BitTorrent and BitTyrant

CS 344G 2/9/2015

Gregory D. Hill

BitTorrent Overview

- Download .torrent file or use magnet URI to get file hash, required metadata
- Get partial list of peers from Tracker or Kademlia-based DHT
- Download file from peers following peer protocol (uses TCP)
  - 10 message types described in following slides
    - keep-alive
    - choke/unchoke
    - Interested/not interested
    - have
    - bitfield
    - request
    - piece
    - cancel

Basic BitTorrent Protocol

- Handshake with peer, after send **bitfield**(length, bitfield) message which has a bit for each piece of the file representing whether that peer has that piece or not
- **have**(piece index) message – tell a peer you have piece x
- **request**(piece index, byte offset, length) to a peer hopefully reciprocated by a **piece**(piece index, byte offset, data) which can be aborted with a **cancel**(piece index, byte offset, length)
- Send **interested**() to a peer that has data you want and undo this by sending them **not interested**()

Piece Selection

- First piece: choose at random until a complete piece downloaded
- Normal: Rarest First, try to download piece fewest peers have
- Endgame: on last pieces, request all subpieces left from all peers, send **cancels** to others when data arrives

BitTorrent Choking

- Refuse to upload to, or **choke**, all but a few (commonly 4) peers
  - Upload even share to these unchoked peers
- Prefer to give to peers who upload most to you (tit-for-tat)
  - Every 10 seconds decide which mutually interested peer to unchoke based on best rolling 20 second average of download rate from peers
  - If no reciprocation from an unchoked peer for over a minute, assume snubbed and don’t upload to them
- In addition, rotate through peers and choose a new one to **optimistically unchoke** every 30 seconds, which searches for better peers
  - 30 seconds is enough time for them to unchoke you if you upload faster than their currently unchoked peers
- Once finished downloading, prefer peers you can upload to fastest

Mismatched claims

- In *Incentives Build Robustness in BitTorrent*, Bram Cohen (creator of BitTorrent) claims "choking algorithms attempt to achieve pareto efficiency" and that BitTorrent "achieves a higher level of robustness and resource utilization than any currently known cooperative technique"
- *Do incentives build robustness in BitTorrent?*, from the University of Washington say their results "suggest that incentives do not build robustness in BitTorrent" and that "robustness requires that performance does not degrade if peers attempt to strategically manipulate the system"

Key Figure

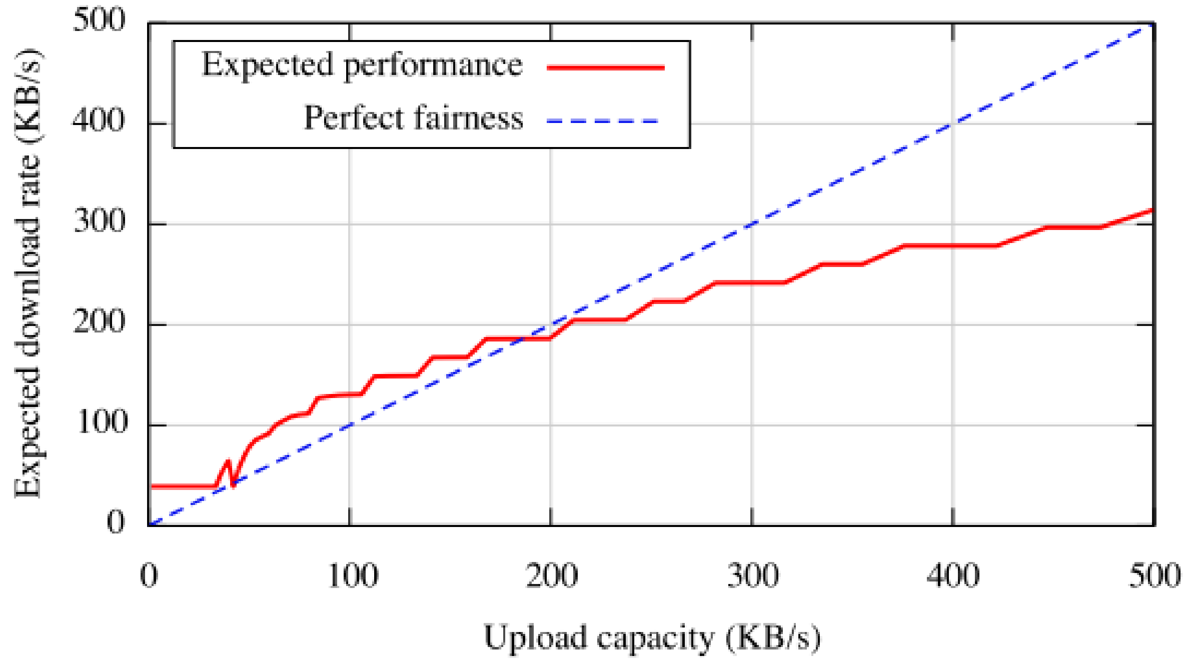
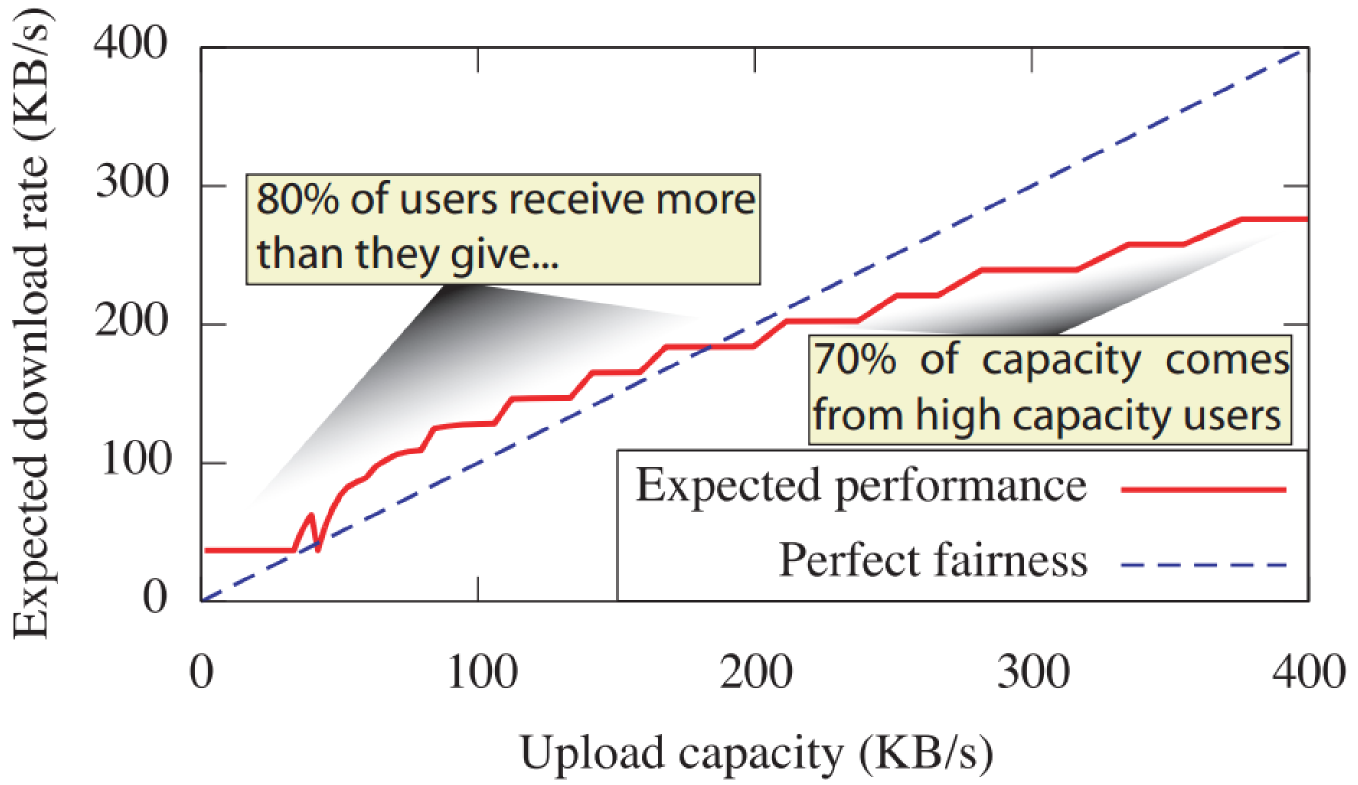


Figure 4: Expectation of download performance as a function of upload capacity. Although this represents a small portion of the spectrum of observed bandwidth capacities, ~80% of samples are of capacity $\leq$ 200 KB/s.

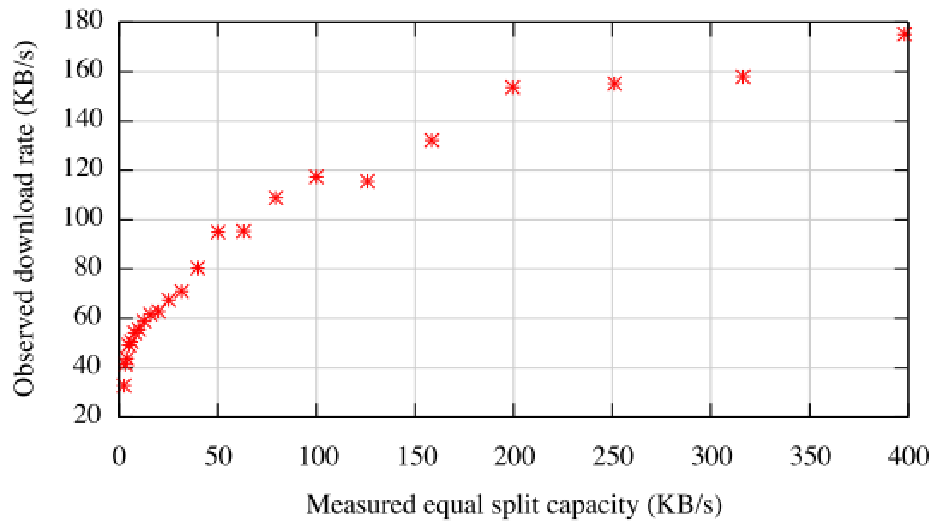
Key Figure



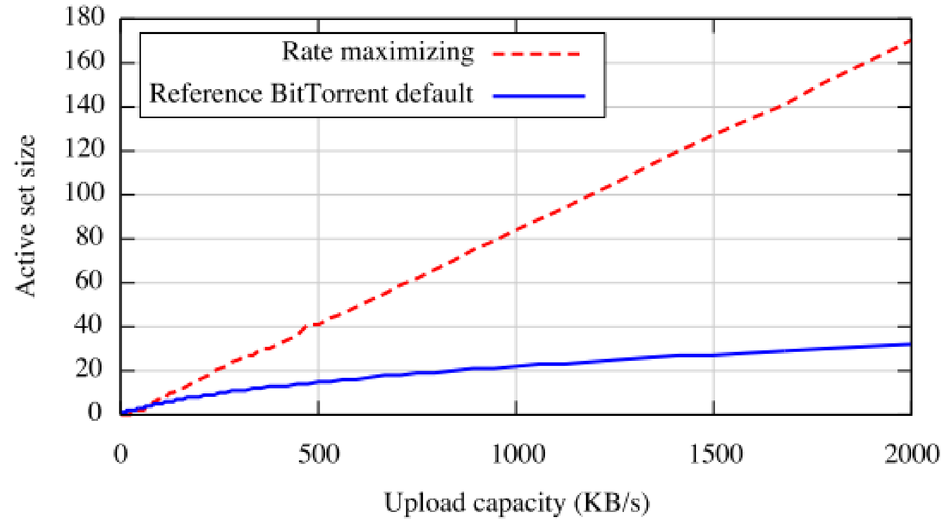
http://www.michaelpiatek.com//papers/BitTyrant_affiliates.pdf

BitTyrant Paper

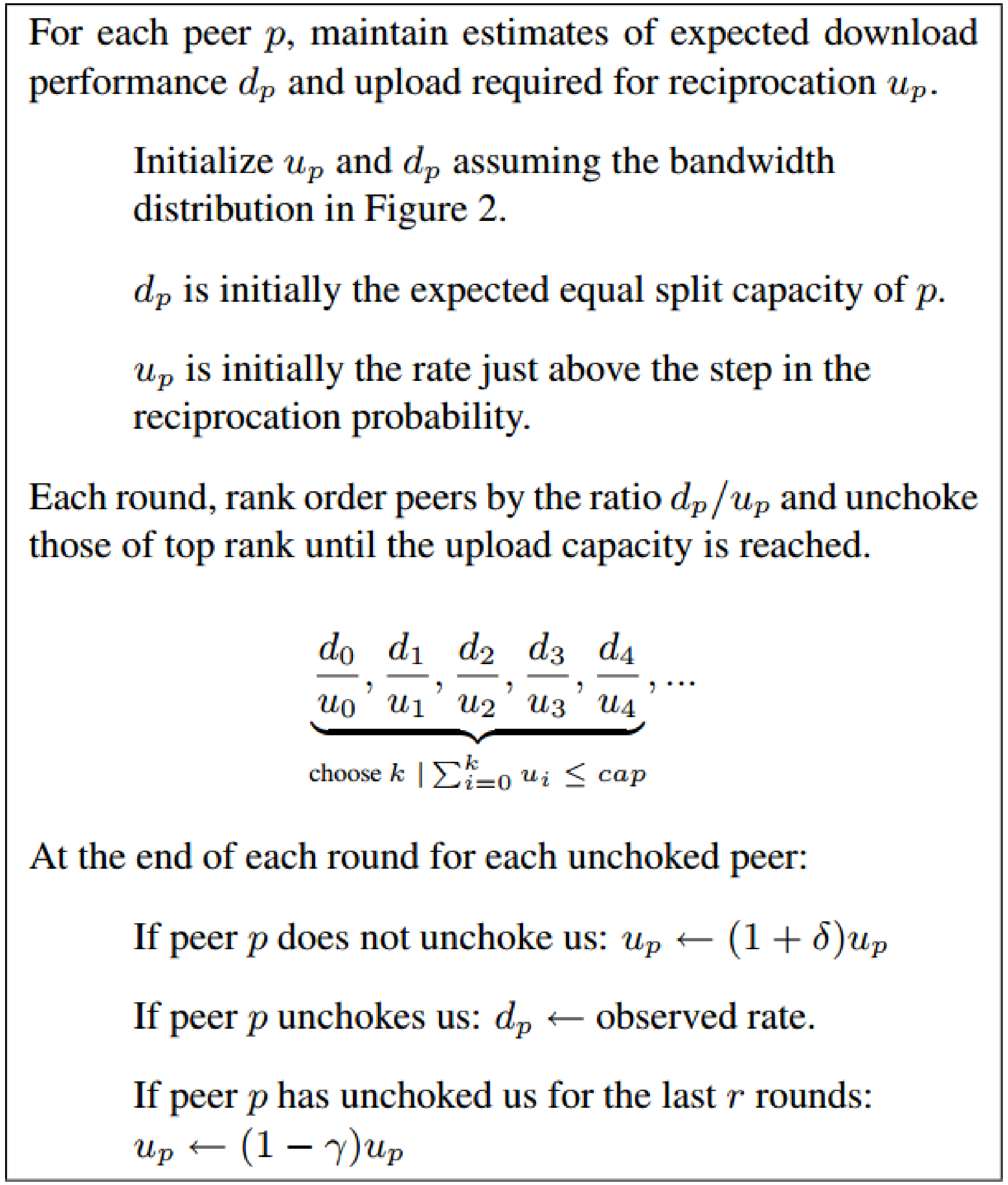
- Claim much of performance due to “altruistic” reasons
- Build “strategic” BitTorrent client called BitTyrant which achieves better performance than other clients
- Focus on number of peers to unchoke and how much to upload to each



- Diminishing returns to sharing more with a single peer



- So modify active set size to maximize download speed



For each peer p, maintain estimates of expected download performance d_p and upload required for reciprocation u_p.

```
    Initialize $u_p$ and $d_p$ assuming the bandwidth distribution in Figure 2.

    $d_p$ is initially the expected equal split capacity of $p$.

    $u_p$ is initially the rate just above the step in the reciprocation probability.
```

Each round, rank order peers by the ratio d_p/u_p and unchoke those of top rank until the upload capacity is reached.

$$\underbrace{\frac{d_0}{u_0}, \frac{d_1}{u_1}, \frac{d_2}{u_2}, \frac{d_3}{u_3}, \frac{d_4}{u_4}}_{\text{choose } k \mid \sum_{i=0}^{k} u_i \leq cap}, \ldots$$

At the end of each round for each unchoked peer:

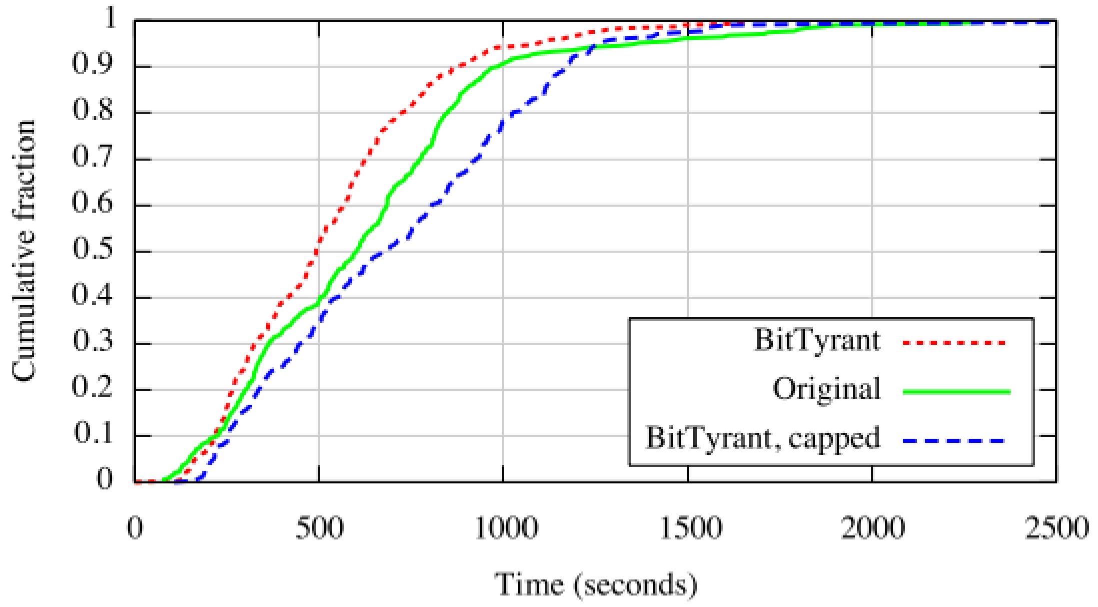
```
    If peer $p$ does not unchoke us: $u_p \leftarrow (1+\delta)u_p$

    If peer $p$ unchokes us: $d_p \leftarrow$ observed rate.

    If peer $p$ has unchoked us for the last $r$ rounds:
    $u_p \leftarrow (1-\gamma)u_p$
```

Figure 9: *BitTyrant* unchoke algorithm

Performance



Bittorrent Questions

- Why not seed to people who share more (maybe add got-from or tree hierarchy to **have** messages)
- Is Rarest First piece selection the best?
- Why TCP?

BitTyrant Questions

- Why pose as evil

BitTorrent extras

- Super-Seeding
- Peer Exchange
- Local Peer Discovery