Do incentives build robustness in BitTorrent?

Bernardo Gonzalez Riede   Xin Ren

I. Motivation

The main motivation for the authors was the fallacy of previous works. Previous works have been executed in synthetic or small environments with conclusions reflecting said settings. This lead to the acceptance of BitTorrent's incentive model as correct where peers with higher capacity are rewarded through higher download speeds. Given the existence of altruism in the network, the authors expect a *strategic* client to be able to exploit it. Proofing such an exploit would change the perception of the incentive in BitTorrent.

II. Contributions

The lesser contribution is the data amassed from analyzing different swarms. Data gained from the analysis is depicted in several graphics with a common tendency; high capacity peers don't receive benefits proportional to their contributions. Moreover, the caveats of the reference implementation at the time of writing the paper are defined and through a custom implementation improved. The insight gained from studying swarm's behaviours is transformed to build a strategic client which maximizes its download capacity by exploiting the existent altruism in the network/swarms.

Regardless of the possibility of strategic clients an important fact is conveyed indirectly. *High capacity peers are of considerably importance for (initial) distribution*

III. Solution

The solution is to experiment with a strategic client, called BitTyrant, running with BitTyrant unchock algorithm developed according to the modeling of altruism, which is based on a distribution of upload capacity and equal split got from empirical measurements of BitTorrent swarms and hosts.

The distribution is obtained by observing 301,595 unique BitTorrent IP addresses over a 48 hour period during April, 2006 from 3,591 distinct ASes across 160 countries and applying reference implementation of BitTorrent.

Based on the distribution, the potential sources of altruism are located:

- By presenting expected TFT matching time as a function of upload capacity, it shows that the time is longer if the upload capacity is higher, which suggests that high capacity clients are forced to peer with those of low capacity while searching for better peers via optimistic unchokes.
- By presenting the expectation of reciprocation probability as a function of each equal split or upload capacity, it shows the certain equal split that assures reciprocation, which suggests the capacity beyond that may be altruistic.
- By presenting the expected download rate as a function of upload capacity, it shows sub-linear growth which suggests altruism from high capacity peers.

Based on above potential sources of altruism, two definitions of altruism are proposed.

- the difference between expected upload rate and download rate
- any upload contribution that can be withdrawn without loss in download performance

With altruism defined, the BitTyrant unchoke algorithm is developed. Each host, for each peer p, maintains estimates of expected download performance d_p and upload required for reciprocation u_p. Each TFT round, each host ranks peers by the ratio d_p/u_p and unchokes those of top rank until the upload capacity is reached. d_p is initially the expected equal split capacity of p, and at the end of each round for each unchoked peer, if p unchokes the host, d_p is updated to the observed download rate from p. u_p is initially the rate just above the step in the reciprocation probability and then periodically updated depending on whether p reciprocates for an offered rate, that is at the end of each round, if p does not unchoke the host, u_p increases by a configurable ratio, while if p has unchoked the host for certain configurable rounds, u_p decreases by another configurable ratio. The algorithm can be easily generalized to handle concurrent downloads from multiple swarms by ordering the d_p/u_p ratios of all connections across swarms.

The strategic client BitTyrant is designed with algorithm above, which allows the client to dynamically size its active set and vary the sending rate per connection to maximize reciprocation bandwidth per connection and number of reciprocating peers, so that the performance of the client is improved, especially for high capacity peers. According to the algorithm, high capacity peers would need a large size of active set, which means the need to increase the local neighborhood size. This is achieved by requesting as many peers as possible from the centralized tracker at the maximum allowed frequency and increasing the query rate of the DHT-based distributed tracker recently incorporated by the BitTorrent protocol.

To evaluate BitTyrant, experiments were executed with a single strategic peer in synthetic and current real world swarms as well as BitTyrant used by all participants in synthetic swarms. The single strategic peer experiment explores the performance of a single strategic peer by comparing its downloading time to that of an unmodified Azureus client in same swarms, while the experiment with BitTyrant used by all participants explores swarm performance by comparing the downloading time in swarm full of strategic peers and both strategic and selfish peers to that in swarm with only unmodified peers.

IV. Strong Points

1) The differences between the paper and other related work are clearly stated.
2) References are well listed and noted with hyperlinks.
3) Predicts readers thoughts and mentions topics which covers difficulties & downsides later on.

V. Weak Points

1) The repetitions of the experiments are not mentioned.
2) There exists an interesting point at an upload capacity of around 48 KB/s which is repeated in several graphs though the paper sheds no light on its peculiarity.
3)

VI. Questions

A. Briefly explain the most important piece selection policy used by BitTorrent to improve piece diversity, and how this policy helps reduce download times.