, thus making locality impossible in practice for that peer.

However, to the best of the authors' knowledge little effort has been dedicated to characterizing the level of locality exhibited by current BitTorrent swarms. To the best of the authors knowledge just Otto et. al [28] addressed this issue by analysing the inter-country and inter-AS BitTorrent traffic using a dataset including traces for 500K users. In our study instead we analyse the locality-biased composition of 50K peers' neighbourhoods as well as their stable neighbours. Furthermore, we report those ISPs and countries in which we observe a more significant locality effect. We believe that the results in both studies are complementary.

8. Conclusions

In this paper we present a comprehensive study of the overlay topology structure and the connectivity properties at peer level of real BitTorrent swarms. For this purpose we leverage information collected from 250 real torrents and more than 150k peers. Our results demonstrate that real BitTorrent swarms are resilient to churn (i.e., random node removal process). However, real swarms are significantly less resilient to possible attacks (i.e., highest-degree node removal process) than equivalent random graphs. Furthermore, our analysis of the composition of peers' neighbourhoods reveal that current BitTorrent implementations make both leechers and seeders modify a significant portion of their neighbourhoods in short periods of time. In addition, a leecher (typically) keeps stable connections with just a handful of its neighbours with which it exchange most of its traffic. In contrast, seeders do not keep long-term connections with other peers in order to guarantee the homogeneous distribution of pieces among the participants in the swarm. Finally, our results reveal that a significant fraction of peers present a clear locality-biased composition of both

their neighbourhoods and their set of stable neighbours. This suggests that locality-enforcing policies of some ISPs, the proliferation of locality-aware BitTorrent clients and some other network effects such as congestion are localizing an important part of BitTorrent traffic within some ISPs. In particular, our results show that some large US and European ISPs along with several Indian ISPs host a large portion of peers with an important locality-biased neighbourhood composition.

These insights seem to be of interest and usefulness for: researchers and practitioners working on the improvement of BitTorrent related algorithms, companies that use BitTorrent to perform critical tasks such as software release or content replication and ISPs carrying BitTorrent traffic.

References

[1] Sandvine. Spring 2012 Global Internet Phenomena Report. Article (CrossRef Link).

[2] G. Urvoy-Keller and P. Michiardi. "Impact of inner parameters and overlay structure on the performance of bittorrent," in *Proc. of 9th IEEE Global Internet Symposium, 2006*. Article (CrossRef Link).

[3] A. Al-Hamra, A. Legout, and C. Barakat. Understanding the properties of the bittorrent overlay. *Technical Report, INRIA, Sophia Antipolis*, July 2007. Article (CrossRef Link).

[4] C. Dale, J. Liu, J. Peters, and B. Li. "Evolution and enhancement of bittorrent network topologies," in *Proc. of IEEE IWQoS'08, Enschede, The Netherlands*, June 2-4, 2008. Article(CrossRef Link).

[5] M. D. Fauzie, A. H. Thamrin, R. V. Meter, and J. Murai. "A temporal view of the topology of dynamic BitTorrent swarms," *IEEE NetSciCom, 2011, Shanghai*. Article (CrossRef Link).

[6] Linux Tracker. http://linuxtracker.org/.

[7] http://torrentfreak.com/facebook-uses-bittorrent-and-they-love-it-100625/.

[8] http://torrentfreak.com/twitter-uses-bittorrent-for-server-deployment-100210/.

[9] Amazon S3. http://aws.amazon.com/s3/.

[10] Comparison of BitTorrent Clients. http://en.wikipedia.org/wiki/Comparison of BitTorrent clients.

[11] M. Dischinger, A. Mislove, A. Haeberlen, and K. P. Gummadi. "Detecting bittorrent blocking," *ACM IMC 2008, Vouliagmeni*. Article (CrossRef Link).

[12] 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, Seattle*. Article (CrossRef Link).

[13] M. Kryczka, R. Cuevas, A. Cuevas, C. Guerrero, and A. Azcorra. "Measuring BitTorrent ecosystem: Techniques, tips and tricks," *IEEE Communications Magazine, Vol 49, 2011*. Article (CrossRef Link).

[14] C. Zhang, P. Dhungel, D. Wu, and K.W. Ross. "Unraveling the bittorrent ecosystem," *IEEE Transactions on Parallel and Distributed Systems*, 2010. Article (CrossRef Link).

[15] R. Cuevas, M. Kryczka, A. Cuevas, S. Kaune, C. Guerrero, and R. Rejaie. "Is content publishing in bittorrent altruistic or profit driven?," in *Proc. of ACM CoNEXT 2010, Philadelphia*. Article (CrossRef Link).

[16] http://www.maxmind.com.

[17] P. Erdos and A. Renyi. "On random graphs," *Publicationes Mathematicae*, 1959.

[18] P. Dhungel, X. Hei, D. Wu, and K.W. Ross. "A measurement study of attacks on bittorrent seeds," *ICC 2011 Kyoto, Japan*. Article (CrossRef Link).

[19] T. Karagiannis, P. Rodriguez, and K. Papagiannaki. "Should Internet service providers fear peer-assisted content distribution?," *ACM IMC 2005, Berkeley*. Article (CrossRef Link).

[20] R. Bindal, P. Cao, W. Chan, J. Medved, G. Suwala, T. Bates, and A. Zhang. "Improving traffic locality in BitTorrent via biased neighbor selection," *ICDCS 2006, Lisboa*. Article (CrossRef Link).

[21] H. Xie, Y. R. Yang, A. Krishnamurthy, Y. Liu, and A. Silberschatz. "P4p: Provider portal for applications," *ACM SIGCOMM 2008, Seattle*. Article (CrossRef Link).

[22] R. Cuevas, N. Laoutaris, X. Yang, G. Siganos, and P. Rodriguez. "Deep diving into bittorrent locality," *IEEE INFOCOM 2011, Shanghai*. Article (CrossRef Link).
[23] T. Hoıfeld, F. Lehrieder, D. Hock, S. Oechsner, Z. Despotovic, W. Kellerer, and M. Michel. "Characterization of bittorrent swarms and their distribution in the Internetm," *Elsevier Computer Networks*, 55(5), 2011. Article (CrossRef Link).
[24] 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. Article (CrossRef Link).
[25] D. Choffnes, J. Duch, D. Malmgren, R. Guimera`, F. Bustamante, and L.A. Nunes Amaral. "Strange bedfellows: community identification in BitTorrent," *USENIX IPTPS 2010, San Jose*. Article (CrossRef Link).
[26] M. Piatek, H. V. Madhyastha, J. P. John, A. Krishnamurthy, and T. Anderson. "Pitfalls for isp-friendly p2p design," in *Proc. of ACM HotNets-VIII, 2009, New York*. Article (CrossRef Link).
[27] D. Le Blond, A. Legout, and W. Dabbous. "Pushing BitTorrent locality to the limit," *Elsevier Computer Networks*, 55(3):541–557, February 2011. Article (CrossRef Link).
[28] J. S. Otto, M. A. Sanchez, D. R. Choffnes, F. E. Bustamante, and G. Siganos. "On blind mice and the elephant: understanding the network impact of a large distributed system," in *Proc. of ACM SIGCOMM 2011, Toronto*. Article (CrossRef Link).
[29] M. Kryczka, R. Cuevas, C. Guerrero, A. Azcorra. "Unraveling the structure of live BitTorrent Swarms: methodology and analysis ," in *Proc. of IEEE P2P 2011*, *Kyoto*. Article (CrossRef Link).
[30] Glasnost. BitTorrent traffic shaping. http://broadband.mpi-sws.org/transparency/results/
[31] Vuze. Bad ISPs. http://wiki.vuze.com/w/Bad_ISPs
[32] P. Gill, M. Arlitt, Z. Li and A. Mahanti. "The Flattening Internet Topology: Natural Evolution, Unsightly Barnacles or Contrived Collapse?," in *Proc. of PAM 2008*, Cleveland. Article (CrossRef Link).
[33] D. Wu, P. Dhungel, H. Xiaojun, C. Zhang, K.W. "Ross. Understanding Peer Exchange in BitTorrent Systems," in *Proc. of IEEE P2P 2010*, *Delft*. Article (CrossRef Link).

Ruben Cuevas obtained his MSc and PhD in Telematics Engineering at University Carlos III of Madrid (Spain) in 2007 and 2010, respectively. He is currently Assistant Professor at the Telematics Engineering Department at University Carlos III of Madrid. He has been research intern at Telefonica Research Lab and Courtesy Assistant Professor at University of Oregon in 2008 and 2012, respectively. He has been the Principal Investigator and member of the research team in several national and international research projects in the areas of social networks, content distribution and green networking. Moreover, he is co-author of more than 40 papers in prestigious international journals and conferences such as IEEE/ACM ToN , IEEE TPDS, WWW, ACM CoNEXT, IEEE Infocom, IEEE P2P or ACM COSN. His main research interests include Online Social Networks, Internet Measurements, Content Distribution and P2P Networks.

Michal Kryczka received his PhD and MSc in Telematics Engineering at University Carlos III of Madrid (Spain) in 2013 and 2009, respectively. Furthermore, he received his Master degree in Computer Science in 2008 at the Technical University of Lodz (Poland). He is currently consultant in Accenture Poland. Previously, he has been research assistant in the Institute IMDEA Networks. His main research interest includes P2P networks and Internet Measurements. He is co-author of several papers in international conferences (ACM CoNEXT, IEEE P2P) and journals (IEEE/ACM Transactions on, IEEE Communications and IEEE Letters).

Angel Cuevas received his Master in Telecommunication Engineering, his MSc and Ph.D in Telematic Engineering from the University Carlos III of Madrid in 2006, 2007 and 2011 respectively. He is currently an Assistant Professor at the department of Telematics Engineering at University Carlos III of Madrid. Previously, he has been Postdoc Researcher at the Department of Wireless Networks and Multimedia Services in the Network and Service Architecture group at Institut Telecom Sud Paris. His research is focused on the areas of social networking, peer-to-peer systems and sensor networks. Dr. Cuevas has published more than 20 papers in high quality international conferences such as ACM CoNEXT, WWW or ACM MSWiM and journals such as IEEE/ACM ToN or ACM Transactions on Sensor Networks. He was recipient of the Best Paper Award in ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems 2010 (ACM MSWiM '10).

Carmen Guerrero received her Telecommunication Engineering degree in 1994 from the Technical University of Madrid (UPM), Spain, and her Ph.D. in computer science in 1998 from the Universidade da Coruna (UDC), Spain. She is currently an Associate Professor since 2003 in the Telematics Engineering Department at the Universidad Carlos III de Madrid. She has been involved in several national and international research projects related to green networking, future Internet, content distribution and Internet measurements.

Arturo Azcorra holds a double appointment as Full Professor at the University Carlos III of Madrid in the Telematics Engineering Department and Director of IMDEA Networks. He has returned to his post as Director of IMDEA Networks in June 2012, after a period, from May 2010 to February 2012, during which he held the position of Director General at the Centre for the Development of Industrial Technology (CDTI), an agency of the Spanish Ministry of Economy and Competitiveness (MINECO), He previously held the position of Director General for Technology Transfer and Corporate Development also at the MICINN. He received his M.Sc. degree in telecommunications engineering from UPM in 1986 and his Ph.D. from the same university in 1989. In 1993 he obtained an M.B.A. from the Instituto de Empresa. Arturo Azcorra is an IEEE Senior Member and an ACM SIGCOMM Member. He has participated in and directed 49 European research and technological development projects,