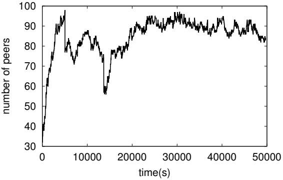

100
90
80
70
60
50
40
30
number of peers
0
10000
20000
30000
40000
50000
time(s)

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 flash-crowd, 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.