Merging Parallel Swarms for BitTorrent Performance Improving and Localization

Yang Wang, Jessie Hui Wang, Donghong Qin, and Jiahai Yang
The Network Research Center, Tsinghua University
Tsinghua National Laboratory for Information Science and Technology (TNList)
Beijing, China
{*wayang84, donghong.qin* }@*gmail.com,* {*hwang, yang*}@*cernet.edu.cn*

***Abstract*—With the prevalence of file distribution systems, P2P traffic has caused great challenges to Internet service providers or network operators. The corresponding problems thereby have been a concern for industrial and academic fields. In this paper, we first present a measurement study of torrents and swarms in BitTorrent systems. We find that most swarms suffer from lack of peers and many resource files are shared by multiple trackers. Based on these two observations, we propose a new architecture, named trackers' tracker (T-Tracker) to provide a new way of traffic control for network providers, bringing a little change to current BitTorrent protocol. With T-Tracker, our architecture can provide more peers by merging parallel swarms for the performance improvement of BitTorrent system and provide more choices for biased peer selection of traffic localization schemes. Therefore, quality of BitTorrent service and network utilization can be optimized at the same time.**

Keywords-BitTorrent; merging swarms; performance improvement; P2P traffic localization.

I. INTRODUCTION

Peer-to-peer content sharing on the Internet has become one of the most popular applications in recent years. BitTorrent, almost the most successful P2P file sharing system, has been widely used for the distribution of large files. Downloading files among BitTorrent peers generates a huge amount of traffic over inter-ISP links. It is a challenge to deal with the crowded network for Internet service providers (ISPs), especially at the links between ISPs.

There are many trackers deployed by different organizations in the Internet, and they are working independently. The set of active peers maintained by a same tracker and sharing the same content is referred to as a *swarm*. In BitTorrent system, one resource file might be shared by independent multiple trackers, which means that there are several swarms in these trackers distributing the same resource file and working in parallel. Peers in these different swarms cannot find each other although they are sharing the same file. Let us define these different swarms as a group of parallel swarms, and the resource file distributed in these swarms is referred to as trans-swarm resource file.

In this paper, we first conduct a measurement study on 2258909 swarms from 41 trackers. From the measurement results, we have two observations. First, most swarms have too few peers based on the snapshot. For example, 46.9% swarms have less than 2 active peers. In these small swarms, downloader cannot connect with enough peers to finish downloading in a reasonable time. Moreover, P2P locality schemes [1] [5] cannot work well since there are no enough local peers to be selected even in some large swarms. The second observation is that, there are lots of parallel swarms and trans-swarm resource files in BitTorrent system. This phenomenon may be caused by several reasons: the content provider tries to share his file among more clients and for longer time, and he makes metadata files with different trackers in order to avoid single point failure of the trackers; tracker's operator wants to reduce its work load, and these trackers trade peer information to balance the load between each other; different content providers share the same file without knowing each other.

If one can get together peers in a group of parallel swarms, so that the group of multiple small swarms are merged into a single bigger one, then one can not only improve the file sharing performance, but also can provide more potential peers for localization, which suffers greatly from the lack of peers. Motivated by these observations, we propose a Tracker's tracker system to merge parallel swarms groups to improve availability and performance of BitTorrent and provide feasibility for P2P localization schemes. The administrator can adopt many kinds of peer recommendation in this system, as required.

The remainder of this paper is organized as follows. Related works and background information of BitTorrent protocol are introduced in Section II. We describe our measurement methodology and the data set we collect in Section III. In Section IV, we present the two important observations from our analysis of the data set. In Section V, we further quantify potential benefit of merging swarms based on two metrics we define. Based on these observations, we propose a system to implement the parallel swarms merging in Section VI. Finally, this work is summarized in Section VII.

II. BITTORRENT AND RELATED WORK

A. BitTorrent protocol

In BitTorrent system, client's downloading a shared content starts from a metadata file (with the .torrent suffix name). The metadata file contains information about the

 ISBN: 978-1-61208-186-1

resource file, including its length, name, piece length, hashing information, and the URLs of one or multiple trackers, etc.A tracker is the central component, storing and managing shared contents information, and is responsible for helping peers sharing the same resource file to connect with each other by providing a list of peers randomly. The shared resource file maintained by a tracker is called *torrent*. A set of peers sharing the same resource file with the help of a tracker is called a *swarm*. A peer uploading and downloading the shared content at the same time is called a *leecher*. When it holds the whole shared content and uploads only, the peer is called a *seeder*.

Although multiple trackers' URLs may exist in a metadata file, the BitTorrent protocol only allows a peer to associate with one tracker. So, peers sharing a trans-swarm resource file but in different parallel swarms cannot collaborate because they cannot find and communicate with each other.

B. Related studies on BitTorrent

Related studies of recent years can be classified into two categories. One is measurement to understand BT systems [2] [8] [9]. The other is to propose schemes to improve performance [1] [5].

In [2], the authors provide new finding regarding the limitations of BitTorrent systems, e.g., the existing BitTorrent system provides poor service availability, fluctuating downloading performance, and unfair services to peers. The recent measurements [8] show that 82 percent of the active swarms have no more than 10 peers. We find a similar problem on BitTorrent in our work.

Several implementations of traffic localization have been proposed, such as iPlane [5], Ono [11]. Localization requires that peers should preferentially select neighbors in multi-scales (city, AS, ISP) or choose the peers according to the network's status and the preferences regarding the application traffic [1]. Choffnes and Bustance [11] propose a method for localizing BitTorrent traffic without need for any additional infrastructure such as iTrackers [1] or cooperation between applications and ISPs. They use a plugin called Ono for a BitTorrent client which does the peer selection. The challenge is that what BT users concern is to maximize the speed of replication, while ISP wants to make the best use of the network and avoids congestion, as well as too much inter-ISP traffic. Furthermore, many solutions proposed by ISPs to control or localize P2P traffic ignore downgrade of availability caused by localization and the fact that all the torrents could not be localized.

Some recent works [4] [6] on the BitTorrent locality provide a characterization of swarms and the distribution of peers to autonomous systems (ASes). The results suggest that most ASes do not have enough potential for locality. This is a real problem faced by every localization scheme in reconciling the interest between ISPs and BT users. So it is more favorable for the applications of locality enhancements which can provide more peers and peer recommendation.

III. Measurement methodology

The BitTorrent trackers can response to two kinds of HTTP GET requests. The first kind is the most common and called as Announce request. It is used by clients to participate in the torrents. It includes the resource file's identification (info-hash, 20-byte SHA1 hash) and metrics from clients that help the tracker keep overall statistics of this peer. A resource file can be identified exactly by a 20-byte SHA1 hash (info-hash), which is calculated from the data including resource file name, file length, piece length, hash of every piece, etc. If a resource file has different names or is blocked in various ways in different torrents, the resource files will be considered as distinct resource files because they have different info-hash.

A lot of trackers support Scrape request which is the second kind of HTTP GET requests. If the Scrape request includes info-hash of a particular resource file, trackers will return statistics information of the swarm distributing this resource. Otherwise, if the scrape request is without info-hash of any files, trackers will return statistics of all swarms they host including info-hash of the file, number of downloaders (downloaded the complete file), seeders and leechers. For example, by sending HTTP GET request "http://tracker.prq.to:80/scrape", a client can get stats of all swarms in this tracker. In this paper, we develop a client using java to send Scrape requests without info-hash to collect stats of all torrents from as many trackers as we can. We conduct following steps to collect our data set for our analysis:

1. Starting from metadata files. We try to find as many metadata files as possible from websites and ftp sites, etc.

2. Extracting Trackers' URLs. Then we can extract trackers' URLs out of each metadata file. In our measurement, we find about 720 trackers' URLs.

3. Getting Scrape pages. We send Scrape requests to trackers in the trackers' URL set and get all torrents' information.

4. Identifying independent trackers. We remove the redundant trackers whose content is of high similarity. We find that many trackers' URLs seem to be totally different, but pointing to a same IP; and the swarm information of these trackers is identical. In addition, some trackers having different URLs and IP addresses maintain the same content. We remove these two kinds of redundant trackers and identify 41 independent trackers from many countries and areas, e.g., China, America, Malaysia, Holland, Sweden, Germany, France, Hong Kong, Taiwan, etc. We captured the stats of 2258909 swarms in the 41 independent trackers which support Scrape convention.

 ISBN: 978-1-61208-186-1

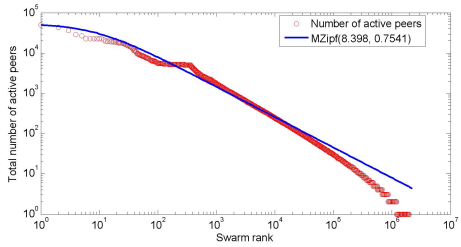

Figure 1. The number of active peers in different swarms (log-log)

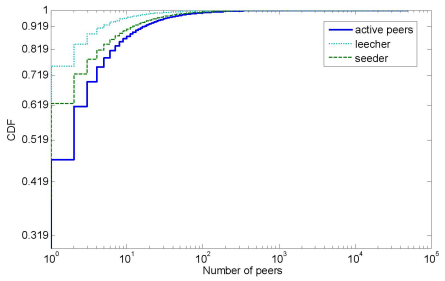

Figure 2. The CDF of the number of active peers of each swarm (log-log)

IV. Observations on Swarms

In this section, we begin to focus on the characteristic of swarms according to our measurement.

A. A Snapshot of swarms' size

Figure 1 shows the number of active peers in different swarms, ordered from the largest to smallest swarm. We can find the curve of the figure can be fitted as a Zpif-Mandelbrot distribution (q=8.398, s=0.7541). Not as shown, a little weak correlation coefficient between the number of seeders in a swarm and that of leechers is 0.6113. The number of seeders is more than that of leechers in about 51.5% swarms. Seeders and leechers have the same number in about 16.2% swarms.

Typically, when a tracker receives Announce requests for a peer list, the tracker chooses at most 50 peers randomly from all active peers in the swarm and returns this peer list to clients. A client seeks to maintain connections to a number of peers, e.g., 30-55 in the official BitTorrent client version 3. When the client maintains fewer connections, it re-contacts the tracker and tries to obtain additional peers.

Figure 2 shows the CDF of the number of active peers of each swarm. We find, unfortunately, most torrents have a small swarm according to our measurement: more than 80% swarms have less than 10 active peers; over 90% swarms have less than 10 leechers or seeders. The downloading performance of peers in these small swarms may be very poor, and it is difficult for peers in these swarms to complete resource file downloading.

B. Parallel swarms

In our measurement, we find 1409659 unique info-hash from 2258909 torrents. Let us define the number of parallel

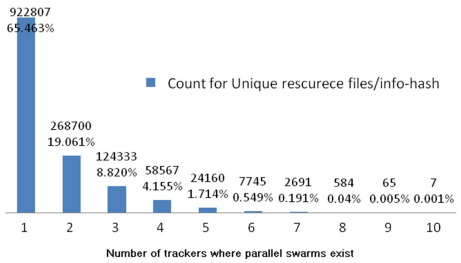

Figure 3. The number of trans-swarm resource files across different amount of trackers

swarms distributing the same resource file F_i as N_i. In Figure 3, we plot the distribution of N_i for all resource files in our data set. We can see that MAX(N_i)=10, and 65.46% of unique resource files are not trans-swarm resource files, which means they only appear at one trackers. The left 34.54% of unique resource files are shared by 40.85% torrents holding trans-swarm resource files. These resource files can benefit from merging parallel swarms potentially.

The total number of requests of all resource files on a tracker reflects how many times BitTorrent users use this tracker to download contents they need since the tracker has been built. The total number of active peers of all swarms maintained by a tracker reflects how many peers the tracker is serving at that time. The number of resource files in the tracker shows how many shared contents for which the tracker can provide service. All these three factors can evaluate a tracker's influence and scale in BitTorrent system.

In our study, we further analyze the 41 trackers and plot the number of trans-swarm resource files that each tracker hosts in Figure 4. In order to study the relationship between the total of trans-swarm resource files of a tracker and other factors, we also plot total of torrents, the number of active peers and BT download amount of all trackers. In this figure, x-axis are trackers sorted in ascending order of the number of trans-swarm resource files in each tracker, and y-axis denotes the percentage of each data item in different trackers.

The tracker size is the total of torrents in one tracker which could have many trans-swarm resource files. We can see that the tracker size follow the similar trend as the number of trans-swarm resource files in each tracker in Figure 4. The correlation coefficient between them is 0.9551. However, it is hard to establish the relationship between the number of active peers and the number of trans-swarm resource files of the tracker.

In Figure 5, all trans-swarm files are divided into 10 parts according to their N_i. We plot the average number of leechers, active peers and BitTorrent download amount of trans-swarm resource files in each part. All three curves show that resource files with higher N_i tend to be downloaded by more peers. We can conclude that the resources files that gain more popularity are more likely to appear at more swarms or trackers.

 ISBN: 978-1-61208-186-1

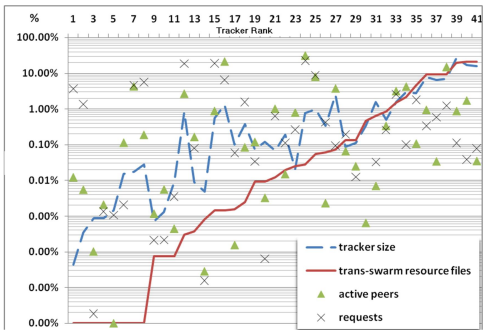

Figure 4. The percentage of trans-swarm resource files and other metrics of 41 trackers under study (semi-log)

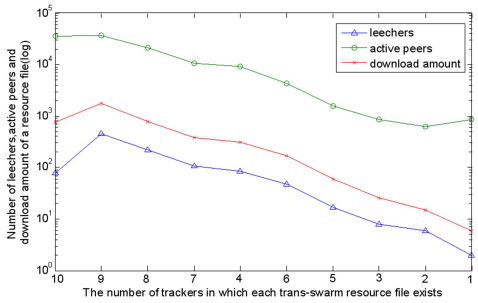

Figure 5. The average number of leechers, active peers and BT download amount of trans-swarm resource files (semi-log)

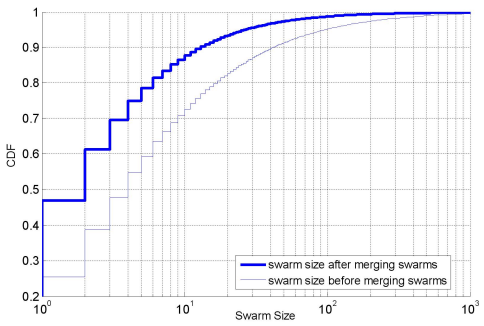

Figure 6. The CDF of swarm size before and after merging (semi-log)

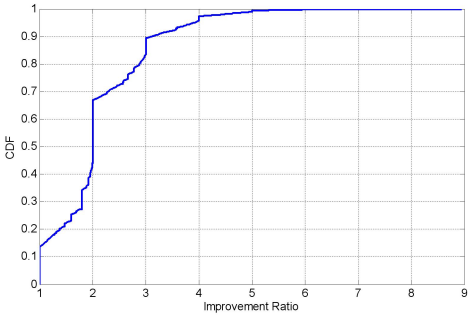

Figure 7. The CDF of improvement ratio of trans-swarm resource files

V. Potential Benefit of Merging Parallel Swarms

The number of active peers is particularly important for BitTorrent where service availability relies purely on the participation of leechers and the volunteer seeders. In general, it is hard to define whether a swarm should be considered to be small or large. But, the result of our measurement reveals there are too few peers in most swarms. From Figure 2, we can find 46.9% swarms have less than two peers. Additionally, it is important to provide more active peers not only for smaller swarms but also for larger ones, because localization in some of larger swarms is still poor in nature [4]. We have found a lot of groups of parallel swarms in our measurement, as illustrated in Figure 3. Also, it would be helpful for implementing all kinds of peer recommendation as well. In this section, we would like to study the benefit of merging parallel swarms.

Based on the snapshot of all swarms, we analyze potential benefit of merging swarms from two aspects: increase in quantity of active peers and the value of swarms.

A. More active peers

Figure 6 shows the CDF of swarm size before and after merging parallel swarms. The numbers of torrents with less active peers are expected to be significantly reduced after merging swarms. The number of swarms with no active peers has been reduced from as many as 268760 to 175125 in this snapshot. It means about 35% of resource files in the swarms with no peers before can be downloaded now. 86.62% swarms had less than 10 peers before merging, and now the percentage becomes 70.96% after merging parallel swarms.

B. Enhancing the value of swarms

As mentioned before, the number of active peers is particularly important for the performance of a swarm. The meaning of performance includes many things, such as the usability, the network efficiency [9], the stability and the accessibility. So, it is not easy to evaluate the importance of swarm size. We use Metcalfe's Law to evaluate and analyze quantitatively the value of swarms. A BitTorrent swarm is an overlay network actually. Metcalfe's Law states that the total value of a network of a number of nodes (n) is proportional to n (n-1)/2, proportional to n^2 asymptotically, if we regard all active peers and connections present the same value.

We define the value of a swarm as F(N_i)= αN_i^2. N_i is the number of active peers in swarm i, α is a constant. Then we use this model to calculate the value of merged swarms. If a trans-swarm resource file shared by a group of m parallel swarms existing in independent trackers, and the percentage of peers in the swarm of tracker i is x_i (i=1...m). So the improvement ratio is defined as $1/\sum_{i=1}^{m} x_i^2$, which is in fact the ratio of the whole swarm value to the sum of each small swarm value.

Figure 7 shows the CDF of improvement ratio of trans-swarm resource files. We observe a sharp increase in the

 ISBN: 978-1-61208-186-1

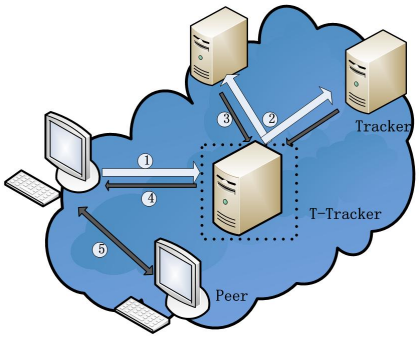

Figure 8. BitTorrent System with T-Tracker

number between 1 and 3. We see that value enhancement takes place in nearly 86% trans-swarm resource files, not in all of them. There are two reasons: 1) all the swarms in some groups of parallels warms have no peers at all. 2) Only one swarm has peers while other swarms do not. In these conditions, the improvement ratio is 1, which means there is no improvement.

VI. TRACKERS' TRACKER ARCHITECTURE

From the analysis in Section V, we can see that merging swarms would be beneficial for BitTorrent system. In this section, we will propose a new architecture to merge parallel swarms for performance improvement of BitTorrent system.

A. T-Tracker

We introduce a proxy tracker in traditional BitTorrent system, called T-Tracker (trackers' tracker) as shown in Figure 8. T-Tracker is deployed by ISPs or some third-party organizations. It is responsible for two tasks: 1) Retrieve peer lists from multiple standard trackers on behalf of clients; 2) After T-Tracker gets peer lists, it can choose which peers to be returned to clients. This is called as peer selection strategy of T-Tracker which can be very flexible, such as localization.

Obviously, both clients and ISPs can benefit from T-Tracker. For clients, there might be more active peers sharing the same resource file in multiple swarms, which can improve the downloading performance. Another advantage can be gained by clients when all the trackers specified in the metadata file are unavailable. Because T-Tracker may seek out other trackers sharing the same file; for ISPs, T-Tracker provides an approach to "tame the torrent" generated by BitTorrent, e.g., localization, which is beneficial for both ISPs and peers [1] [5].

Client software needs to be modified to exploit the benefits brought by T-Tracker. A peer can still use current client software to contact with standard trackers as before, if it doesn't want to rely on T-Tracker. But, we believe peers have incentives to use new software for the advantages mentioned above.

B. Sharing procedure

The file sharing steps in BitTorrent system with T-Tracker are listed as follows:

0. T-Tracker maintains a database of resource files, containing the data of standard trackers. It should maintain not only the information of the trackers but also the addressed and the state of peers. We will talk more about the database below.

1. When a client wants to download a resource file, the client constructs a query for a peer list to T-Tracker through Announce request. The client needs to report to the T-Tracker with standard trackers specified in metadata file when it first connects with T-Tracker. The information of trackers comes from the metadata file which the client gets.

2. T-Tracker searches its database and find which trackers the file may exists in. And then it asks standard trackers which hold the file for the active peers and it acts like an ordinary client when doing so. These trackers may be not the ones which the client reports to T-Tracker.

3. The standard trackers response to the T-Tracker with their peer lists.

4. T-Tracker collects peer lists and recommends a selected peer set according to T-Tracker's peer selection algorithm for the client. The peer set also contains which trackers these peers come from.

5. The client receives the peer set and begins to share the file with these peers.

If a client fails to maintain a fixed number of connections, it re-contacts the T-Tracker to obtain additional peers. Then T-Tracker re-contacts standard trackers to get more active peers and then forwards the selected peers to the client.

C. Implementation of T-Tracker System

On an implement level, we only need to do a little extension to BitTorrent protocol by adding a process of the client connecting with T-Tracker instead of standard trackers specified in metadata files. And T-Tracker serves BitTorrent users like a standard tracker.

In BitTorrent protocol, some information about upload and download rates, joining, completing or leaving event is sent to the tracker periodically for statistics gathering. In our system, clients need to report their state to the T-Tracker as well as standard trackers whose peers the clients are connecting with.

In step 0, we assume that T-Tracker has the knowledge of from which trackers it can get peer lists of the resource files that the clients are interested in. In fact, T-Tracker gets this knowledge in the following two ways: 1) A client needs to notify T-Tracker of the information of standard trackers from a metadata file when it begins to download a resource file. T-Tracker holds an active tracker set, and adds the new trackers to the tracker set if it can connect with these standard trackers. T-Tracker updates the resource

 ISBN: 978-1-61208-186-1

file database as well. 2) T-Tracker sends Scrape requests to the trackers at intervals in their spare time.

When T-Tracker receives a request for a content that it doesn't maintain before, T-Tracker only sends requests to the trackers which the client reports instead of all trackers in its active tracker set. This is because sending requests to all trackers will bring a great burden to standard trackers if they don't host that shared content.

For the trackers that do not support Scrape requests, T-Tracker can only find the related trackers through the first way. When T-Tracker is used by more BitTorrent users and works for a longer time, it will find more trackers and cover more parallel swarms.

D. Discussion on working load

T-Tracker hosts files on lots of trackers it can connect with. But the load of T-Tracker is not unbearable due to following reasons:

1. Bandwidth requirement of T-Tracker: the bandwidth requirements of the tracker are very low. The tracker's responsibilities are strictly limited to helping peers find each other. The network bandwidth is consumed mainly between peers, and the cost of uploading pieces of the file to downloaders is carried by peers, not by T-Tracker, due to the advantage of P2P.

2. The number of torrents and users: there would be not too many users or torrents for T-Tracker. In our measurement, the total number of torrents we find is only 2.77 times more than that of the tracker with the most torrents, and the number of active peers which need to report their stats to T-Tracker is 2.24 times more than that of the tracker connecting with the most peers.

3. T-Tracker will send Announce requests to related standard trackers, only when users ask for active peers to download a resource file. T-Tracker can cache results from different users to improve T-Tracker query performance.

4. For file sharing system, time-delay is considered to be tolerated. So tolerate delay can decrease pressure for T-Tracker.

VII. Conclusion

We performed measurement for 41 independent BitTorrent trackers, which collectively maintain state information of a combined total of over 2.2 million unique torrents. The measurement data we present in this paper shows that peers of most swarms suffer from the lack of active peers. And a lot of peers in parallel swarms can be connected with each other to improve performance.

Based on our measurement of BitTorrent System, we propose a Trackers' tracker (T-Tracker) as an extension to BitTorrent system, aiming to look for a solution to enable inter-tracker collaboration. Our scheme can enhance availability of small swarms. Furthermore, all kinds of localization/traffic improvement solutions always suffer from the lack of active peers to recommend. Our T-Tracker can draw out the potential of BitTorrent system, and provide more active peers sharing the same file by merging multiple smaller swarms into a single one. Therefore, T-Tracker provides useful means for ISPs to carry out P2P localization. It can also be used easily with other peer selection algorithms by ISPs for other traffic engineering goals.

Acknowledgment

Our work is supported by "973 Program" of China under Grant No. 2009CB320505, the National Science and Technology Supporting Plan of China under Grant No. 2008BAH37B05, the Natural Science Foundation of China (NSFC) under Grant No. 61170211, Specialized Research Fund for the Doctoral Program of Higher Education (S-RFDP) under Grant No. 20110002110056, and "863 Program" of China under Grant No. 2008AA01A303 and 2009AA01Z251.

References

[1] Haiyong Xie, Y. Richard Yang, Arvind Krishnamurthy, Yanbin Liu, and Abraham Silberschatz, *P4P: Provider Portal for Applications*, SIGCOMM Comput. Commun. Rev., vol. 38, no. 4, 2008, pp. 351-362.

[2] L. Guo, S. Chen, Z. Xiao, E. Tan, X. Ding, and X. Zhang, *Measurements, Analysis, and Modeling of BitTorrent-like Systems*, Proc. of the 5th ACM SIGCOMM conference on Internet measurement, CA, USA, 2005, pp. 35-48.

[3] M. Izal, G. Urvoy-Keller, E. W. Biersack, P. A. Felber, A. Al Hamra, and L. Garces-Erice, *Dissecting BitTorrent: Five Months in a Torrent's Lifetime*, Passive and Active Network Measurement, 2005, pp. 1-11.

[4] H. Wang, J. Liu, and K. Xu, *On the locality of BitTorrent-based video file swarming*, Proceedings of the 8th USENIX International Conference on Peer-to-Peer Systems, Boston, 21 April 2009, pp. 1-6.

[5] Harsha V. Madhyastha, Tomas Isdal, Michael Piatek, Colin Dixon, Thomas Anderson, Arvind Krishnamurthy, and Arun Venkataramani, *iPlane: An Information Plane for Distributed Services*, Proc. Symp. Operating Systems Design and Implementation, 2006, pp. 367-380.

[6] Tobias H., F. Lehrieder, D. Hock, S. Oechsner, Z. Despotovic, Kellerer W., and Michel M., *Characterization of BitTorrent Swarms and their Distribution in the Internet*, Computer Networks, 2011, pp. 1197-1215.

[7] S. Le Blond, A. Legout, and W. Dabbous, *Pushing BitTorrent locality to the limit*, Computer Networks, 2011, pp. 541-557.

[8] Chao Zhang, P. Dhungel, Di Wu, and K. W. Ross, *Unraveling the BitTorrent Ecosystem*, Parallel and Distributed Systems, IEEE Transactions, vol. 22, no. 7, 2011, pp. 1164-1177.

[9] A. Al Hamra, A. Legout, and C. Barakat, *Understanding the properties of the bittorrent overlay*, INRIA, Sophia Antipolis, July 2007, pp. 01-18.

[10] A. Bellissimo, B. N. Levine, and P. Shenoy, *Exploring the use of BitTorrent as the basis for a large trace repository*, Tech. Rep., University of Massachusetts, Amherst, 2004, pp. 04-41.

[11] D. R. Choffnes and F. E. Bustamante, *Taming the Torrent: A Practical Approach to Reducing Cross-ISP Traffic in P2P Systems*, Proc. ACM SIGCOMM, Seattle, WA, USA, Aug. 2008, pp. 363-374.

 ISBN: 978-1-61208-186-1