Enhancing Traffic Locality in BitTorrent via Shared Trackers

Haiyang Wang[1], Feng Wang[1], Jiangchuan Liu[1], and Ke Xu[2]

[1] School of Computing Science, Simon Fraser University,
British Columbia, Canada
{hwa17,fwa1,jcliu}@cs.sfu.ca
[2] Department of Computer Science and Technology, Tsinghua University,
Beijing, China
xuke@csnet1.cs.tsinghua.edu.cn

Abstract. The fast-growing traffic of peer-to-peer (P2P) applications, most notably BitTorrent, is putting unprecedented pressure to Internet Service Providers (ISPs). P2P locality has therefore been widely suggested to mitigate the costly inter-ISP traffic. In this paper, we find that even in the most popular ASes (Autonomous Systems), very few individual torrents are able to form large enough local clusters of peers, making state-of-the-art locality mechanisms for individual torrents quite inefficient.

Inspired by peers' multiple torrent behavior, we develop a novel framework that traces and recovers the available contents at peers across multiple torrents, and thus effectively amplifies the possibilities of local sharing. We address some key design issues in this framework; in particular, we discuss the detection of peer migration and further explore the trends of improving peers' incentive. We develop a smart detection mechanism with shared trackers, which achieves 45% success rate without any tracker-level communication overhead. Our trace-based simulation results indicate that our framework can successfully reduce the cross-ISP traffic and minimize the possible degradation of peers' downloading experiences.

Keywords: BitTorrent, Traffic locality, Multiple-torrent.

1 Introduction

Peer-to-peer (P2P) communications have gained tremendous popularity in the past decade. The most successful peer-to-peer file sharing application, BitTorrent (BT), enjoys phenomenal growth since its deployment in 2001, and now contributes to almost 35% of Internet's data exchanges [1]. Its exceptional scalability and robustness come from the enormous computation, storage, and communication resources collectively available at participating peers. Unfortunately, the ever-increasing traffic among the peers has also put unprecedented pressure to Internet Service Providers (ISPs). In particular, even though many BT peers interested in identical contents are located in the same or nearby Autonomous Systems (ASes), they are unnecessarily connected in the existing BT systems, thereby persistently increasing the costly cross-AS/ISP traffic.

To alleviate the cross-AS traffic, many solutions have been proposed beyond the straightforward throttling of P2P flows [2]. Among them, P2P locality [3] has been

R. Bestak et al. (Eds.): NETWORKING 2012, Part II, LNCS 7290, pp. 59–70, 2012.

widely suggested, which explores the access localities to reduce the long-haul traffic. Yet, so far the distribution of BT peers has seldom been examined in the global Internet [4]. As such, the potential benefit and even the applicability of the locality mechanisms in the real world remain unclear.

In this paper, we examine the existence and distribution of peer locality through a large-scale measurement. Our measurement lasts three months, collecting information from more than $800,000$ peers. The results demonstrate that the BitTorrent peers do exhibit strong geographical locality that could be explored. Unfortunately, if we focus only on individual torrents, very few torrents are able to form large enough local cluster of peers. Even for the most popular ASes, this ratio is less than 5%, which makes state-of-the-art locality mechanisms for individual torrents quite inefficient.

Recent measurements, on the other hand, suggest that over 85% of the peers indeed participate in multiple torrents [5], which is also validated by our data. Inspired by this, we develop a novel framework that traces and recovers the available contents at peers across multiple torrents, thus effectively promoting the locality. We address the key design issues in this framework, particularly, the detection of peer migration. We demonstrate that the detection does not necessarily involve complex and costly tracker-level cooperations. Instead, a clever use of shared trackers can successfully detect around 45% of the peer migrations without extra communication overhead.

The performance of our locality mechanism across multiple torrents has been evaluated through extensive trace-driven simulations with various detection rates. Compared to state-of-the-art locality mechanisms for individual torrents, our solution improves the local content availability, thus significantly reducing cross-AS traffic. In addition, it brings minimal impact to the peer downloading experiences.

The rest of this paper is organized as follows. In Section 2, we illustrate the related works. We then present our measurement results in Section 3, which reveal the challenges to the design and implementation of P2P locality. In Sections 4 and 5, we explore the P2P locality across multiple torrents, and present an effective detection mechanism for peer migration. Finally, after the trace-driven evaluation in Section 6, we conclude the paper and offer some future directions in Section 7.

2 Related Works

There have been numerous studies on the implementation, analysis, and optimization of the BitTorrent system [6]. P2P locality has recently attracted particular attention following the pioneering work of Karagiannis et al. [3]. Based on real traces and simulated torrents, they proposed the concept of locality in peer-to-peer systems and evaluated its benefit. Blond et al. [7] showed through a controlled environment that high locality values (defined by [3]) yield up to two orders of magnitude savings on cross-AS traffic, without any significant impact to the peers' download completion time. Xie et al. [8] further suggested cooperation between peer-to-peer applications and ISPs by a new locality architecture, namely, P4P, which can reduce both the external traffic and the average downloading time. Choffnes et al. [9] proposed Ono, a BitTorrent extension that leverages a CDN (Content distribution network) infrastructure, which effectively locates peers that are close to each other. Bindal et al. [10] also examined a novel approach to enhance BitTorrent traffic locality, namely, *biased neighbor selection*. Using

this method, a peer chooses the majority, but not all, of its neighbors from peers within the same ISP.

Guo et al. [5] revealed that more than 85% of all peers participate in multiple torrents and noted the peer migration behavior. This *migration behavior* indicates that some BT peers have the potential to serve others even when they have already left the swarm. They proposed an inter-torrent approach through tracker-level collaborations. The main idea is to build a tracker site overlay for tracker-level collaboration; the peers migrating between different torrents can then be detected and recovered as potential seeders for the torrents. Dan et al. [11] further investigated how the separated torrents can be merged together to improve the performance of an entire torrent. The measurement from Piatek et al. [12] however found that about 91% of peers in any single swarm do not arise in any other swarms. This observation seems to contradict the study in [5]; yet this is mainly due to the difference of their objective as well as their measurement schemes. On the other hand, the measurement study by Neglia et al. [13] investigated the availability of BitTorrent system among different tracker configurations. The popularity and the performance of the multi-tracker configuration [14] was discussed. Their study showed that around 35% of the torrents enable multi-tracker configurations. Pouwelse et al. [15] further discussed the relationship between BT trackers and torrents, and examined the tracker availability across multiple websites, albeit with individual torrents.

It is worth noting that, the studies of content bundling [16] also provide useful insights to understand multiple torrents behavior. A pioneering work from Menasche et al. [16] studied the content unavailability problem in the BitTorrent system. This study for the first time proposed a model to analyze the availability and the performance implications of bundling through an extensive measurement. Follow up studies such as [17] also studied some other aspects for content bundling in BitTorrent systems.

Our work was motivated by these studies; yet we explore the multi-tracker configuration across multiple torrents simultaneously, providing a seamless and light-weight solution to locality in the real BitTorrent system.

3 Pitfalls of BitTorrent Traffic Locality

We extracted a large collection of real torrents as advertised by www.btmon.com, one of the most popular torrent sites. We developed a script to automatically detect the 'href' field in each given HTML file and downloaded the metainfo files ending with '.torrent', which resulted in $74,732$ metainfo files. Within our data set, there are 316 bad metainfo files, $1,027$ unavailable torrents due to tracker failures, and $3,340$ torrents having only 1 peer. We excluded these abnormal torrents, and, to balance accuracy and measurement overhead, randomly selected $8,893$ out of the $70,049$ normal torrents for our study.

We then ran a modified version of CTorrent (a typical BitTorrent client in FreeBSD) [18] on the PlanetLab nodes. Different from conventional pure PlanetLab experiments in which the clients communicate with others within the PlanetLab only, our modified CTorrent clients actively joined existing torrents in the global Internet and recorded the observable peer information from the trackers and from other peers over time. As such, the small set of controlled PlanetLab nodes were able to capture the information of most peers in the torrents, in particular, their IP addresses. With a maximum of 50

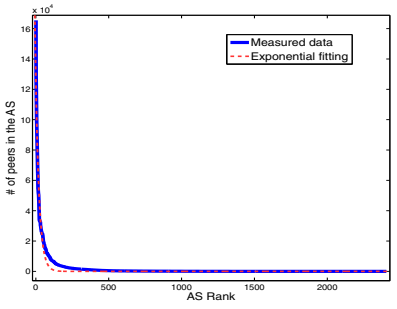

Fig. 1. Total peer popularity of 2864 ASes

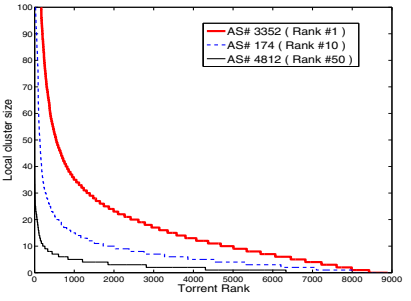

Fig. 2. Distribution of local clusters

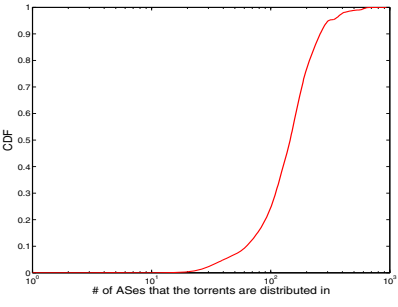

Fig. 3. Peer distribution of the torrents

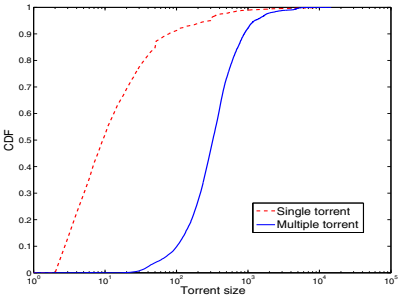

Fig. 4. Single torrent vs. multiple torrents

initial peers from the trackers, we successfully detected the IP addresses of over 95% peers for most of the torrents.[1]

Given the IP addresses of the peers, we extracted their corresponding ASes through the 'whois' command in Linux. This resulted in $2,405$ distinct ASes, and Figure 1 shows the peer popularity across all torrents in these ASes. We can see that it can roughly be fitted by an exponential distribution ($y = a^{bx}$, where $a = 1.261 \times 10^5$, $b = -0.0480$); in other words, despite the common belief that BitTorrent is extremely popular everywhere, a majority of the ASes indeed do not host a noticeable number of BitTorrent peers, e.g., 65% of them have less than 100 peers across all torrents.

Since the existing locality mechanisms have focused on individual torrents only, it is important to further investigate the distribution of local clusters, where a local cluster is the collection of local peers downloading the same content in an AS. Unfortunately, as shown in Figure 2, even for the very popular ASes, only a few torrents are able to form large local clusters. As an example, in the most popular AS ($AS3352$), most of the torrents (over 95%) have less than 50 peers, even though these torrents are of quite large client populations (generally more than 500 peers). A close look reveals that the peers of most torrents are distributed in more than 150 ASes (the big picture of this distribution is shown in Figure 3), thus unavoidably involving extensive cross-AS communications.

[1] This ratio is calculated by comparing the number of detected peers with the total number of peers as advertised by the tracker of a torrent.

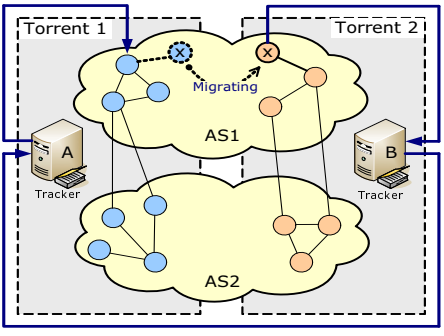

Fig. 5. Multiple torrent based P2P locality

Such results suggest that a locality mechanism designed exclusively for individual peers can be ineffective for many of the torrents. In addition, since it only works with local peers that simultaneously participate in the same torrent; once a peer leaves the torrent, its downloaded contents will become invisible immediately. Fortunately, recent studies have revealed that over 85% of the peers indeed remain in the BT system, participating in other torrents after their departure [5]. Assume that the trackers can keep tracking those peers remaining in the system, the available local peers for most torrents could be increased significantly. Figure 4 validates the potentials of this locality approach across multiple torrents, where the peer population of most torrents (more than 85%) is tripled after 10 hours.

4 P2P Locality across Multiple Torrent: An Overview

We now proceed with a framework design for exploring P2P locality across multiple torrents. We particularly focus on the *tracker-and-client-based* solutions [10], which rely only on modifications to end-system implementations. These locality solutions typically replaces the random peer selection by an *AS hop count*-based metric. Upon a request, the modified tracker sorts all other peers in the torrent in ascending order of their AS hop count to the requesting peer, and then sends the prefix of this sorted list (e.g., first 50 peers) to the requesting peer. The requesting peer would then choose the majority, but not all, of its neighbors from peers within the same ISP. Typically, 35 peers within the same ISP (AS hop count 0) can be returned together with 15 other random peers [10].

For the individual torrent scenario, many neighbor selection approaches have been proposed [10] [19], which could also be applied in the multiple torrent scenario. The new challenge, however, is the detection of peer migrations among torrents. That is, if a peer has finished downloading in a torrent (say torrent 1) and left, but remains in other torrents,[2] how can we detect it, so as to recover the previously downloaded content to

[2] For ease of exposition, we will focus on the scenario that the migrating peer remains in only one another torrent. Our solution however can be easily extended to the scenario that the peer remains in more than one torrent.

facilitate the locality for the remaining peers in torrent 1? This is illustrated in Figure 5, where peer x leaves torrent 1, but remains in torrent 2. If this migration can be detected, peer x can still serve as a potential seeder for torrent 1, which will greatly promote the locality for the peers in AS1.

It is easy to see that there are two challenges in this design: First, this solution may need a tracker overlay for tracker-to-peer and tracker-to-tracker communications; in particular, adding extra collaboration among the trackers to trace the migration of peer x [5]. Unfortunately, besides the overheads, enforcing communications between the public trackers can be quite difficult. Second, as shown in Figure 5, since peer x is no longer interested in torrent 1, it is hard to guarantee that this peer will have enough incentive and online time to seed the content again for others. To address these challenges, we will explore the detecting of peer migration as well as the incentive issues in the remaining sections.

5 Detecting Peer Migration with Shared Trackers

In this section, we will consider the migration detection with shared trackers. Assume torrent 1 and torrent 2 are both managed by tracker A; any peer migrating between these two torrents can simply be detected by tracker A without communication to other trackers. While this seems to be an ideal case, we now show that it indeed exists and is not uncommon.

Our observation starts from the fact that the latest BitTorrent metainfo file can include multiple tracker sites stored in the *announce-list* section [14]. This multi-tracker configuration allows peers to connect to more than one tracker at the same time, which brings two tangible benefits: (1) better accommodates tracker failures; and (2) balances load among the trackers. Figure 6 offers an example with the multi-tracker configuration, where torrent 1 is managed by both trackers A and B, and torrent 2 is managed both by tracker B and C. In this case, if there is a BT peer x migrating from torrent 1 to torrent 2, tracker B will receive the arrival message of peer x twice with different content identifications (one arrival message for each torrent). Therefore, tracker B can actually be aware of any peer migration between torrent 1 and torrent 2 without any tracker-level collaboration.

The question now becomes (1) how popular is the multi-tracker configuration in the real world? and (2) how many migrations can be detected by this configuration in practice? To answer the first question, we consider all the 1192 trackers in our measurement. We record the announce-list of the torrents in our dataset, and show the cumulative distribution of the trackers that have been used in Figure 7. It indicates that more than 90% torrents have specified at least two trackers, and a few torrents even have announce-lists of multi-hundred trackers. This is much higher than an earlier measurement in 2007 [13] (observed multi-trackers in 35% of the torrents), and thus suggests the multi-tracker configuration has been quickly recognized and deployed in the BitTorrent community.

To answer the second question, we model the relationships among different torrents as two $n \times n$ matrixes, M_1 and M_2, where n is the number of torrents in the whole system. Each component of M_1, $M_1^{i,j}$ is of a binary value, indicating whether torrents i and j have at least one common tracker (1-Yes, 0-No); similarly, each component of M_2, $M_2^{i,j}$ indicates whether torrents i and j share at least one migrating peer.

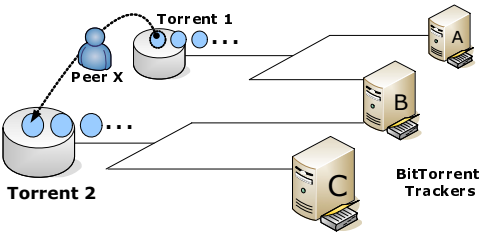

Fig. 6. Peer migration in the shared tracker environment

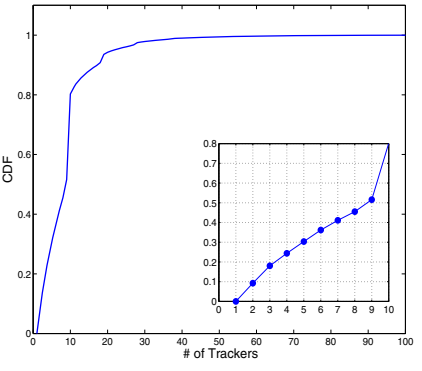

Fig. 7. # of trackers used by torrents

It is easy to verify that a dot product between these two matrixes, $M_3 = M_1 \cdot M_2$, gives the detectable migrations by the shared tracker approach. Specifically, $M_3^{i,j} = 0$ indicates that peer migrations between torrent i and j are either undetectable or do not exist at all; otherwise, the migrations between these two torrents will be detected even when $M_2(i, j) > 1$. In our measured data, matrix M3 has 2538 non-zero entities, where M2 has 5707. Therefore, the peer migrations among about 45% torrents can be detected with shared trackers.

Once detected, the shared tracker can then use the biased neighbor selection [10] to improve the P2P locality. It may also forward the migration information to other trackers; however, this collaboration is not compulsory in our framework.

6 Performance Evaluation

We now evaluate the performance of our framework design in the multiple torrent environment. We also compare it with other state-of-the-art locality solutions; in particular, the biased neighbor selection for individual torrents [10][19]. To achieve a fair comparison and also to examine the diverse factors that would affect their performance, we also use the discrete-event BitTorrent simulator developed by Stanford University [20] as [10] did; we summarize the key network settings as follows (more configuration details can be found in [10]):

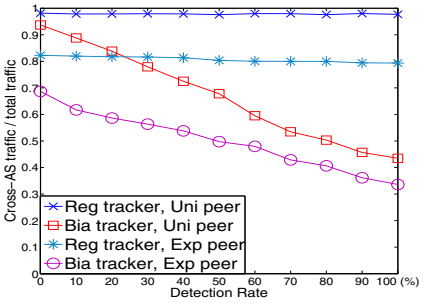

Fig. 8. Percentage of cross-AS traffic (a small torrent with 100 initial peers)

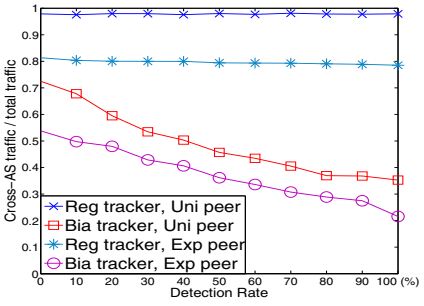

Fig. 9. Percentage of cross-AS traffic (a large torrent with 600 initial peers)

All peers inside the ISPs are modeled after cable modem and DSL nodes, and have asymmetric upload/download bandwidth. The upload bandwidth of these peers is 100kbps and downloading bandwidth is 1Mbps. Considering the peer arrival/departure, most peers are joining the network at once, i.e. the flash crowd scenario. We focus on this feature since it is the most challenging for ISPs to handle. For each torrent, there is one original seeder that will always stay online (with 400Kbps uplink bandwidth), and other peers (except for the migrating peers) will leave the BT network forever as soon as they finish downloading. This is in accordance with the measurements because only 85% peers are participating in multiple torrents.

For the multiple torrent scenario, we assume that 1000 peer migrations occur during a 48-hour simulation, which is consistent with the data in Figure 4. We then evaluate the locality performance with different peer distributions and migration detection rate. The downloaded content of detected peers will be recovered for locality. These extra peers however will not simply serve as selfless seeders, but rather normal peers that expect data, albeit from other related torrents through a cross-torrent credit approach [21]. This will eliminate biases related to seeding incentives, which has been discussed in section 6.

We will focus on two metrics: cross-AS traffic and downloading completion time of peers, which reflect the potential benefit and impact of P2P locality, respectively.

We first calculate the percentage of the cross-AS traffic over the total downloading/uploading traffic of the peers in different torrents in the multiple torrent environment. Figures 8 and 9 show the results of two typical torrents. The first is a relatively small torrent with 100 initial peers, and the second is a large torrent with 600 initial peers. With regular unmodified trackers, we can see that the cross-AS traffic is quite high (over 95%) when the peers are uniformly distributed among the ASes; for the exponential peer distribution, the cross-ASes traffic is relatively lower, implying that certain peer localities have been naturally utilized. Even though this exponential peer distribution is more realistic as validated in our earlier measurement (Fig. 1 and 2), the regular trackers do not take full advantage of the localities, and hence the cross-AS traffic remains high (around 80%). On the other hand, the biased tracker design prioritizes

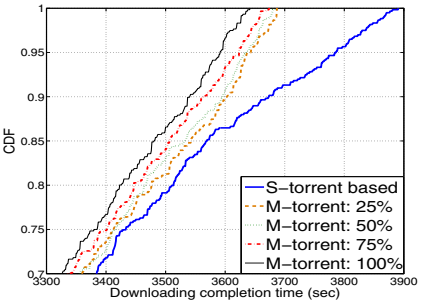

Fig. 10. Downloading completion time of 600 peers (uniform distribution with regular tracker)

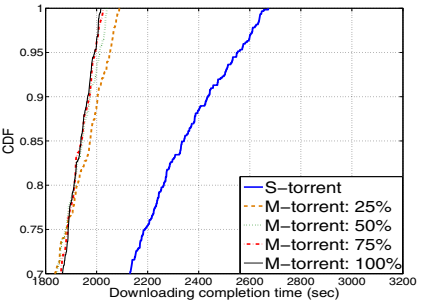

Fig. 11. Downloading completion time of 600 peers (uniform distribution with biased tracker)

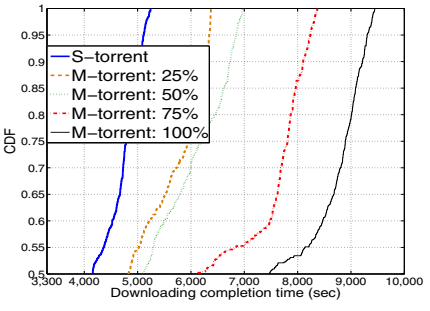

Fig. 12. Downloading completion time of 600 peers (exponential distribution with regular tracker)

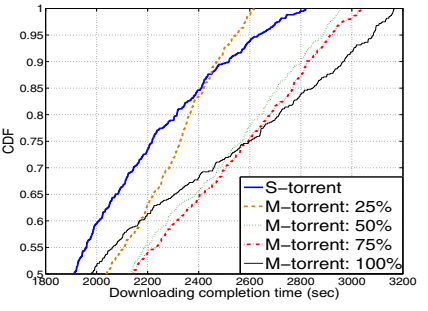

Fig. 13. Downloading completion time of 600 peers (exponential distribution with biased tracker)

local peers for sharing, which, as shown in Figures 8 and 9, significantly reduces the cross-AS traffic. This is particularly true for larger torrents that enable more opportunities for local connections.

Note that, when the detection rate is 0, the multiple torrent setting degenerates to a single torrent setting with no previously downloaded content being recovered from migrating peers. In this case, the cross-AS traffic is the highest in the figures. With biased trackers, the percentage of cross-AS traffic is also decreasing with the increase of migration detection rate. This suggests that the combination of locality and multiple torrent is quite effective in reducing cross-AS traffic. Recall that, for detection with shared tracker only, we have a success rate of 45% (see Section V), which translates into percentages of cross-AS traffic of roughly 50% and 35% for the 100-peer and 600-peer torrents, respectively. Even for uniform peer distribution, the traffic reduction is still remarkable, suggesting the necessity for exploring locality.

We next examine whether the re-shaping of the traffic will affect user experience; in particular, whether it will slow down the peer completion time. Figures 10 to 13 present the cumulative distribution of the downloading completion time with different

peer distributions and torrent-tracker combinations. In the figures, we use *M-torrent* and *S-torrent* to represent the multiple-torrent-based and single-torrent-based solutions, respectively (the percentage values refer to the possible detection rate of peers' migration behavior). We show the results of the larger torrent with 600 peers, and we have observed similar curves for torrents of other sizes.

We first look at the case of peers uniformly distributed among ASes, as shown in Figures 10. Surprisingly, although no extra peers will serve as selfless seeders, the downloading completion time of the peers is still improved by the multiple torrent approach. Moreover, as shown in Figures 11, all peers will finish their downloading within $2700sec$ in the individual locality torrent; this completion time will be further improved to $2100sec$ with the proposed multiple torrent based locality. Note that the peers are assumed to be uniformly distributed among different ASes, all ASes therefore have enough local resources to utilize. Intuitively, potential benefits can be obtained by accessing these local peers.

However, for the exponential peer distribution (a more realistic case yet seldom been discussed in the previous studies), the downloading completion times of most peers are increased as shown in Figures 12. In particular, if the peers are connected to regular trackers, the multiple torrent based approach will slow down the downloading completion time of all peers significantly. The peers' downloading completion time is almost doubled when the detection rate reaches to 100%. This result shows that the exponential peer distribution across the ASes will potentially reduce peers' downloading experience with an increase of torrents' population. An intuitive explanation is that the flash crowd of peers as well as the trackers' random peer selection will put more pressure to the cross-ISP links and unavailable cause link overload (especially for the most popular ASes). Moreover, we have also observed that a great number of peers in the most popular ASes have very close downloading completion time (also leave the BT networks at similar time). Their departure will also reduce downloading performance of other peers in the BitTorrent system. Fortunately, as shown in Figure 13, the biased trackers can well address such a problem and peers' completion times only slightly increase.

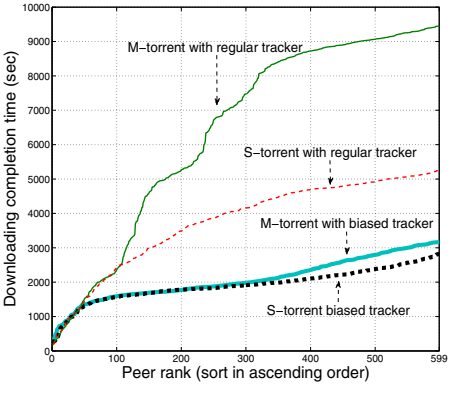

Fig. 14. Comparison of downloading completion time

Note that, to clarify the possible degradation of the downloading performance, we have ignored the first quartile (25th percentile) of the CDF where the lines are too close to each other.

For easy comparison, we also show the completion times of the four typical torrent-tracker combinations in Figure 14, where the peers are sorted in ascending order of their downloading completion time (the detection rate of M-torrent is set to 100%). It clearly shows that the combination of locality and multiple torrent will minimize the impact to the peer downloading experiences.

7 Conclusions

In this paper, we proposed a novel framework that traces and extracts the available contents at peers across multiple torrents, thus effectively improving the locality. A series of key design issues are addressed in this framework; in particular, we discussed the detection of peer migration and further explored the trends of sharing incentive. We developed a smart detection mechanism with shared trackers, which incurs no extra communication overhead. The performance of our framework was evaluated through extensive trace-driven simulations. Compared to the locality for individual torrents, our solution has successfully promoted the local content availability, thus significantly reducing cross-AS traffic and yet keeping minimal impact to peers' downloading experiences.

Acknowledgement. This research was supported by a Canadian NSERC Discovery Grant, a Discovery Accelerator Supplements Award, an NSERC Engage Grant, China NSFC projects (61170292, 60970104), and a China NSFC Major Program of International Cooperation Grant (61120106008).

References

1. CacheLogic, `http://www.cachelogic.com/`
2. Dischinger, M., Mislove, A., Haeberlen, A., Gummadi, K.P.: Detect Bittorrent Blocking. In: Proc. ACM/USENIX IMC (2008)
3. Karagiannis, T., Rodriguez, P., Papagiannaki, K.: Should Internet Service Providers Fear Peer-Assisted Content Distribution? In: Proc. ACM/USENIX IMC (2005)
4. Otto, J.S., Sanchez, M.A., Choffnes, D.R., Bustamante, F.E., Siganos, G.: On Blind Mice and the Elephant: Understanding the Network Impact of a Large Distributed System. In: Proc. ACM SIGCOMM (2011)
5. Guo, L., Chen, S., Xiao, Z., Tan, E., Ding, X., Zhang, X.: Measurements, Analysis, and Modeling of BitTorrent-like Systems. In: Proc. ACM/USENIX IMC (2005)
6. Qiu, D., Srikant, R.: Modeling and Performance Analysis of Bit Torrent-Like Peer-to-Peer Networks. In: Proc. ACM SIGCOMM (2004)
7. Blond, S.L., Legout, A., Dabbous, W.: Pushing BitTorrent Locality to the Limit. INRIA Tech, Rep. (2008)
8. Xie, H., Yang, R.Y., Krishnamurthy, A., Liu, Y.G., Silberschatz, A.: P4p: Provider Portal for Applications. In: Proc. ACM SIGCOMM (2008)

9. Choffnes, D.R., Bustamante, F.E.: Taming the torrent: a practical approach to reducing cross-isp traffic in peer-to-peer systems. In: Proc. ACM SIGCOMM (2008)
10. Bindal, R., Cao, P., Chan, W., Medved, J., Suwala, G., Bates, T., Zhang, A.: Improving Traffic Locality in BitTorrent via Biased Neighbor Selection. In: Proc. IEEE ICDCS (2006)
11. Dan, G., Carlsson, N.: Dynamic Swarm Management for Improved BitTorrent Performance. In: Proc. USENIX IPTPS (2009)
12. Piatek, M., Isdal, T., Krishnamurth, A., Anderson, T.: One hop reputations for peer to peer file sharing workloads. In: Proc. NSDI (2008)
13. Neglia, G., Reina, G., Zhang, H., Towsley, D., Venkataramani, A., Danaher, J.: Availability in BitTorrent Systems. In: Proc. IEEE INFOCOM (2007)
14. BitTorrent Multi-tracker Specification, `http://www.bittornado.com/docs/multitracker-spec.txt`
15. Pouwelse, J.A., Garbacki, P., Epema, D.H.J., Sips, H.J.: The Bittorrent P2P File-sharing System: Measurements and Analysis. In: Proc. USENIX IPTPS (2005)
16. Menasche, D.S., Rocha, A.A.A., Li, B., Towsley, D., Venkataramani, A.: Content Availability and Bundling in Swarming Systems. In: Proc. CoNext (2009)
17. Lev-tov, N., Carlsson, N., Li, Z., Williamson, C., Zhang, S.: Dynamic File-selection Policies for Bundling in BitTorrent-like Systems. In: Proc. IEEE IWQOS (2010)
18. Ctorrent, `http://ctorrent.sourceforge.net/`
19. Liu, B., Cui, Y., Lu, Y., Xue, Y.: Locality-Awareness in BitTorrent-Like P2P Applications. Proceedings of the IEEE Transactions on Multimedia 11(3), 361–371 (2009)
20. BT-SIM, `http://theory.stanford.edu/~cao/btsim-code.tgz`
21. Yang, Y., Chow, A.L.H., Golubchik, L.: Multi-Torrent: a Performance Study. In: Proc. IEEE/ACM MASCOTS (2008)