Achieving Fairness in BitTorrent

Peter Selby Jonathan MacDougall

April 22, 2009

Abstract

By harnessing the collective storage capacity and bandwidth of Internet users, peer-to-peer file sharing systems have proved invaluable for storage and distribution of files. However, such systems have a clear and fundamental problem: users have an obvious incentive to download, but no inherent incentive to upload. In naive file sharing protocols, this results in a prisoner's dilemma: the pareto-optimal outcome is for everyone to upload as much as they can, and download as much as they want, resulting in high download rates for all users and the costs involved in uploading evenly distributed across peers; however, any single agent stands to benefit from defection. By reducing uploads, an agent can both improve his download rate and reduce his total upload. This being the case, the Nash equilibrium occurs where users share little, or nothing at all. BitTorrent, a popular and effective peer-to-peer file sharing system, attempts to increase incentives to share in order to achieve, or at least approach, the pareto-optimal outcome. However, analysis and practice have revealed flaws with the existing implementation of BitTorrent, and have suggested methods and strategies which a peer can use to minimize his own uploads without significant reprecussions. We introduce and motivate the BitTorrent protocol, and introduce some of the models that have been used to analyse it. We then examine some of the flaws that have been discovered and exploited. Finally, we introduce some of the strategies and methods that have been proposed to combat the free rider problem, leading users to a more appealing and socially equitable equilibrium.

1 Introduction

Achieving fairness amongst peers in a peer-to-peer setting has proved to be a challenging problem. It has generally been found that agents have very little incentive to upload data to other users, with the result that overall social welfare drops. This problem strikes especially hard at those who do share, since all peers will download from them. One protocol which attempts to remedy this problem is BitTorrent; this is one of the most popular peer-to-peer applications. Our paper aims to survey both the known problems that lead to these inefficiencies and proposed solutions to these problems.

In section 2 we will discuss the motivation behind peer-to-peer networks as well as some of the details of their implementation. This section describes some of the history and context of file sharing leading up to the popularity of BitTorrent. In section 3 we discuss the ways in which the BitTorrent protocol has been modeled as a multi-agent or game-theoretic sense in order to analyze it's strengths and weaknesses. This includes understanding the protocol as a tit-for-tat game (3.2), or viewing it as a series of auctions (3.3). Both models expose different problems and suggest different solutions.

Section 4 looks at problems with fairness in the established protocol. It surveys problems such as Sybil attacks, whitewashing, and cheap pseudonyms (4.2) and optimistic unchoking (4.3). Each of these flaws can be exploited by peers to maximize their downloads and minimize their uploads.

Finally, in section 5, we survey potential solutions to these problems. These involve solutions that are currently used in practice such as private trackers (5.1) as well as some suggested but still academic so-

lutions such as a more strict tit-for-tat strategy (5.2), proportional sharing (5.3) and possibilities of using some form of the VCG mechanism (5.4).

2 Motivation

Computer networks have become vitally important in the distribution of data and information, from simple text to audio, video, and applications. Networking has expanded the uses and functionality of computer systems tremendously, to the point that they have become essential to businesses, governments, and to individuals. Networking is, essentially, the transfer of files from computer system to computer system. For simple, relatively small files, such as HTML files, and for relatively few users, this is trivial and inexpensive. However, when file size or user demand increases, distribution becomes a much more challenging problem.

2.1 Centralized File Sharing

The simplest form of file sharing uses a simple client-server model, in which one or more central servers provides the file or files to all interested clients. Examples of this model include most web servers, mail systems, and file distribution systems like FTP, NFS, and Samba. The same principle is behind media distribution systems like YouTube, Hulu, and the iTunes music store. This model tends to result in systems that are easy to construct, maintain, and control, given an appropriate level of demand. However, the cost and complexity of these systems can increase dramatically as demand or file size grows. Any individual or group wishing to widely distribute files in high demand would quickly find the costs to be prohibitively expensive. Even companies with large resources and experience with large, centralized, high-demand systems can find themselves overwhelmed by hardware, bandwidth, and development costs as demand increases. On the Internet, demand has a tendency to increase exponentially, rather than geometrically, making it very difficult for even the largest organizations to keep pace with demand.

2.2 Traditional Peer-to-Peer Networks

With the appearance of Napster in 1999, peer-to-peer file sharing emerged a powerful new paradigm for distribution of files. It was inspired by literal peer-to-peer file transfers in IRC chat rooms, where individuals would, through word of mouth, locate other users who had the files they desired. Often, such transfers would involve a trade, the files themselves acting as a form of currency. Obviously, this system was cumbersome and slow to use. Napster streamlined this process by allowing users to connect to a centralized server, and search one another's fines directly. When file transfers occured, data was moved directly from one user's system to the other, bypassing the central server, and thus distributing the cost of file hosting. A key revelation here is that files in high demand are (or will quickly become) highly distributed.

Shortly after the appearance of Napster, other peer-to-peer file systems began to appear. Most were modeled closely after Napster, relying on a centralized server. Others, in particular Gnutella, further decentralized the network; peers would simply connect to one another, in liu of a central server, and would pass searches and search results between themselves.

However, these systems share the drawback that they do not provide sufficient incentive for users to contribute files to the network. This is known as the *free rider* problem, and addressing it is essential to the efficient funtionality of peer-to-peer networks. [1] found that 70% of the users on the Gnutella network in 2000 provided no files for download; at the same time, 37% of files available on the network were shared by the top 1% of peers on the network. By 2005, [7] found that the situation had degenerated further, with 85% of the peers on the Gnutella network sharing no files.

This is not surprising, when one considers each peer as a rational, self-interested agent, and the network as a game. Trivially, acquiring new files provides positive utility to agents, while uploading files generally results in negative utility, due to bandwidth costs, opportunity costs of used bandwidth, resource usage and inconvenience, not to mention fear of litiga-

tion in the case of copyright materials. This negative utility may be offset somewhat by an altruistic desire to share, but will generally remain negative, as illustrated by [1, 7]. This problem is exacerbated by the fact that those who do share will end up uploading far more than your fair share (i.e. far more than you download) because so many other peers share nothing. At the same time, most of these networks do not punish freeloaders. This being the case, the dominant strategy for each agent is to download without sharing.

2.3 BitTorrent

BitTorrent was created in 2001 by Bram Cohen to combat the problems with traditional peer-to-peer systems, in particular the free rider problem. Unlike the earlier systems, BitTorrent allows users to trade pieces of a single file. It does not provide a file indexing or search mechanism, leaving that to other applications or web search engines; instead it focuses on allowing a group, or 'swarm', of peers to simultaneously download a file or files desired by all of them, sharing the cost of uploading. As a peer gets pieces of the files, it makes them available to other peers. In return, those peers provide the pieces they have. When a peer has all pieces of a file, they can assemble it into the complete file, at which point they can choose whether to continue sharing.

BitTorrent 'swarms' are made up of peers, and a tracker, which simply provides a list of all peers. To initiate a file download, a peer downloads this list from the tracker, then contacts some of the peers and begins to download pieces of the file; the download proceeds as described above. Critically, BitTorrent uses a *tit-for-tat* approach to sharing, which provides incentive for peers to share. Each peer will prefer to upload to peers that have given more data to them; if a connected peer stops sharing, the connection will be *choked*, or constricted, until they increase their rate of uploading. Thus, the more a peer shares, the faster it will acquire new pieces. Ideally, peers who refuse to share will find it impossible, or at least significantly slower, to download a file.

However, perfect tit-for-tat sharing is impossible, because new peers joining the swarm will have no pieces to share. Therefore, *optimistic unchoking* is used to randomly unchoke some connections, allowing peers to gain more pieces, which they can then use to acquire still more pieces. Also, peers who have a complete file no longer need to trade for new pieces. It is easy to imagine a swarm where such peers uniquely hold some pieces of a file; in such swarms, the peers with incomplete copies of the file (known as *leachers*) must rely on the peers with complete copies (or *seeders*) to share these pieces, despite the fact that they have nothing to gain by doing so.

BitTorrent has proved to be a very successful. According to [12], BitTorrent traffic accounted for more than 30% of all Internet traffic and 53% of all peer-to-peer traffic in 2005.

3 Models of BitTorrent Swarms

3.1 Model Assumptions

Throughout this paper, we will use the following general model for an agent's utility for downloading a file (where it is assumed than *some* uploading is required):

$$u_i = \left[f_i^{content} + f_i^{speed} + f_i^{altruism}\right] - \left[f_i^{bandwidth} + f_i^{resources}\right]$$

Where:

- $f_i^{content}$ represents the utility the agent will gain by acquiring the file, or the file's inherent utility; this will generally be constant, though in some cases, external factors may affect it (for example, timely content may decrease in value over time)
- f_i^{speed} represents the speed at which the download is completed
- $f_i^{altruism}$ represents any utilility gained though generosity
- $f_i^{bandwidth}$ represents the cost of bandwidth, both in terms of monetary costs and opportunity costs, along with any other related costs

- $f_i^{resources}$ represents any local computer resources, such as RAM, CPU, or hard drive space, used in sharing the file
- $f_i^{neg} = f_i^{bandwidth} + f_i^{resources}$

3.2 Tit-for-Tat

The most simple model for analyzing BitTorrent networks, and the model used in it's creation, is the *tit-for-tat* model. Strict tit-for-tat would imply that you trade with other peers on a one-to-one basis, giving them one piece of the file for each piece they give you. For the reasons given in Section 2.3, BitTorrent couldn't use this model without some modifications, as noted in [9]. Instead, BitTorrent uses a more flexible approach, described in [4], which still closely resembles the tit-for-tat model. In addition to trading with peers in a tit-for-tat way, a peer will also 'optimistically unchoke' other peers, meaning that it will increase it's upload to one of it's connected peers. This helps newcomers, who can begin to gather pieces. It also allows peers to discover better connections, since the other agent will also unchoke the providing peer. If the newly unchoked peer provides better upload speeds than the existing uploaders, the peer will continue to upload to this new peer and 'choke' (stop uploading) the slowest peer.

3.3 BitTorrent as an Auction

According to [10], it is useful to model BitTorrent as an auction. In a sense, this models emphasizes a different stage of interaction. Once a connection is established, it is roughly tit-for-tat; however, this doesn't address the critical question of how peers are chosen and connections established. Levin observes that the method of chosing peers to unchoke more closely resembles a two-stage auction than a simple tit-for-tat model. The two stages are:

1. Assuming a peer wishes to establish n connections, it gathers bids from other peers for available connections (where each bid is considered to be the amount of bandwidth each other agent can commit to the connection, b_j).
2. The peer selects the top $(n-1)$ bids, and establish a connection with each of the winning peers, committing $1/n^{th}$ of his bandwidth to each connection. The last connection is reserved for a random peer, for optimistic unchoking.

The ideal strategy in an iterated tit-for-tat game is cooperation; if an agent refuses to cooperate, other agents will also refuse to cooperate with him, and his expected utility will decrease. If BitTorrent were truly a tit-for-tat game, cooperation should be the dominant strategy. Viewing BitTorrent as an auction, however, suggests another strategy, based on the observation that a peer need only be the *lowest winning bidder* in order to establish a connection. This being the case, a peer may begin with a very low bid to another peer j, and slowly increase it until it becomes a winner in the auction, thus establishing a connection to j with the lowest possible bid. Using this strategy, a peer can decrease the amount of data it must upload without significantly increasing the time to download. This is the approach used by BitTyrant [13], a BitTorrent client developed to illustrate this strategy in action. If all nodes in a swarm used this strategy, the game would quickly become degenerate; since bidding high is a dominated strategy, every peer would try to be the lowest winning bidder, and the bids would get lower (and thus the downloads would get slower) with each iteration.

The auction model, however, suggests a dominant-strategy truthful auction to avoid this problem, which we discuss in section 5.3.

3.4 Externalities

There are several external factors which will affect a peer's decision on whether to share and how much he wishes to share:

Responsibility In many cases, a peer may be considered responsible for the contents of files which he provides to other peers. Examples include copyrighted material, politically sensitive material, potential malware, and any unverified material. This might put him at risk of legal, political, or other reprecussions.

Reputation In most peer-to-peer networks, reputation does not play a significant role; however, in some cases, it has a significant impact on the willingness of peers to share files. Examples include IRC file sharing, which relied largely on reputation, and private, registered BitTorrent trackers which track each peer's aggregate total uploads and downloads, in some cases culling contributers whose social contribution is low, and in other cases rewarding those with the highest social contribution.

4 Fairness Issues with BitTorrent

4.1 Dealing with Newcomers

The special case for newcomers might seem unfair to an established user. A newcomer is able to receive file pieces without contribution. This is a clear avenue for attack. So-called white washing is when a user repeatedly becomes a newcomer and, as such, does not have to contribute to the swarm. A user implementing this strategy becomes a free-rider. It is difficult to distinguish whitewashing users from legitimate newcomers but some strategies have been proposed such as unique node ID's [3] or simply punishing all newcomers. In practice with a traditional client, BitTorrent punishes newcomer such that they have a slower download rate. This does not appear to have been done explicitly but is simply a result of the optimistic unchoking' protocol. It is possible to modify a client to take better advantage of this exploit as shown in section **??**.

Sybil attacks can also be implemented on a system where new identities are easily created. A sybil attack is where users are able to collude with a number of falsely created identities to increase there individual download utility.

4.2 Cheap Identities: Whitewashing, Sybil Attacks & Collusion

The special case for newcomers might seem unfair to an established user. A newcomer is able to receive file pieces without contribution. This is a clear avenue for attack. So-called 'whitewashing' is when a user repeatedly becomes a newcomer and, as such, does not have to contribute to the swarm. A user implementing this strategy becomes a free-rider. It is difficult to distinguish whitewashing users from legitimate newcomers but some strategies have been proposed such as unique node ID's [3] or simply punishing all newcomers. In practice, BitTorrent punishes newcomer such that they have a slower download rate. This does not appear to have been done explicitly but is simply a result of the 'optimistic unchoking' protocol.

Sybil attacks can also be implemented on a system where new identities are easily created. A sybil attack is where users are able to collude with a number of falsely created identities to increase there individual download utility.

4.3 Optimistic Unchoking Exploits

As [4] has shown, optimistic unchoking is vital to the performance of bittorrent as it is currently constructed. It allows users to connect to peers that could potentially provide a higher download speed. These peers would go missed if a simple choke/unchoke model were used. Yet this property can be exploited to allow for free riders. As has been shown in [14, 15] one can achieve at least as good and in some cases better download rates by taking advantage of optimistic unchoking. This leads to a higher utility for agents as f_i^{speed} increases while $f_i^{bandwidth}$ decreases. Essentially, a user is able to implement a sybil attack making it more likely that other peers will optimistically unchoke the offending peer. They see more copies of this peer and, as such, have a higher likely-hood to unchoke them. This unchoking may only last for 30 seconds (the bittorrent default for optimistic unchoking) but the user is able to connect to so many other peers via this method that they are still able to maintain a decent download rate. As well as this an agent can take advantage of the so called large-view problem[15]. This exploit is when an agent to attempts to connect to as many peers as it can and thereby cause more optimistic unchoking to be directed toward said agent. In conjunction with these methods, users are able to selectively connect to

seeds which do not require any reciprocity to further enhance there download speed as seen in [11].

4.4 Rare Pieces

A further special case in bittorrent is rare pieces. To ensure the health of a torrent file, that is, prevent file piece disparity, bittorrent prioritizes the download of rare pieces first. The idea is that as more users download rare pieces they become less rare thereby eliminating disparity. The importance of this idea is further explained in [4]. Another benefit of this is that a single seed will quickly distribute every piece of the file to peers. This allows the peer to leave the swarm as early as possible as it quickly builds a distributed copy of the file. Yet this prioritization can also be exploited as show in [11]. Users can falsely advertise the ownership of rare file blocks in an effort to solicit more connections. These extra connections will then allow for a higher instance of optimistic unchoking and therefore allow the offending peer to lower there $f_i^{bandwidth}$cost with no penalty to f_i^{speed}.

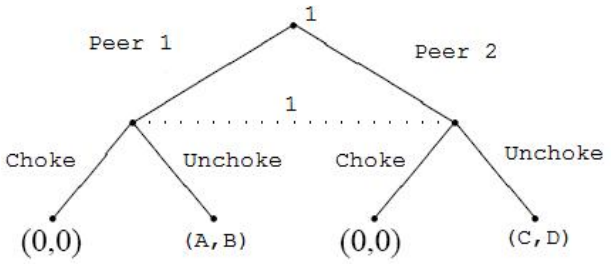

Figure 1: Imperfect information extended form game for optimistic unchoking in a situation with no known agent reputation. Say peer 1 misbehaves and peer 2 behaves, then $a < 0$ as the selecting user will upload to peer 1 without downloading while $b >$ c and d as peer 1 receives a file while not having to upload to the selecting peer. The values of c and d will be approximately equal and greater than 0 as the peers are behaving as expected.

5 Survey of Potential Solutions

5.1 Reputation, Private Trackers & Tracker Incentives

Most of the problems iterated in section 4 align with the fact that agents generally maintain anonymity from file to file. There is no way to track which agents misbehave and which behave in a social responsible way. Each file can be thought of as an imperfect information extensive form game. A node represents selecting a peer to optimistically unchoke. This selection is an equivalence class between all peers as there of telling the difference between a misbehaving peer and a responsible peer as shown in figure 1. The only rational strategy in peer selection is evenly mixing between all peers as they all seem equal. A simple way to remove this equivalence class is by tracking a reputation of an agent between file transfers. This is actually used in practice through the idea of private trackers. These trackers are maintained by a community and one must hold an account to access them. They track the amount of data uploaded as well as content added to the community and will kick users out if they fail to maintain a certain minimum. This reputation tracking eliminates free riding as there behavior will identify them and they will not be able to use the service. Private trackers tend to offer better or harder to find content than that found on public trackers. This further provides incentive to join the community and maintain a positive reputation.

Private trackers are a decent solution to many of the problems in bittorrent but suffer from two drawbacks. The first is there exclusivity. An agent must become a member of the community to participate and often this can be challenging. Many of these trackers only offer accounts through an invitation process. Others offer a simple sign up with a unique e-mail address. The extra steps needed to become a member often limit the user base. The more crucial drawback is centralization. A central authority is required to track reputation. One of the main goals of peer to peer applications is to become as decentralized as possible.

5.2 Strict Tit-for-Tat

In [8], the authors suggest moving BitTorrent to a more strict tit-for-tat model, inspired by the iterated Prisoner's Dilemma tournaments analyzed in [2]. At present, the tit-for-tat model is relaxed in order to improve performance, and allow newcomers to join the swarm; however, as described in section 3.3, some of these relaxations in the model result in potential exploits; peers can form connections in which they download more than they upload; it is thus clear that the peers are not using strict tit-for-tat strategies.

In Jun's proposed model, a peer i will establish connections to another peer j, and will continue to upload as long as:

$$u_{ij} - d_{ij} \leq f \cdot c$$

Where u_{ij} is the amount of data uploaded to j, d_{ij} is the amount of data downloaded from j, c is the size of the data blocks being transfered, and $f \geq 1$ is peer i's *niceness factor*. Thus, peer i will continue to upload until the *deficit* of the link $(u_{ij} - d_{ij})$ exceeds the maximum deficit that i is willing to accept $(f \cdot c)$. If agent j stops uploading at any point, i will continue to upload only as long as this property remains true, and then halt. If j begins to upload again, i will also resume uploading. This closely follows the tit-for-tat strategy, in which agents forgive other agents if they begin to cooperate again (i.e. they choose the action which the other agent chose on the previous iteration of the game).

Jun et al. tested this model to determine it's effectiveness by downloading a 33 MB file using a swarm exclusively consisting of conventional BitTorrent peers, and another swarm containing only peers modified to use the tit-for-tat strategy described above. They found that the new model had the following properties:

- Average completion time was significantly longer for the modified tit-for-tat client:
  - Tit-for-tat swarm (s): 1672 mean, 1441 median
  - Conventional swarm (s): 1123 mean, 1139 median
- The range of upload speeds was significantly smaller in the tit-for-tat swarm: $[1.62, 30.65]$ versus $[0.78, 66.56]$
- The largest *total deficit*, defined as $\sum_j (u_{ij} - d_{ij})$ for all clients j connected to i, is much greater in the conventional swarm (-32 MB, meaning less than 1 megabyte uploaded) than for the tit-for-tat swarm (-9 MB). In the tit-for-tat swarm, for any given peer i, the maximum deficit is bounded by $\sum_j (f_i \cdot c)$.
- In the conventional swarm, all clients finished downloading the complete file in roughly the same amount of time, with no significant correlation between upload rate and completion time. In the tit-for-tat swarm, there was a strong correlation between rate of upload and the time to completion; however, the fastest peers still generally took longer than the average peer in the conventional swarm.

Generally speaking, the conventional swarm outperformed the tit-for-tat swarm, the only exceptional metrics being the maximum deficit and maximum upload rate, where the worst case was much improved in the tit-for-tat swarm.

The other main argument for using this strategy is that it strongly discourages free riding. Free riding peers in the tit-for-tat swarm took much longer to complete the file than peers who uploaded in proportion with their downloads. The authors argue that in the conventional swarm, peers are motivated to minimize their deficit; they can minimize both of the terms in f_i^{neg} (see Section 3.1) while, as their tests illustrated, f_i^{speed} remains relatively constant. In the tit-for-tat swarm, however, f_i^{speed} will decrease proportionally as $f_i^{bandwidth}$ and $f_i^{resources}$ decrease. They argue that this punishment to free riding makes the tit-for-tat swarm more robust, and less likely to degenerate.

5.3 Proportional Sharing

The analysis performed by Levin et al., introduced it Section 3.3, and the BitTyrant client software ana-

lyzed in [13], point out a flaw in BitTorrent; in terms of the auction model, a peer need only submit the *lowest winning bid* to form a connection with another agent. This motivates peers to attempt to find the lowest bid possible which will still cause the remote peer to upload to them. The tit-for-tat strategy described in 5.2 combats this problem by keeping track of the *deficit* of each connection, and choking the connection when the deficit exceeds an acceptable amount; this punishes free riders, but also punishes everyone, since it increases the average download speed relative to the conventional client. In [10], the authors suggest an alternative strategy.

The authors suggest changing the auction model discussed in 3.3 to the following:

1. Assuming a peer wishes to establish n connections, it gathers bids from other peers N for available connections (where each bid is considered to be the amount of bandwidth each other agent can commit to the connection, b_j).

2. Letting B_i represent i's total available bandwidth, give each peer $j \in N$ bandwidth proportional to his bid:

$$b_i^j = B_i \cdot \frac{b_j(n-1)}{\sum_{k \in N} b_i^k (n-1)}$$

The authors observe that a peer in a swarm of peers playing this strategy who wishes to maximize his download speed will have incentive to maximize his upload. They found that, in a perfect-information swarm, the best response to other peers playing the prop-share strategy is *not* to play prop-share in response; however, their tests indicated that download time using prop-share was consistantly within 1% of playing the best response. Since most swarms are not perfect-information environments, and prop-share requires only known information, most agents should choose to use the prop-share strategy in a swarm of prop-sharing peers.

Proportional sharing also has the following interesting properties:

- It is resistant to Sybil attacks: Since the attacker must divide up his own bandwidth among the Sybils, and the victim of the attack will respond by uploading in proportion to the upload from each Sybil, the total download rate will be at best as good as if the attacker had simply uploaded as much as possible from one client. In practice, Sybil attacks will perform significantly worse, due to additional overhead and the fact that some Sybils may fail to meet the victim's minimum bid.

- It is collusion-resistant: If many attackers A collude by uploading a small (and, we will assume for simplicity, equal) amount to a victim i, then i will divide up his bandwidth between them, resulting in i giving each of the attackers $B_i \left(\frac{1}{|A|}\right)$, where B_i is i's total available bandwidth, even though $\sum_{a \in A} b_a^i$ might be significantly less than B_i. However, if a single new peer j connects to i, or any one of the colluders defects and begins to upload more, they could recieve a very large return on investment, while significantly reducing the bandwidth remaining for the coalition A, $B_i - b_j^i$. At this point, the coalition can either accept much smaller returns (more proportional to their uploads), or defect and upload more, thus gaining higher (but still more proportional) returns. Regardless, the coalition's effectiveness is drastically reduced, to the point that it may be considered to be dissolved, by the actions of a single agent $j \notin A$.

- In a swarm in which all peers use the prop-transfer strategy, the total rate of transfer and the speed at which all agents complete their downloads, both of which are a good indication of social welfare, are higher than they would be under the strict tit-for-tat strategy (5.2), since all agents choose to upload at their maximum bandwidth.

There remains the problem of how to bootstrap new peers in a swarm, since they will have no data to upload in order to gain a portion of any existing peer's bandwidth. Ideally, the method used should not be exploitable by the *large view* exploit (see **??**), in which peers can avoid uploading by connecting to

many peers, always pretending to be a newcomer. The authors propose a method whereby a new peer i is given a block of data encrypted with a symmetric key, which is then passed to a third peer j, who desires the same data. Only by uploading to m can i get the key, at which point both m and i can decrypt the data. This data can then be used to establish new connections.

Since peers have to upload any data they download while claiming to be a new peer, they no longer have any incentive to claim to be new when they have data available to trade. It is possible to imagine non-trivial joint attacks to avoid this shortcoming (in which m and i are collaborators); additionally, existing peers may choose to ignore new peers because of the complexity involved in bootstrapping them. These concerns are not addressed in the paper.

5.4 VCG Mechanisms

At first glance bittorrent appears to be the perfect candidate for mechanism design, but on closer inspection it becomes clear that this is a very difficult problem. Ideally, one would want the BitTorrent mechanism to follow VCG so as to elicit Pareto-efficiency and truthfulness in dominant strategies. Unfortunately BitTorrent is a distributed platform, which makes implementing VCG quite challenging; there is no centralized peer on which to implement the mechanism. Efforts have been made to distribute VCG by [5] in what is called DAMD, or distributed algorithmic mechanism design. Work in this area has not yet been directly applied to BitTorrent, focusing instead on routing mechanisms, but it was proposed in [6] that this work may be adapted, leading to an effective, distributed implementation of VCG for BitTorrent.

6 Conclusion

Peer-to-peer networks have already become one of the most important forms of network traffic on the Internet, and in the future they may play a still larger role. They allow users to widely distribute large files, while requiring only a fraction of the resources that would be required to achieve similar results using traditional client-server architectures, by distributing the costs among peers. However, traditional peer-to-peer networks have serious shortcomings, since the peers involved tend to behave as self-interested agents, and thus avoid sharing and the costs involved while continuing to download. The few incentives to share were overwhelmed by the cost of doing so, leading to the disappointing results found in [1, 7].

BitTorrent was the first significant peer-to-peer system to attempt to address this shortcoming by providing incentive to share through a simple tit-for-tat model. It proved to be a tremendously successful system, and to this day it accounts for a large portion of all Internet traffic. In the years since it's release, however, several flaws have been found and exploited in the protocol. As a result, peers are able to achieve similar download rates and speeds without contributing in kind. This may cause BitTorrent swarms to degenerate, as more and more peers chose to act as *free riders*. We introduced and discussed some of the most promising methods and strategies that have been put forward to combat this trend: reputation-based systems which track aggregate contributions over time, strict tit-for-tat strategies that accept slower overall downloads in exchange for fairness and the ability to punish free riders, and proportional sharing, which strives to punish free riders and improve the fairness of the system without significantly decreasing the overall social welfare of the system. There remains the possibility that some form of distributed VCG mechanism may prove important in this area, but as of yet no such mechanism has been presented.

References

[1] Eytan Adar and Bernardo A. Huberman. Free riding on Gnutella. *First Monday*, 5(10), October 2000.

[2] Robert Axelrod. *The evolution of cooperation.* Basic Books, New York, 1984.

[3] M. Castro, P. Druschel, A. Ganesh, A. Rowstron, and D. Wallach. Secure routing for

structured peer-to-peer overlay networks. In *Fifth Symposium on Operating Systems Design and Implementation (OSDI '02)*, Boston, Massachusetts, December 2002.

[4] Bram Cohen. Incentives build robustness in bittorrent, 2003.

[5] Joan Feigenbaum and Scott Shenker. Distributed algorithmic mechanism design: Recent results and future directions, 2002.

[6] Michal Feldman and John Chuang. Overcoming free-riding behavior in peer-to-peer systems. *SIGecom Exch.*, 5(4):41–50, 2005.

[7] Daniel Hughes, Geoff Coulson, and James Walkerdine. Free riding on gnutella revisited: The bell tolls? *IEEE Distributed Systems Online*, 6(6):1, 2005.

[8] Seung Jun and Mustaque Ahamad. Incentives in bittorrent induce free riding. In *P2PECON '05: Proceedings of the 2005 ACM SIGCOMM workshop on Economics of peer-to-peer systems*, pages 116–121, New York, NY, USA, 2005. ACM.

[9] R. LeMay. Bittorrent creator slams microsoft's methods, June 2005.

[10] Dave Levin, Katrina LaCurts, Neil Spring, and Bobby Bhattacharjee. Bittorrent is an auction: analyzing and improving bittorrent's incentives. In *SIGCOMM '08: Proceedings of the ACM SIGCOMM 2008 conference on Data communication*, pages 243–254, New York, NY, USA, 2008. ACM.

[11] Nikitas Liogkas, Robert Nelson, Eddie Kohler, and Lixia Zhang. Exploiting bittorrent for fun (but not profit), 2006.

[12] A. Parker. The true picture of peer-to-peer filesharing, July 2004. `http://www.cachelogic.com/`.

[13] Michael Piatek, Tomas Isdal, Thomas Anderson, Arvind Krishnamurthy, and Arun Venkataramani. Do incentives build robustness in bittorrent? In *NSDI'07*, Cambridge, MA, April 2007.

[14] J. Shneidman, D. Parkes, and L. Massoulie. Faithfulness in internet algorithms. In *Proc. SIGCOMM Workshop on Practice and Theory of Incentives and Game Theory in Networked Systems (PINS'04)*, Portland, OR, USA, September 2004. ACM SIGCOMM.

[15] Michael Sirivianos, Jong Han, Park Rex, and Chen Xiaowei Yang. Free-riding in bittorrent networks with the large view exploit. In *In Proc. of IPTPS 07*, 2007.