BitTorrent

Katrina LaCurts
MIT CSAIL
katrina@csail.mit.edu

(much content borrowed from Dave Levin, dml@cs.umd.edu)
BitTorrent Overview

peer
BitTorrent Overview

peer

tracker
BitTorrent Overview

peer

tracker

130.136.254.21
130.136.254.22
171.66.3.182
128.232.103.202
155.98.35.4
128.163.142.20
...

...
BitTorrent Overview

peer

leechers
BitTorrent Overview

Peer

Neighbors
BitTorrent Overview

peer

neighbors
BitTorrent Overview

peer

neighbors
BitTorrent Overview

- peer
- neighbors
- seeders
Phases of BitTorrent

- **Bootstrapping**: Getting the first pieces
- **Steady-state**: Trading with peers
- **End-game**: Getting the last pieces
Phases of BitTorrent

0%

Bootstrapping: Getting the first pieces

% Downloaded

Steady-state: Trading with peers
assumption: peers have pieces to trade with other peers

End-game: Getting the last pieces
Steady-state

in steady-state, a BitTorrent peer uploads to and downloads from different neighbors
in steady-state, a BitTorrent peer uploads to and downloads from different neighbors.
Steady-state

in steady-state, a BitTorrent peer uploads to and downloads from different neighbors

how does he decide who to upload to, how much to upload, etc.?
Steady-state

Round $t$

divide protocol into rounds. peers that upload the most to us in round $t$ get uploaded to in round $t+1$
Steady-state

divide protocol into *rounds*. peers that upload the most to us in round \( t \) get uploaded to in round \( t+1 \)
BitTorrent’s Unchoker

Round $t$

13
10
4
12
7
9
15
BitTorrent's Unchoker

Round $t$

13
10
4
12
7
9
15
BitTorrent’s Unchoker

20?
BitTorrent's Unchoker
BitTorrent's Unchoker

15
13
12
10
20

20?
BitTorrent's Unchoker

20?
BitTorrent’s Unchoker
BitTorrent’s Unchoker
BitTorrent's Unchoker

15
13
12
10
11

11?
BitTorrent’s Unchoker

10
10
10
10

11?
BitTorrent’s Unchoker

Best strategy: Come in last

peers do *not* have incentive to give as much as possible

BitTorrent's Unchoker
BitTorrent’s Unchoker
BitTorrent’s Unchoker
BitTorrent’s Unchoker
BitTorrent’s Unchoker
BitTorrent’s Unchoker

Sybil Attack: Create additional identities to subvert the system
PropShare Unchoker

Round t

PropShare Unchoker

Round t

PropShare Unchoker

Round $t$

13
10
4
12
7
9
15

Total: 70

PropShare Unchoker

Round $t$

13
10
4
12
7
9
15

Round $t+1$

13/70
10/70
4/70
12/70
7/70
9/70
15/70

Total: 70

PropShare Unchoker

[Diagram showing yellow pieces moving towards a green keyhole with an arrow labeled 50, and a thought bubble with the number 20]
PropShare Unchoker

[Diagram showing 5 yellow key icons moving to a green key icon with an arrow labeled 50 and another green key icon with an arrow labeled 20 moving to a red key icon. A thought bubble with the number 20?]
PropShare Unchoker

Total: 70

[Diagram showing 5 yellow key icons moving towards a green key hole with an arrow of 50, and another green key hole with an arrow of 20 moving away from a red key hole.]

20?
PropShare Unchoker

Total: 70

50/70  20/70

20?
PropShare Unchoker

50
PropShare Unchoker

50

11

11?
PropShare Unchoker

Total: 61

50

11?
PropShare Unchoker

Total: 61

50/61  11/61
PropShare Unchoker

Total: 61

Upload Less →
Receive Less →
Incentive to Upload More
PropShare Unchooker

Total: 70

50

20
PropShare Unchoker

Total: 70

50/70  20/70
PropShare Unchoker

Total: 70

50/70 + 20/70 = 70

5 + 5 + 5 + 5 = 20
PropShare Unchoker

Total: 70

PropShare is Sybil-proof
Steady-state Results

• BitTyrant and PropShare are both faster than BitTorrent
  • For different reasons
• PropShare performs comparably to BitTyrant
• PropShare does *not* suffer from a *tragedy of the commons*
  • BitTyrant does
Phases of BitTorrent

0% - Bootstrapping: Getting the first pieces

% Downloaded

Steady-state: Trading with peers

100% - End-game: Getting the last pieces
Phases of BitTorrent

- **Bootstrapping**: Getting the first pieces
  assumption: peers have *nothing* to give to other peers

- **Steady-state**: Trading with peers

- **End-game**: Getting the last pieces
Optimistic Unchoking

reserve a portion of bandwidth to give freely to other peers
(presumably new peers)
Optimistic Unchoking

reserve a portion of bandwidth to give freely to other peers
(presumably new peers)
Optimistic Unchoking

reserve a portion of bandwidth to give freely to other peers
(presumably new peers)
Optimistic Unchoking

reserve a portion of bandwidth to give freely to other peers
(presumably new peers)
Optimistic Unchoking

reserve a portion of bandwidth to give *freely* to other peers
(presumably new peers)
Optimistic Unchoking

reserve a portion of bandwidth to give freely to other peers
(presumably new peers)
Optimistic Unchoking

reserve a portion of bandwidth to give *freely* to other peers (presumably new peers)
Optimistic Unchoking

reserve a portion of bandwidth to give freely to other peers
(presumably new peers)
Optimistic Unchoking

reserve a portion of bandwidth to give freely to other peers
(presumably new peers)
Optimistic Unchoking

reserve a portion of bandwidth to give freely to other peers (presumably new peers)

exploit: always asked to be optimistically unchoked (i.e., never upload)
Optimistic Unchoking

reserve a portion of bandwidth to give freely to other peers (presumably new peers)

explore: always asked to be optimistically unchoked (i.e., never upload)

tragedy of the commons: system will collapse if everyone does this

Locher, et al. “Free Riding in BitTorrent is Cheap”. HotNets, 2006
Junk Updates

force peers to upload useless data

Junk Updates

force peers to upload useless data

Junk Updates

force peers to upload useless data

Junk Updates

force peers to upload useless data

Junk Updates

force peers to upload useless data

Junk Updates

force peers to upload useless data

Junk Updates

force peers to upload useless data

Junk Updates

force peers to upload useless data

Junk Updates

force peers to upload useless data

Junk Updates

force peers to upload useless data


no incentive to repeatedly ask for unchoking, but wastes system resources
Junk Updates

force peers to upload useless data


no incentive to repeatedly ask for unchoking, but wastes system resources

can we put new peers to work doing something useful?
TBS
Levin, et al.
TBS
Levin, et al.
TBS
Levin, et al.
TBS
Levin, et al.

both and get pieces from
TBS
Levin, et al.
TBS
Levin, et al.
TBS
Levin, et al.
TBS
Levin, et al.

problem: can send junk
TBS
Levin, et al.

problem: can send junk or nothing at all
TBS
Levin, et al.

Solution: Encryption

problem: can send junk or nothing at all
TBS
Levin, et al.
TBS
Levin, et al.
TBS
Levin, et al.
Bootstrapping Summary

- Bootstrapping is not a very large part of the download. Even so, it can be exploited.
- A better bootstrapping mechanism has potential to yield better performance throughout the download.
- Moreover, it can be used whenever a peer becomes uninteresting, not just in the bootstrapping phase.
Phases of BitTorrent

0%

Bootstrapping: Getting the first pieces

% Downloaded

Steady-state: Trading with peers

100%

End-game: Getting the last pieces
Phases of BitTorrent

0%

Bootstrapping: Getting the first pieces

% Downloaded

Steady-state: Trading with peers

100%

End-game: Getting the last pieces
assumption: not many peers are mutually interesting
Strategic Piece Revelation

Round $t$

Goal: Be as interesting as possible to lots of peers
Goal: Be as interesting as possible to lots of peers
Strategic Piece Revelation

Round $t$

Goal: Be as interesting as possible to lots of peers
Strategic Piece Revelation

Round $t+1$

Goal: Be as interesting as possible to lots of peers
Strategic Piece Revelation

Round $t+1$

Goal: Be as interesting as possible to lots of peers
Strategic Piece Revelation

Round t
Strategic Piece Revelation

Round t
Strategic Piece Revelation

Round $t$
Strategic Piece Revelation

Round $t$
Strategic Piece Revelation

Round \( t+1 \)
Strategic Piece Revelation

Round $t+1$
Strategic Piece Revelation

Round $t+1$
Strategic Piece Revelation

Round $t+1$

Strategically reveal pieces → Peers are interested in me longer
Peer Selection

- Before our download even starts, a BitTorrent client gets a set of peers from the tracker.
- During the download, the peer figures out the “best” of this set.
- What if we could decide which peers would be best without trading with them first?
Peer Selection

- Measuring link characteristics is sometimes seen as a threat, and doesn’t scale
- Many measurement systems require a “map” of the Internet, which is hard to obtain
- Network coordinate systems don’t require a map, but are complicated and don’t always work
- Could try simple things (use peers in our ISP, e.g.), but it’s not clear that these work either
Summary

- BitTorrent is a large system; lots of things to tweak
  - Bootstrapping, steady-state, end-game phases
  - Peer selection
- Not all strategies are fair
- A combination of techniques (from various phases) would probably result in an extremely fast client