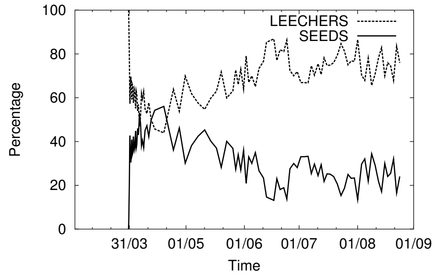
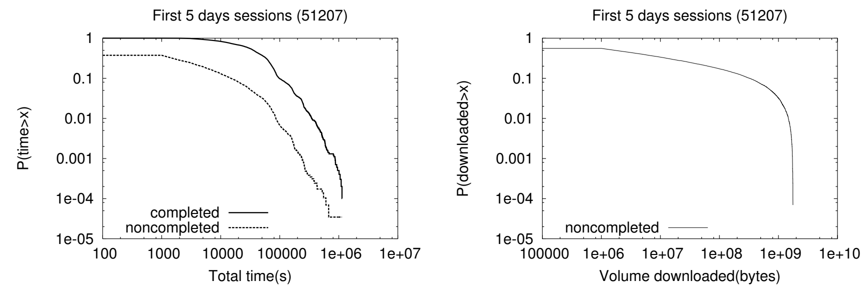

LEECHERS
SEEDS
100
80
60
40
20
0
Percentage
31/03
01/05
01/06
01/07
01/08
01/09
Time

how many peers eventually completed the download and the fraction of bytes that the peers that did no complete the transfer downloaded. Since BitTorrent allows users to suspend and resume the download at any time, a peer might complete the download after one or several sessions. In the tracker log, a session id identifies the session (hash of the IP address of the host and its current time), along with the IP address of the host as seen by the tracker. Thus, identifying single session downloads is easy based on the session id. To reconstruct multi-sessions downloads, we assumed that the IP address of the host does not change from one session to another. NATs often used port numbers to disambiguate connexions and not IP addresses, so our assumption can hold even in this case. Of course, if two or more hosts behind a firewall simultaneously participate to the same torrent, it might be difficult to disambiguate them. In addition, since a peer provides the amount of bytes it has downloaded in each of its reports (sent every 30 minutes), we mated two sessions with the same IP if the amount of bytes of the last report for the first session is close to the amount of bytes of the first report of the second session. These two values are not necessarily the same if for instance, the user disconnected the BitTorrent client improperly which results in the latter not sending its disconnection report with its current amount of bytes downloaded. A set of sessions form a multi-sessions download if the download of the file is completed during the last session (on average, we found that a multi-session download consists of 3.9 sessions). We also observed some sessions that started with already 100% of the file downloaded. This is the case if the user is kind enough to rejoin the torrent after the completion of the download to act as a seed. We thus categorize the sessions into four categories : single session download, multi-session download, seed sessions and session that never complete (incomplete downloads). We provide statistics for these sessions in Table 1.

Type	Number of sessions	Total down. (TB)	Total up. (TB)	Down. /session (MB)	Up. /session (MB)	Down. rate (kb/s)	Up. rate (kb/s)	Down. Duration (hours)
Single session downloads	20584	34.7	28.5	1765.7	1450.2	1331.1	325. 2	8.1
Multi-session downloads	14285	6.2	4.3	455.2	319.2	390.6	145.0	19.2
Incomplete downloads	138423	11.2	9.1	84.5	69.1	114.7	50.5	-
Seed sessions	8555	-	12.7	-	1556.6	-	223.6	-
Total	181847	52.1	54.6	-	-	-	-	-

Table 1. Sessions types and their characteristics

Overall, Table 1 indicates that only 19% of the sessions are part of a transfer that eventually completed. However, the 81% of sessions that did not complete the transfer represent only 21.5% of the total amount of downloaded bytes. Moreover, the incomplete sessions uploaded almost as much bytes as they downloaded. It is difficult to assess the reason behind the abortion of a transfer. It might be because the user eventually decides he is not interested in the file or because of a poor download performance. To investigate this issue, we look at the statistics (durations and volumes) for sessions that start with 0% of the file during the first five days. We classify these sessions into two categories denoted as "completed" and "non-completed". The completed category corresponds to the single session downloads defined previously while the non-completed category corresponds to transfers that never complete or the first session of transfers that will eventually complete. We present on Figure 4 the complementary cumulative distribution functions for the download volumes and session durations for the completed and non-completed sessions. We can observe from this figure that 90% of the non-completed sessions remain less than $10,000$ seconds in the torrent (and 60% less than 1000 seconds) and retrieve less than 10% of the file. Most of the non-completed sessions are thus short and little data is downloaded.

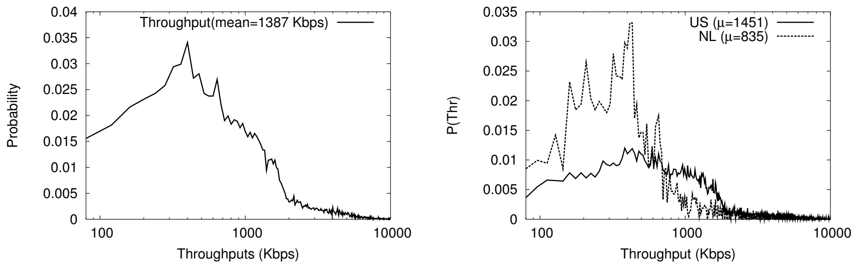


(a) Duration

(b) Download volume

Fig. 4. Single sessions statistics for sessions seen during the first 5 days

Let us now further focus on the performance achieved by single session downloads. The average download rate of these $20,584$ sessions is close to 1.3Mb/s over the whole trace, which is larger than the download rate averaged over all sessions (500kb/s). The average download time for single session download is about $29,000$ seconds overall. Such values reveal a high variability in the download rates achieved by these sessions since if they had all achieved a download rate of 1.3Mb/s, the average download time would be $\frac{1.77\text{GB}}{1.3\text{Mb/s}} \sim 10,000$ seconds. This is confirmed by the distribution of these download throughputs (see Figure 4.2) that clearly exhibits a peak around 400kb/s (a typical value for ADSL lines), a value significantly smaller than the mean.



(a) Global throughput distribution

(b) Throughput distribution per cluster

Fig. 5. Throughputs for single session downloads

4.3 Geographical Analysis

The tracker log provides the IP addresses of the clients. Based on these addresses, we have used NetGeo (`http://www.caida.org/tools/utilities/netgeo/`) to obtain an estimation of the origin country of the peers that participated to the torrent. The estimation might be imprecise as NetGeo is not maintained any more. Overall, we were not able to obtain information for around 10% of the hosts. In Table 2 we indicate the top five countries for the first 5 days, the first 4 weeks and the complete trace. We can observe that this set of countries, as well as the ranking is consistently the same for all three time scales. Most of the clients are US clients while Europe is represented only through the Netherlands. Still, for Netherlands, 50% of the hosts originate from ripe.net (an information also provided by NetGeo), which acts as the Regional Internet Registry for the whole of Europe, Middle East, Africa and part of Asia also. Thus we can guess that most of these peers are spread all over Europe.

Countries	First week	First four weeks	Complete trace
United States	44.6%	44.3%	32%
Netherlands (Europe)	14.9%	15.3%	23.9%
Australia	12.7%	12.6%	17.8%
Canada	5.7%	5.8%	4.9%
% of total hosts	77.9%	78%	78.6%

Table 2. Geographical origins of peers

We next investigate the relative performance of the four clusters identified above (US, NL, AU and CN). To do so, we consider again the $20,584$ peers that complete the download in a single session. 17,000 peers out of the 20,584 initial peers, are in the four clusters (11498 in US cluster, 2114 in NL, 1995 in AU and 1491 in CM). We plot in Figure 4.2 the distribution of download throughputs achieved by the peers in the NL and US clusters (the AU and CN clusters are highly similar to the US cluster and not depicted for sake of clarity). Figure 4.2 reveals that the download throughput of the hosts in the NL cluster is significantly smaller than the throughput of the hosts in the US cluster (where more mass is on the right side of the curve). This can indicate that clients in the US have, in general, better access links than in Europe.

5 Client Log Analysis

To better observe the individual behavior of a BitTorrent peer, we have run an instrumented client behind a 10Mb/s campus network access link. Our client joined the torrent approximatively in the middle of the 5 months period (i.e., far after the initial flash crowd). We experienced a transfer time of approximately 4,500 seconds (i.e., much lower than the average download times previously mentioned) and our client remained connected as a seed for 13 hours. Our client logged detailed information about the upload or download of each individual chunk. In Figure 6, we represent the number of peers with whom our client is trading. At time 0, our client knows around 40 peers whose addresses have been provided by the tracker. We next continuously discover new peers (peers that contacted us) up to the end of the download (at time 4,500 seconds) where we observe a sudden decrease of the number of peers, most probably because the seeds we where connected to have closed their connection to us as soon as we have completed the download. After the download, we stay connected as seed to between 80 and 95 leechers (4 of them being served while the others are choked).

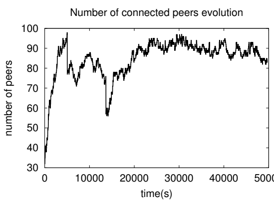


Fig. 6. Number of peers during and after download

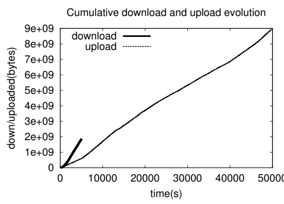


Fig. 7. Complete torrent

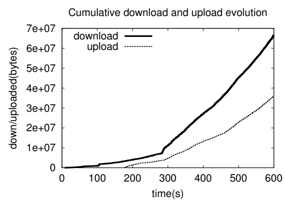


Fig. 8. First 10 minutes of download

Figures 7 and 8 show the amount of bytes downloaded and uploaded with respect to time for the complete trace and the first 10 minutes of the trace respectively. From these figures, we can draw several conclusions:

- There is a warm-up period (around 100 seconds) to obtain some first chunks. But as soon as the client has obtained a few chunks, it is able to start

uploading to others peers. This means that the rarest first policy works well, otherwise our client would maybe not find anyone interested in its first chunks.
- The download and upload rates are (positively) correlated, which indicates that the tit-for-tat policy works. It also indicates that we always find peers interested in our chunks and peers from which we can download chunks. Otherwise, we would observe some periods where the curves are flat, indicating that our client is stalled. Also, the way peer sets are built with "old" and "new" peers mixed together must have a positive impact on the efficiency of BitTorrent. Indeed, a torrent can be viewed as a collection of interconnected sets of peers. A peer that joins a torrent will obviously be the youngest peer in its initial set, but it may later be contacted by younger peers. It may also be contacted by older peers since peers try to keep contact with a minimum number of peers (generally 20) throughout their lifetimes in the torrent. This temporal diversity is a key component of the efficiency of BitTorrent, since it guarantees with high probability that a given peer will find other peers (younger or older) that hold some missing chunks of the file.
- It takes twice as much the download period to upload the same quantity of bytes (1.77GB) to the system. This illustrates the importance of peers staying as seeds once they have completed the download. This means also that we downloaded at a faster rate than we uploaded. This is due to the fact that we have a high speed link and since seeds always seek for the fastest downloaders, we should be consistently favored by the seeds that serve us.

We further investigated the type of clients we have been trading with. Overall, we found that approximately 40% of the file was provided by seeds and 60% by leechers. We also observed that more than 85% of the total file was sent by only 20 peers, including the 8 seeds that provided 40% of the file. An interesting remark is that these 20 top uploaders were not in our initial peer set provided by the tracker, but contacted us later. We can thus conjecture that to obtain the best possible performance with BitTorrent, clients should not be behind firewalls or NATs that prevent inbound connections.

We also want to assess the efficiency of the tit-for-tat policy. A good tit-for-tat policy should enforce clients to exchange chunks with one another, but with enough flexibility so as not to artificially block transfers when data does not flow at the same rate in both directions. BitTorrent avoids this type of problem by chocking/unchoking connections every 10 seconds, a long period at the time scale of a single TCP connection. We have studied the correlations between upload and download throughput, as well as between upload and download traffic volumes. Results show that, while traffic *volumes* are correlated (positive correlation close to 0.5), the upload and download *throughputs* are not correlated (close to 0), which means that BitTorrent is flexible. We also computed the correlations between download volumes and download throughputs on one side and upload volumes and upload throughputs on the other side. Both are correlated with value 0.9 and 0.5, respectively. The high correlation observed between downloaded throughputs and volumes is probably due to the fact that the top three downloaders are seeds that provide 29% of the file. The upload throughput and volume are also correlated because, once our client becomes a seed, it continuously seeks for the best downloaders, which is not the case during the download phase where a peer primarily seeks for peers that have some chunks he is interested in.

6 Conclusion

Large content replication has become a key issue in the Internet; e.g. big companies are interested in a replication service to update simultaneously a large set of hosts (e.g., virus patches or software updates). BitTorrent is a very popular peer-to-peer application targeted at large file replication. We have extensively analyzed a large torrent (with up to a few thousands simultaneously active clients) over a large period of time. The performance achieved, in terms of the throughput per client during download and the ability to sustain high flashcrowd, demonstrate that BitTorrent is highly effective.

There still remain open questions in the area of large content replication such as: (i) what is the optimal (at least theoretically) replication policy, in terms of response times, for a given user profile (arrival rates, willingness to stay as seeds), (ii) how to build a robust replication service, i.e. to ensure that all machines eventually complete the downloads (a must for corporate usage), (iii) how to efficiently protect these applications against denial of service and malicious peers.

Acknowledgment

We are extremely grateful to Eike Frost for kindly providing us with the tracker log of the torrent analyzed in this paper.

References

1. E. Adar and B. A. Huberman, "Free Riding on Gnutella", *First Monday*, 5(10), October 2000.
2. L. Cherkasova and J. Lee, "FastReplica: Efficient Large File Distribution within Content Delivery Networks (USITS 2003)", In *Proceedings of the 4th USENIX Symposium on Internet Technologies and Systems*, March 2003.
3. Y.-H. Chu, S. G. Rao, and H. Zhang, "A case for end system multicast", In *ACM SIGMETRICS 2000*, pp. 1–12, Santa Clara, CA, USA, June 2000.
4. B. Cohen, "Incentives to Build Robustness in BitTorrent", http://bitconjurer.org/BitTorrent/bittorrentecon.pdf, May 2003.
5. J. Jannotti, D. K. Gifford, and K. L. Johnson, "Overcast: Reliable Multicasting with an Overlay Network", In *Proc. 4-th Symp. on Operating Systems Design and Implementation*, Usenix, October 2000.
6. S. Ratnasamy, M. Handley, R. M. Karp, and S. Shenker, "Application-Level Multicast Using Content-Addressable Networks", In *NGC 2001*, pp. 14–29, November 2001.

7. P. Rodriguez and E. W. Biersack, “Dynamic Parallel-Access to Replicated Content in the Internet”, *IEEE/ACM Transactions on Networking*, 10(4):455–464, August 2002.
8. A. Rowstron et al., “SCRIBE: The design of a large scale event notification infrastructure”, In *Proc. NGC 2001*, November 2001.
9. R. Sherwood, R. Braud, and B. Bhattacharjee, “Slurpie: A Cooperative Bulk Data Transfer Protocol”, In *Proceedings of IEEE INFOCOM*, March 2004.