CS 136 Pset2: Peer-to-Peer BitTorrent Simulation

Peter Hickman & Daniel Yue
Team PH&DY

February 12, 2016

1 Analysis

(a) We followed the strategy in the text and assumed the following:

(i) one "round" in our game corresponded to 10 seconds in the behavior of the client described in the text. We thus recomputed which peers to unchoke each round.

(ii) for optimistic unchoking, we should keep the same person optimistically unchoked for *three* rounds in a row, then unchoke someone new. If the person we optimistically unchoked in round t ended up in our top three uploaders in round $t+1$ or round $t+2$, then we had them both optimstically unchoked and regularly unchoked (comprising of two upload spots), thus receiving a double portion of our upload bandwidth. This strategy was more simple to implement than optimistically unchoking someone new whenever our previously optimistically unchocked person entered the top 3 uploaders, and functions comparably.

(b) We started with the BitTyrant client for our Tourney client because its strategy of unchoking people who gave it the best download/upload ratio seemed the one with the most potential to keep our downloads high given our limited bandwidth. We tried several optimizations including adopting the rarest first request strategy rather than requesting from everybody randomly, but this actually slowed down the client download speed dramatically. Another optimization that we tried was the resetting of the estimated needed upload amount u_j every few rounds, so as to avoid over or understimating the quantity, but this also seemed to slow the client. In the end, the only optimization that we kept was to increase the amount uploaded to people by a small amount, so as to force an "overestimate." This seemed to improve the runtime of the simulations, indicating that we had been underestimating the amount needed to upload to other peers in our algorithm.

(c) Table 1 reports the average download completion rounds for a single *Tyrant, Tourney,* and *PropShare* client when it is in a field of 20 other standard clients. We use the "likely parameters for the competition" described in the assignment specification and include two seeds. We do 10 iterations rather than the specified 256 iterations, for time's sake.

Table 1: Performance of PH&DY Clients

Test Client	Ref Comp. Round (Mean)	Test Comp. Round	Pct Change
PhadyTyrant	262.8	228.9	-12.9%
PhadyTourney	261.5	223.4	-14.6%
PhadyPropShare	262.8	263.3	+0.2%

Values reported are averaged over 10 iterations.

While PhadyPropShare is virtually identical to the reference client, PhadyTyrant does significantly better. It appears that a strategy of sharing upload bandwidth proportionally, rather than doing an even split, yields no improvement. However, sharing upload bandwidth to maximize the download/upload ratio yields a quicker download time. The tournament client performs slightly better than PhadyTyrant, suggesting that our optimization to Tyrant was useful.

(d) Table 1 reports the average download completion rounds for populations of 20 of the given test clients and 2 seeds, with the specifications otherwise the same as in part (c).

Table 2: Performance of PHADY Clients

Test Client	Comp. Round (Mean)	Comp. Round (Final)
PhadyStd	260.8	262.5
PhadyTyrant	572.7	598
PhadyTourney	602.7	615.1
PhadyPropShare	662.1	676.9

Values reported are averaged over 10 iterations.

Surprisingly, Tyrant, Tourney, and PropShare were all much worse in terms of average completion time and worst-cast completion time, with Tyrant slightly better than Tourney and Tourney slightly better that PropShare. Thus, we see that these clients do not share as generously as does the reference client. We hypothesize that we may have "rich get richer" phenomenon in which some peers finish quickly and then stop contributing while other peers lag behind and finish much later.

(e) We learned that it is very important to develop data structures to keep track of information about previous interactions with other clients. We also realized that implementing a game theory algorithm often requires making assumptions about how the *general* algorithm should be implemented in a *specific* situation. Finally, we learned that when assessing a client's performance, the context of the simulation is key. If we run PhadyTyrant with only dummies, PhadyTyrant does not perform well because the dummies do not reciprocate. But if we run PhadyTyrant with a group of reference clients, PhadyTyrant is able to exploit the strategies of the reference clients and do well.

2 THEORY

(a) (i) There are many more actions in PtP than in the prisoner's dilemma, since one can choose not only whether to unchoke another peer or not but how much upload bandwidth to give to them.

(ii) PtP involves many clients, not just two players as in the prisoner's dilemma. This is important because another client's behavior is influenced by the behavior of all of its peers, not just the peer that it is interacting with.

(iii) Interactions depend upon the availability of specific pieces. That is, a given client is not agnostic about which peers to interact with.

(b) (i) The TfT strategy requires exact reciprocation of action of the opponent, where as the reference client of BitTorrent only follows the *general* principle of giving data to the people that give data. (Not reciprocating exact bandwidth.)

(ii) The TfT strategy would require that the client reciprocate to everybody that shared data, but the reference client only shares consistently with its top contributors.

(iii) The reference client has a slot for *optimistic unchoking* which allows for free uploading of data in the hopes of finding new partners. This differs from TfT in that an optimistic component continues throughout the game, rather than in TfT, where optimism is only expressed in the cooperating start state.

(c) (i) Even if there was a client that was a Nash equilibrium, the game might not remain in this equilibrium because *multiple* users could switch to a new client. While if only *one* user switched to a new client, it would not be advantageous, if *multiple* users switched, there could be benefits to those who switched. We already know that in BitTorrent, some users behave altruistically, and thus it is not a far stretch to think that some users might switch for "irrational" reasons, which would lead other users to "rationally" switch.

(ii) Different client users may have different objectives. Some users might want to minimize their download time, while others want to also minimize their uplaod bandwidth. Some users might even want to share as much with others as possible. Thus, there may not be a single client that is always a "best response" to itself, because the "best response" varies by agent.