

Achieving High-Bandwidth Peer-to-Peer File Distribution

Michelle Teo, Cristina Carburaru, Ben Leong, Yashas Nataraj,
Hoang Minh Le Vu, Raymond Tan, and Yong Meng Teo
Department of Computer Science
National University of Singapore

Commercial entities have started using peer-to-peer (P2P) algorithms for file distribution. For example, Sub Pop Records and Blizzard Entertainment use BitTorrent [2] to distribute large files to their clients. Given that last-mile bandwidths to home users are increasing rapidly, a P2P approach is perhaps the only viable solution to avoid a bottleneck at the servers. Furthermore, the fact that P2P algorithms exploit the bandwidth available to the clients and reduces the provisioning costs at the server, makes P2P even more attractive.

In our work, we ask and attempt to answer the following questions by investigating the BitTorrent [2] and Slurpie [3] protocols: (i) Are existing P2P algorithms effective for file distribution? (ii) Is the available bandwidth of the clients utilized efficiently? (iii) How much server load do existing protocols incur and how efficiently do these P2P algorithms utilize available server bandwidth? We study these protocols in the context of the *file distribution* problem, where the goal is to *upload a file from a server to a large number of clients as quickly as possible*. File sharing is not the same problem as file distribution because efficiency is often not the overriding consideration for file sharing applications.

We evaluate these protocols by selecting a random set of $n+1$ geographically-dispersed PlanetLab nodes, picking one of them at random as the server, and uploading a 100 MB file from this server to the remaining n client nodes using BitTorrent. All client nodes start downloading the file at approximately the same time and we record the time taken for each data block. Immediately following the BitTorrent download, we repeat the same process with Slurpie, and then with TFTTP, with the same server and the same set of client nodes. This allows us to minimize variations in the network conditions

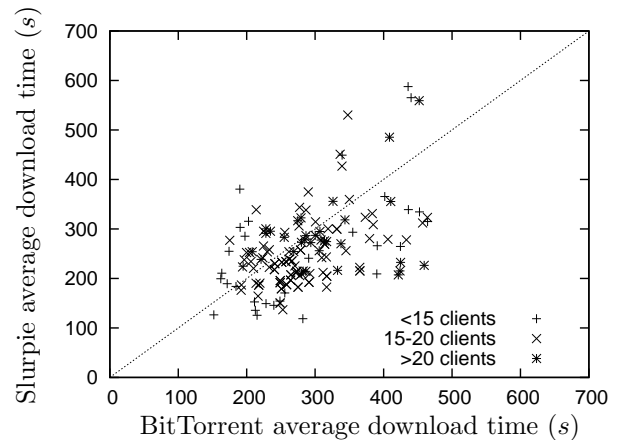


Figure 1: Comparison of average download times for 100 MB file.

and have a basis for comparing the two algorithms.

The average download times for BitTorrent and Slurpie are shown in Figure 1. Each point on the plot corresponds to the same set of server and client nodes for both BitTorrent and Slurpie. We observe that the average download times for each experiment for both protocols are comparable. It turns out that the plot of average download times does not provide us with the complete picture. In Figure 2, we plot the download times on a per-node basis and observe that Slurpie achieves more uniform download times with the majority of nodes completing within 400 s.

These results demonstrate that while BitTorrent is a successful and widely-used P2P file sharing algorithm, it is not optimized for high-speed data transfer and is hence not the optimal choice for P2P file distribution. BitTorrent and Slurpie are unable to “match” the downloading of blocks between peers efficiently, and they do not have sufficiently robust incentive schemes. Based on these insights, we designed a new P2P algorithm for file distribution, called the *Tit-for-Tat Transfer Protocol (TFTTP)*.

TFTTP incorporates two key innovations to address the shortcomings in BitTorrent and Slurpie: (i) first, in addition to allowing peers to trade blocks they al-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ACM CoNEXT 2008 Student Workshop, December 9, 2008, Madrid, SPAIN

Copyright 2008 ACM 978-1-60558-264-1/08/0012 ...\$5.00.

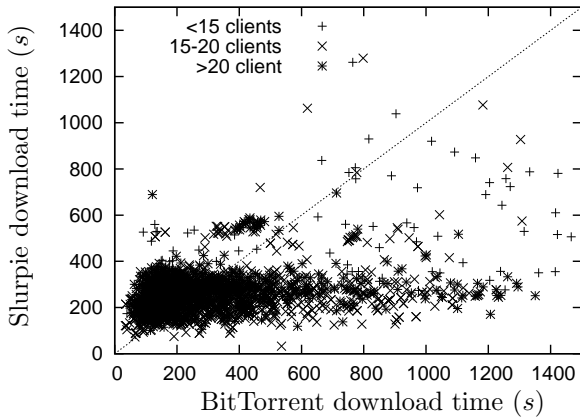


Figure 2: Comparison of download times for 100 MB file on a per-node basis.

ready possess, we introduce a new mechanism called a *promise*, that is an agreement between a pair of nodes to trade blocks that they may not already possess but are expected to receive from a peer or the server in the near future; (ii) second, these trades are conducted on a *block-for-block* basis, which is much more stringent than the rate-based incentive scheme in BitTorrent.

Our preliminary experiments (see Table 1) show that TFTP achieves on average download times 45% and 35% lower than BitTorrent and Slurpie respectively. When we consider the performance of individual nodes as shown in Figure 3, TFTP is faster about 80% of the time, more than twice as fast 50% of the time, and more than three times faster about 20% of the time.

In Figure 4, we compare the ratio of blocks uploaded to blocks downloaded in each experiment for TFTP to BitTorrent. The results for Slurpie are similar. As to be expected from the block-for-block mechanism, the share ratio for TFTP never exceeds one, unlike BitTorrent and Slurpie. We also see that there is large variation in share ratios of the individual nodes for BitTorrent. This is rather surprising, as we had expected the BitTorrent tit-for-tat mechanism to have ensured that download rates will approximately match upload rates. However, in reality, BitTorrent’s rate-based incentive mechanism is quite ineffective in enforcing fairness; low bandwidth peers tend to contribute less to the swarm compared to

Table 1: Aggregate statistics for 100 MB downloads.

Average Value	TFTP	BitTorrent	Slurpie
Download Time (s)	173	305	265
Throughput (kB/s)	973	547	460
Share Ratio	0.713	0.800	0.906

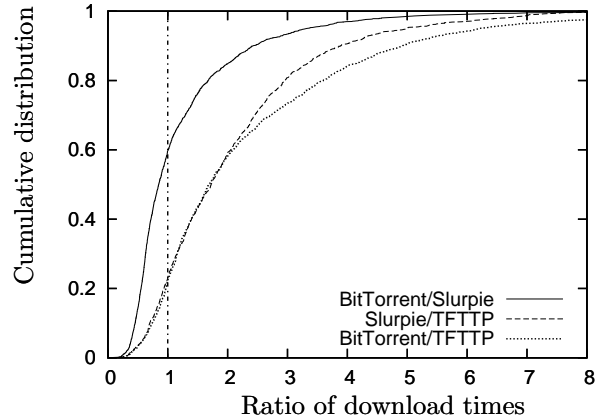


Figure 3: Cumulative distribution of the ratio of download times.

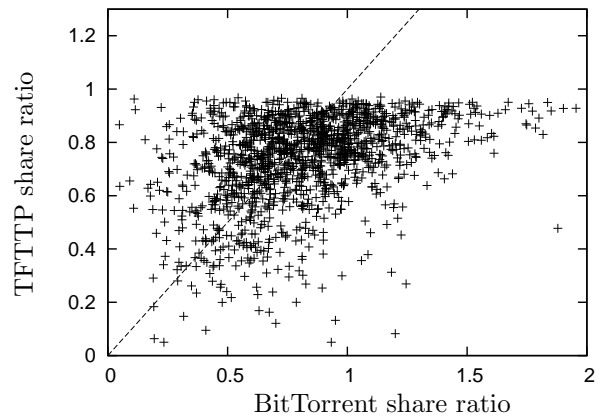


Figure 4: Distribution of share ratios for BitTorrent and TFTP on a per-node basis.

high bandwidth peers [1].

Our current implementation of TFTP is an unoptimized proof-of-concept implementation and we have to admit that we have not fully understood how it should be tuned. TFTP supports a simple interface that is amenable to different strategies at both the server and client, and there is significant scope for further innovation.

REFERENCES

- [1] A. R. Bharambe, C. Herley, and V. N. Padmanabhan. Analyzing and improving a BitTorrent networks performance mechanisms. In *Proceedings of INFOCOM '06*.
- [2] B. Cohen. Incentives build robustness in bittorrent. In *Proceedings of the P2P Economics Workshop*, 2003.
- [3] R. Sherwood, R. Braud, and B. Bhattacharjee. Slurpie: A cooperative bulk data transfer protocol. In *Proceedings of the IEEE INFOCOM 2004*.