5. Bittorrent and cooperation

Modelling Social Interaction in Information systems

http://davidhales.com/msiis

David Hales, University of Szeged

dave@davidhales.com

Quick recap of previous lecture on the evolution of cooperation

- Tragedy of the commons / social dilemmas
- The Prisoner's Dilemma game
- Iterated Prisoner's Dilemma game
- Axelrod's tournaments
- The tit-for-tat strategy and it's properties

Bittorrent

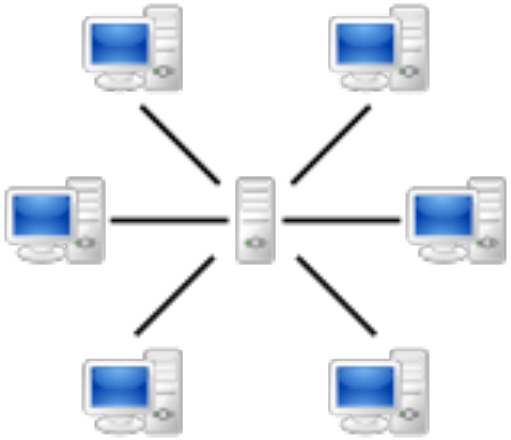
- Bittorrent is a sophisticated protocol for sharing files
- The software associated with it is complex needing to address many issues
- We will only concentrate here on one core aspect of how cooperation is maintained to improve performance
- I hope to show how ideas we have considered previously have inspired aspects of this

If you want concrete details you could start here:
https://wiki.theory.org/BitTorrentSpecification

Bittorrent

- A peer-to-peer (P2P) file-sharing protocol
- Created by Bram Cohen in 2001 (now, millions of users)
- Files are copied between peer nodes rather than from a central server
- Downloaders are also uploaders
- This is useful because:
  - Allows people without access to fast servers to share large files with lots of others quickly
  - Scales well since the more peers who want to download a file the more capacity is added to the system
  - Deals with “flash crowds” where many peers want to download the same file at the same time

P2P Architecture



Traditional Client / Server

Peer-to-Peer

From: http://en.wikipedia.org/wiki/Peer-to-peer

Aside: Bittorrent in the news

- Bittorrent is a file sharing protocol (like FTP)
- But, since it allows anyone to share large files with many people at the same time,
- It is sometimes associated with piracy of copyrighted material (movies, software etc)
- Sites like the PirateBay have nothing to do with BT, they just list .torrent files which are pointers (like hyperlinks)
- Because BT does not require powerful central servers it challenges some existing business models
- In fact, it can be argued, that the PirateBay has nothing to do with piracy since it hosts no content (yet the law says different – in many countries)

Bittorrent

- It is “open” meaning anyone can write a “client” that can connect with others supporting the “wire protocol”
- A client is a piece of software that runs on a user’s machine and connects to the BT network as a peer
- The wire protocol specifies the type and format of messages that can be sent between peers
- Each peer maintains a dynamic set of connections to other peers (essentially via their IP addresses)
- So-called “overlay network”

Overlay networks

- An overlay network is a logical network built on top of another underlying network
- Hence the links between nodes may represent a complex route through the underlying network
- In general this allows applications to maintain dynamic topologies
- Without concerning itself with the underlying network details
- In P2P systems, generally, each node maintains a dynamic set of links to other nodes

Some Bittorrent Terminology

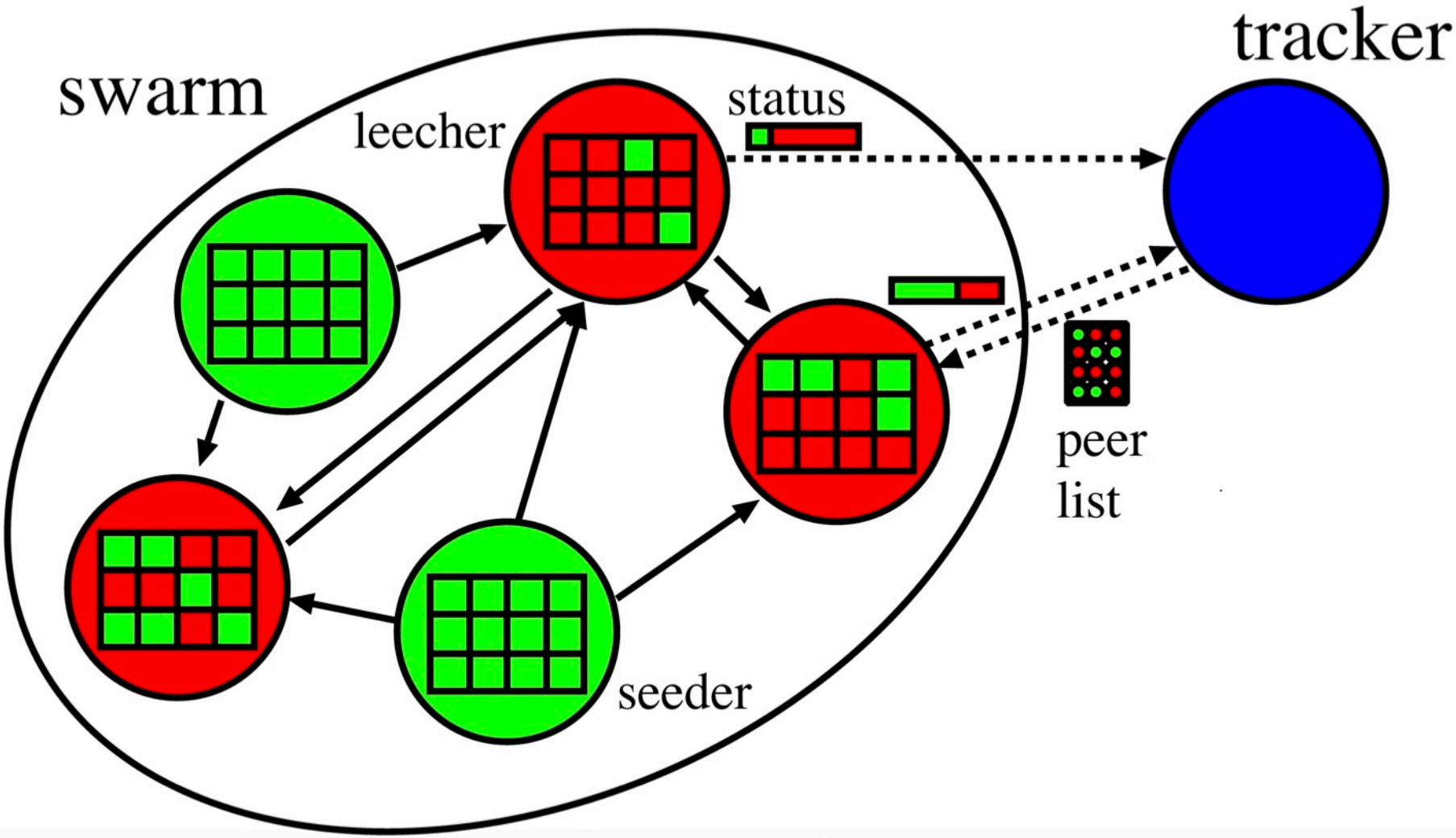
- Swarm: set of peers interested in a particular file
  - file is split in smaller chunks called pieces
  - Seeder: holds a full copy of the file
  - Leecher: holds only a part of the data (initially nothing)
- Tracker: centralised manager (or DHT)
  - keep track of all peers in the swarm
  - return list of current peers in swarm
  - Maintains some statistics about peer behaviour
- Torrent file: meta-data
  - For a file to be shared in BT a .torrent file must be created
  - contains pointer to tracker hosting the swarm for the file
  - details about the file - hashes, no. of pieces, size etc.

DHT = Distributed Hash Table. It is a way of implementing the function of the centralised tracker using a distributed P2P overlay network

Bittorrent overview

- Given a .torrent file a peer does the following:
  - Contact the tracker listed in the .torrent file
  - The tracker sends back a list of current peers in the swarm for that file
  - Connect to some of those peers
  - Ask peers in the swarm what pieces of the file they have and tell them what pieces you have
  - Download and upload appropriate pieces from / to relevant peers

Bittorrent overview



So far so good but..

- It is in the collective interest for all to upload to others so everyone gets the file quickly
- But *may* be in individual interest to save bandwidth by only downloading and hence free-riding on others
- Free-riding (or free-loading) is a perennial problem in P2P file-sharing systems
- Any efficient system needs to tackle it

A social dilemma

- Hence we have a form of social dilemma because:
- There is no central control to enforce sharing (uploading)
- Individual peers may have an incentive to free-ride by only downloading and not uploading
- Empirical examination of earlier P2P file-sharing systems showed high-levels of free-riding (see Adar & Huberman (2000) Free riding on Gnuteller)
- “tragedy of the digital commons”

Reciprocity to the rescue?

- Bittorrent is designed to incorporate incentives into the protocol to promote sharing
- It uses a variant of the “tit-for-tat” strategy we looked at previously
- Essentially those wishing to download will tend get better performance by uploading to others
- Those who never upload will tend not to get downloaded to

Incentives in Bittorrent

- Since the file is broken into many pieces
- And leeching peers can help each other by exchanging pieces with each other
- The process of uploading and downloading may be viewed as a repeated game of cooperation
- This is similar to the Iterated Prisoner's Dilemma (IPD) game we considered previously

Incentives in Bittorrent

- Uploading to another peer can be viewed as a form of cooperation (playing C in the IPD game)
- Downloading without uploading can be viewed as a form of defection (playing D in the IPD game)
- Axelrod's tournaments demonstrated that the tit-for-tat strategy, in repeated games, did well overall against other strategies submitted to him
- One can say it was "robust" against the other strategies

Bittorrent and tit-for-tat

- In his original paper Cohen (2003) gives an overview of the BT system
- It describes how a form of tit-for-tat is implemented in BT
- His aim is to achieve a form of robust "Pareto efficiency" (more detail in later lectures)
- Does not indicate influenced by the Axelrod's results directly but his paper applies some of the insights we saw in that work

B. Cohen (2003) "Incentives build robustness in bittorrent," in 1st Workshop on the Economics of Peer-2-Peer Systems.

Aside: Pareto efficiency

- An idea developed and applied in economics and engineering developed by Vilfredo Pareto
- We will look at PE in a little more detail in a later lecture however…
- But basically it means if a set of agents can change the current allocation of resources such that at least one is better off and no other is worse off then the current allocation is *not* PE
- Suffice to say mutual defection in the PD game is *not* Pareto efficient
- In the context of the BT work discussed here you can consider it as "making efficient use of resources"

Bittorrent tit-for-tat

- The implementation of tit-for-tat in BT is part of what is termed the “choking”
- Choking involves not uploading pieces to another peer even if you have the data and other peer requests it
- However, it is still possible to download from a choked peer if it continues to upload to you
- Hence any BT client needs to implement a choking algorithm (policy)
- Of course there many other aspects to BT including a piece selection algorithm (policy) but we will not cover these aspects here

Bittorrent choking

- Cohen states: “A good choking algorithm should utilize all available resources, provide reasonably consistent download rates for everyone, and be somewhat resistant to peers only downloading and not uploading”
- And also: “BitTorrent’s choking algorithms attempt to achieve pareto efficiency using a more fleshed out version of tit-for-tat than that used to play prisoner’s dilemma. Peers reciprocate uploading to peers which upload to them, with the goal of at any time of having several connections which are actively transferring in both directions. Unutilized connections are also uploaded to on a trial basis to see if better transfer rates could be found using them.”

Bittorrent choking

- Each peer maintains a small set of (say 4) unchoked peers in addition to a possibly larger set of choked peers (say 40)
- It uploads pieces (sub pieces) to all unchoked peers (sharing its upload capacity equally)
- It downloads from all peers that upload to it
- After a 10 second time period:
  - It then calculates the download it received (20 second rolling average) form all peers it is connected to
  - Updates the unchoked peer set based on those who gave best download performance

Bittorrent choking (tit-for-tat)

- By uploading a peer is “cooperating”
- By downloading but not uploading a peer is “defecting”
- The choking algorithm cooperates (uploads) to those who have in the last 20 seconds, on average, cooperated with them (downloaded) the most
- Hence this is more complex than a simple defect / cooperate binary Prisoner’s Dilemma game
- The aim is to make the best use of available bandwidth such that peers can increase their upload / download by locally optimising (in some sense)
- Those that don’t (or can’t) upload will tend to get poor download over time because other peers will choke them

Bittorrent choking (tit-for-tat)

- Cohen states: “uploading to the peers which provide the best download rate would suffer from having no method of discovering if currently unused connections are better than the ones being used. To fix this, at all times a BitTorrent peer has a single ‘optimistic unchoke’, which is unchoked regardless of the current download rate from it. Which peer is the optimistic unchoke is rotated every third rechoke period (30 seconds). 30 seconds is enough time for the upload to get to full capacity, the download to reciprocate, and the down- load to get to full capacity. The analogy with tit- for-tat here is quite remarkable; Optimistic unchokes correspond very strongly to always cooperating on the first move in prisoner’s dilemma.”

Bittorrent (seeding)

- It is interesting to note that BT has *no incentives for seeding*
- Cohen states “Once a peer is done downloading, it no longer has useful download rates to decide which peers to upload to. The current implementation then switches to preferring peers which it has better upload rates to, which does a decent job of utilizing all available upload capacity and preferring peers which no one else happens to be uploading to at the moment.”
- Interestingly, if you go on “torrent sites” you might often see comments saying “please seed!”

Incentives in Bittorrent

- Since BT is an open protocol with client software running on user devices
- Users can change the client code or limit the bandwidth allocated to the client so long as they adhere to the BT "wire protocol"
- In some sense one can view BT clients as strategies within a highly complex game
- Different clients may implement the protocol in subtly different ways – such as applying a different choking / unchoking policies (algorithms)
- For example, a peer could divide upload bandwidth to unchoked peers proportional to the download it had received

Bittorrent client ecology

BitTorrent client	FOSS	Linux/Unix	Windows	Mac OS X	IPv6[1]	Programming language	Based on	Interface	Spyware/Adware/Malware-free
ABC	Yes	Partial	Yes	No	buggy[2]	Python	BitTornado	GUI and web	Yes
Acquisition	No	No	No	Yes	↗	Objective-C and Cocoa	Limewire	GUI	Yes
Anatomic P2P	Yes	Yes	Yes	Yes	No	Python	BitTornado	GUI and old CLI	Yes
Arctic Torrent	Yes	No	Yes	No	No	C++	libtorrent	GUI	Yes
aria2	Yes	Yes	Yes	Yes	↗	C++	-	CLI	Yes
Azureus	Yes	Yes	Yes	Yes	Partial[3]	Java and SWT	-	GUI, CLI, Telnet, Web, XML over HTTP remote control API	Yes
BitComet	No	No	Yes	No	No	C++	↗	GUI	Yes[4]
BitFlu	Yes	Yes	No	Yes	Yes	Perl	-	Telnet and Web	Yes
BitLet	Planned	Yes	Yes	Yes	↗	Java and JavaScript	-	Web XHTML	Yes
BitLord	No	No	Yes	No	No	C++	BitComet	GUI	Adware
BitPump	No	No	Yes	No	No	C++	-	GUI	Yes
Bits on Wheels	No	No	No	Yes	No	Objective-C and Cocoa	-	GUI	Yes
BitSpirit	No	No	Yes	No	No	C++	BitComet	GUI	Yes
BitThief	No	Yes	Yes	Yes	↗	Java	↗	GUI	Yes
BitTornado	Yes	Yes	Yes	Yes	Yes	Python	BitTorrent	GUI and CLI	Yes
BitTorrent 5 / Mainline	Yes	Yes	Yes	Old version	No	Python	-	GUI and CLI	Yes
BitTorrent 6	No	No	Yes	No	Yes	C++	µTorrent	GUI and CLI	Yes
BitTyrant	Yes	Yes	Yes	Yes	Partial[3]	Java and SWT	Azureus	GUI, CLI, Telnet, Web, XML over HTTP remote control API	Yes
Blizzard Downloader	No	No	Yes	Yes	↗	↗	BitTorrent client for early version	GUI	Yes
Blog Torrent	Yes	No	Yes	Yes	↗	↗	BitTorrent client for early version	GUI	Malware-Status: unknown
BTG	Yes	Yes	Partial[5]	Yes	No	C++	libtorrent	CLI, GUI and web	Yes

From: http://en.wikipedia.org/wiki/Comparison_of_BitTorrent_clients

Which clients spread?
Interesting variants (with papers)

http://bittyrant.cs.washington.edu/

BitTyrant

A strategic BitTorrent client that improves performance

Paper: Piatek, M. et al (2007) Do incentives build robustness in BitTorrent? 4th USENIX Symposium on Networked Systems Design & Implementation (NSDI)

BitThief http://dcg.ethz.ch/projects/bitthief/

A Free Riding BitTorrent Client

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Paper: Locher, T. et al (2006) Free Riding in BitTorrent is Cheap. HotNets 2006

Which clients will spread?

- BT can still be cheated by exploiting the protocol
- Selfish clients have been released by researchers to see if they spread
- However, clients have been released that claim to improve on the standard BT implementation
- The BT ecology can be viewed as *global social cooperation experiment* using algorithms – rather like Axelrod's tournaments
- Several large-scale measurement papers have explored client behaviour

Pouwelse, J. et al (2005) The bittorrent p2p file-sharing system: measurements and analysis. In Proceedings of the 4th international conference on Peer-to-Peer Systems (IPTPS'05), Springer-Verlag, Berlin, Heidelberg.

Take home messages

- Distributed file sharing systems suffer from a potential “tragedy of the commons”
- A widely used system called Bittorrent address this through a form of “tit-for-tat” approach
- Previous results using the abstract model of the Prisoner’s Dilemma game provide inspiration for Bittorrent protocol
- But the implemented version required significant creativity in order to apply it the results from the abstract model

Some history

- Prior to releasing Bittorrent Bram Cohen worked for "Evil Geniuses for a Better Tomorrow" start-up company
- They created a system called Mojo Nation
- It was a general purpose P2P middleware that attempted to apply economic (market) ideas to P2P coordination
- Others could build applications on top of it (such as filesharing)
- It included:
  - Evil Geniuses Transport Protocol (EGTP) that allowed end-to-end encrypted messaging between peers
  - Digital money called "Mojo" that could be used by peers to buy and sell services from each other
- In 2002 it seems, Evil Geniuses ran out of real money and closed
- Some aspects of Bittorrent appear to have evolved out of ideas from Mojo Nation (breaking files into bits and distributing them)

See: http://web.archive.org/web/20011216040718/http://mojonation.net/intro.shtml

Readings and questions

- Paper:
  - B. Cohen (2003) “Incentives build robustness in bittorrent,” in 1st Workshop on the Economics of Peer-2-Peer Systems
- Questions:
  - Can you think of any other applications that might benefit from a tit-for-tat like approach?
  - given no incentives why do you think peers seed?
  - Do you believe that this “tit-far-tat” choking approach is responsible for the success of BT?
  - Is there, in reality, a tragedy of the digital commons when everyone has lots of bandwidth to spare?