Adding Incentives to File-Sharing Systems

Aviv Zohar
School of Engineering and Computer Science
The Hebrew University of Jerusalem
Jerusalem, Israel and
Microsoft Israel R&D Center, Herzlia, Israel
avivz@cs.huji.ac.il

Jeffrey S. Rosenschein
School of Engineering and Computer Science
The Hebrew University of Jerusalem
Jerusalem, Israel
jeff@cs.huji.ac.il

ABSTRACT

Modern peer-to-peer file sharing systems rely heavily on the willingness of users to distribute files to others. A selfish user can choose to download a file and consume resources without uploading in return. This form of free-riding plagues all currently deployed peer-to-peer systems.

We present a novel protocol for a BitTorrent-like system (i.e., one in which only one file is being shared) that strongly discourages users from downloading a file without sharing it. Our protocol requires very little computation, and can easily be implemented in today's peer-to-peer systems. It is resistant to all forms of manipulation, including the use of multiple free identities, and does not require any coordination among seeds. We analyze our protocol and show that if downloading peers are rational, a new system equilibrium is reached in which all peers upload at least some percentage of the file they are given.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Multiagent Systems*; C.2.2 [**Computer-Communication Networks**]: Network Protocols—*Applications*

General Terms

Algorithms, Design, Economics

Keywords

File Sharing, Incentives, Peer-to-Peer, BitTorrent

1. INTRODUCTION

Before peer-to-peer file sharing systems appeared, large-scale content distribution on the internet was a costly affair. Purchasing bandwidth and storage space for the dissemination of content that could potentially be downloaded by millions of users world-wide put a strain on content providers.

Peer-to-peer file sharing systems shifted some of the cost of distribution to the consumer of information. In addition to downloading files for his own use, a downloader (or peer, as he is usually called in peer-to-peer systems) is also required to upload pieces of the file he has obtained to his peers. The result is a system in which the original provider of content need only distribute as little as a single copy of the file (possibly in pieces) to others, who then share the different pieces among themselves.

Since the early days of file sharing systems, it has been apparent that some users are not eager to take up the burden of distributing files to others [1]. Users may decide not to upload for many reasons: some may prefer to reserve their upload bandwidth for other traffic that they deem more important, others are unable to offer high upload speeds due to asymmetric internet connections, and some are downloading illegal content and wish to avoid prosecution. While the protocols of file sharing systems require everyone to upload pieces of the file to others, no mechanism has been put into place that enforces this securely. Since users run their own copy of the peer-to-peer client, they are able to modify it or choose an alternate client so as to avoid uploading to other peers. Those who choose not to upload are consuming resources without contributing in return, and are thus free-riding at the expense of others.

In this paper, we propose a file sharing protocol that provably changes the incentives to share files in the system. Our protocol motivates peers to share a certain portion of the file they are downloading even before they complete the download. In this manner we support the main goal of peer-to-peer file sharing: to shift much of the cost of content distribution to the downloading peers. Our protocol resists manipulation by rational peers that seek to maximize the use of their upload and download resources, and guarantees a more egalitarian distribution of costs in the system. In our suggested mechanism, peers cannot use multiple identities (which we assume are created effortlessly) in order to alleviate the high costs associated with not sharing. Furthermore, peers cannot compensate for the losses they incur that arise from their sharing less, by establishing a large number of links to others. The implementation of our mechanism is quite straightforward, and requires no complex interaction or coordination among uploading seeds. The resulting system is slightly less robust to the failure of all uploading peers, but we offer several mechanisms that mitigate this deficiency. We also suggest a model of costs for the peer-to-peer network, and analyze our protocol within it.

Our approach does not assume the existence of a monetary or reputation system that keeps track of the behavior of peers and that can be used to reward or punish them. Such systems usually have to be centralized in order to be reliable (and thus constitute a possible point of failure for the entire file-sharing network), or prove to be hard to scale up. To the

Cite as: Adding Incentives to File-Sharing Systems, Aviv Zohar and Jeffrey S. Rosenschein, *Proc. of 8th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2009)*, Decker, Sichman, Sierra and Castelfranchi (eds.), May, 10–15, 2009, Budapest, Hungary, pp. XXX-XXX.

best of our knowledge, there is currently no deployed system that enables micro-payments or reputation aggregation at the scale needed for file-sharing networks.

1.1 Related Work

An important previous step towards installing proper incentives for uploading was taken with the creation of the BitTorrent protocol [3]—currently the most commonly used protocol for file sharing [10]. In BitTorrent, a group of peers shares only a single file and locates one another through a small-scale centralized server that is called a *tracker*. Peers that have a full copy of the file and are no longer downloading are called *seeds*, and are expected to remain online and donate pieces of the file to their peers in order to make sure that a full copy of the file exists. The novelty in BitTorrent is in the interaction between downloading peers, who trade pieces of the file being downloaded. When trading with others, peers attempt to reciprocate and upload more to peers who have given them more. This strategy, which operates according to the principles of "Tit-for-Tat", rewards uploaders with more pieces of the file and punishes peers who do not upload, giving them an overall lower download rate.

Despite attempts at hurting free-riders, experiments conducted with the BitTorrent protocol show that it is not resistant to protocol manipulation by selfish users. The BitThief client [14] demonstrates that files can be obtained quickly even without uploading. While BitThief does not upload at all, it still enjoys downloads from seeds (who give out information freely—since they do not download the file themselves) and some limited amount of download from other peers, that still offer some information before they discover none is being sent in return. A similar approach is taken up in the *Large-View exploit* [15], where a modified BitTorrent client maximizes its download rate by simultaneously opening links to many seeds and peers, thereby multiplying the small download rates from each link by a large factor. Our own mechanism operates in an environment similar to that of BitTorrent (we similarly assume that only one file is being shared) and is resistant to the Large-View exploit.

There is a large body of work on incentives in peer-to-peer systems (see [4] for a broader overview). Previous work has often assumed the existence of a monetary system [9] that assists in rewarding contributing users, or a reliable reputation system [2] that keeps tally of the contribution of each peer and helps decide who gets more of the system's resources. Others use different models for peer-to-peer systems and free-riding behavior, including the use of population dynamics [5, 7, 13]. In [12], BitTorrent's trading mechanism is analyzed as an auction, and some modifications to its trading policies are suggested. In this paper, we offer an alternative trading mechanism but, more importantly, we also suggest changes to the upload protocol used by seeds that upload to others for free.

In [6], file sharing is modeled as a social phenomenon—users consider whether or not to contribute files based on the number of other users who contribute. A mechanism is added to impose a penalty on users entering the system to discourage them from leaving and then rejoining with a new identity (and thus dodge any record of past behavior).

1.2 Structure of the Paper

In Section 2 we introduce our model for the costs associated with sharing files. In Section 3 we discuss a first version of our suggested protocol for file sharing, and analyze the incentive structure it imposes. In Section 4 we suggest a more developed protocol that distributes files more efficiently and analyze it as well. We then present, in Section 5, a strategy for fairly and equally trading files between two peers, and show additional improvements to our protocol. In Section 6, we discuss open problems and future work.

2. MODELING COSTS

We model a BitTorrent-like system where participants belong to one of two groups: seeds and peers. Seeds are nodes that have a full copy of the file being shared and are offering it to others, without expecting anything in return. Peers are nodes that do not have the complete file, and are trying to obtain missing pieces. Our model treats only the peers as rational players (there is no way to provide motivation for seeds to upload in a single-file model without payments).

We assume that the file being shared is partitioned into r regions. Each region is further partitioned into m blocks. Blocks can be sent and received separately by the peers.

We assume that downloading and uploading blocks of data consumes resources from both downloaders and uploaders (bandwidth that must be purchased, time to download, etc.). We denote the cost of downloading a single block as C_{dl} and the cost of uploading a block as C_{up}. Consequently, the cost that *must* be paid by a peer to download the entire file is at least the cost of downloading all the blocks: $r \cdot m \cdot C_{dl}$. We model peers as rational players who want to obtain all the blocks belonging to a file while paying as little as possible for resources expended in the process.

Peers are assumed to have the ability to connect anonymously to seeds and to other peers—as many times as they want, and without any significant overhead. We further assume that different peers (or even the different identities assumed by the same peer) are indistinguishable, and so no information regarding the past behavior of a specific peer can be maintained from one session to another.

Our model of costs for upload and download of a single block fits several common scenarios. The first is one in which customers pay a flat rate for a constant amount of bandwidth purchased from a service provider (this is common in many broadband connections). In this case, the cost of downloading a single block can be thought of as the amount of time it took to download this block at the given bandwidth. A downloader who minimizes his total cost will minimize the time it takes to obtain the full file, thus freeing up download and upload bandwidth for use by other applications.

Another common scenario is one in which users pay according to the amount of data they send or receive. This is quite common, for example, in the connectivity packages that cell phone companies sell for third generation cell phones. In this case, the cost of downloading or uploading can be measured in actual currency.

The main difference between our model and previous models that have considered the costs of download or upload is our assumption that if a portion of the bandwidth of a peer is not being used to download at a certain moment in time, that "resource" is not lost and is probably used to download another file, or for some other purpose. To truly hurt a misbehaving peer, one has to waste the resources of that peer by making it download or upload excessively.[1] Therefore,

[1] Merely slowing the rate of transfer to a misbehaving peer is

when computing the total costs paid by each participant, we only count downloaded and uploaded blocks.

To simplify the analysis of our protocol for the time being, we add the assumption that peers can trade blocks (1-for-1) if they wish to do so, and that trading is performed securely, so that both parties involved in the transaction are unable to get a block without giving one in return. We shall later show that this can be achieved in an efficient manner.

3. THE FILE SHARING PROTOCOL

The basis of our approach is as follows. When a peer connects to a seed it will be allowed to specify a region of the desired file from which it wishes to receive blocks. The seeds will not grant requests for specific blocks, but will instead randomly select blocks from that region and send them to the requesting peer. In this manner, a peer who attempts to contact any seed and request blocks from that region a second time will receive (by chance) some of the blocks it already owns.[2] This means that the peer downloads new blocks at a higher effective cost if he insists on downloading from seeds. As an alternative, the peer will be able to request *specific* blocks that he is missing from other peers, but only in exchange for specific blocks that they will request. This is indeed more costly than downloading for free (since upload is required in return) but may be worthwhile if the effective cost of downloading the same data from seeds is higher. We shall say that a block has been *covered* if it was sent at least once to the peer, and that a block *collides* if it has already been downloaded once, and is downloaded again. While a misbehaving peer may experience many such collisions, a compliant peer will suffer none, and the overall efficiency of the system will not be damaged, so long as all peers act rationally.

We assume that seeds wish to donate a fraction of the file, n/q, such that q is a prime number or a prime power (i.e., if p is a prime then q can be one of $p, p^2, p^3, \ldots$). We will later describe the exact fraction n/q that can be shared without violating incentive compatibility (see Theorem 3.2). We also assume that there exists an integer d such that $m = q^d$.

Let $\mathbb{F}_q$ be the finite field with q elements. Let V be the vector space $(\mathbb{F}_q)^d$ (of dimension d) over the field $\mathbb{F}_q$. We assign to all blocks within a region an address that is expressed in base q: $(a_1, \ldots, a_d) \in V$ where $\forall i$, we have $a_i \in \mathbb{F}_q$.

Protocol I:

1. Receive a request from a peer for blocks from some region.
2. Randomly and uniformly select a vector $\vec{s} \in V \setminus \{\vec{0}\}$.
3. Randomly and uniformly select n values $Vals = \{b_1, \ldots, b_n\}$ such that $b_i \in \mathbb{F}_q$ without repetitions: $i \neq j \rightarrow b_i \neq b_j$.
4. Send to the peer all blocks in the requested region that have an address $(a_1, \ldots, a_d)$ for which $\sum_{i=1}^{d} s_i \cdot a_i \in Vals$ where this calculation is performed in the field $\mathbb{F}_q$.

Note that the addresses selected in the protocol reside on a set of hyperplanes described by $\vec{s} \cdot \vec{a} = b$ for various values $b \in Vals$. This will be a key ingredient in our analysis, since the intersection of two hyperplanes in a finite vector space is of known size.

3.1 Analysis of Protocol I

We begin our analysis of the protocol by showing the number of blocks that are transmitted over one session.

LEMMA 3.1. *Exactly $\frac{n}{q}$ of the blocks in the region are sent to the peer in one session.*

PROOF. All blocks that are sent to the peer have an address $(a_1, \ldots, a_d)$ for which $\sum_{i=1}^{d} s_i \cdot a_i$ attains some value in the field $\mathbb{F}_q$. The number of addresses that give each value is the same—each value defines a similar but parallel hyperplane, and each hyperplane covers $1/q$ of all addresses. Since we select n values without repetition, we cover an n/q fraction. □

We now present our main result: if the values of n, q in the protocol are set correctly, rational peers will only access a seed once per region, and will trade for the rest of the blocks in that region.

THEOREM 3.2. *A rational peer that is offered a fraction $\frac{n}{q} \geq \frac{C_{up}}{C_{dl}+C_{up}}$ of the region according to Protocol I will prefer to trade the rest of the blocks with other peers, rather than downloading again from a seed.*

The theorem implies that any rational peer can be incentivized to upload (during trade) a portion that is equal to $1 - \frac{n}{q}$ of the file. For example, if $C_{dl} = C_{up}$, peers can be made to upload one half of the file.

PROOF. First, we denote by k the number of blocks being sent at every phase by the seed:

$$k = \frac{n}{q} \cdot m \tag{1}$$

In order to prove the theorem, we shall require the following lemma regarding the number of blocks that collide in expectation:

LEMMA 3.3. *A peer that downloads pieces of the same region twice from some seed, will have in expectation $\frac{k^2}{m}$ collisions.*

The lemma will be proven below.

The cost of getting k blocks from a seed, and then trading for the rest, is:

$$\begin{aligned} Cost_1 &= k \cdot C_{dl} + (m-k) \cdot (C_{up} + C_{dl}) \\ &= m \cdot C_{dl} + (m-k) \cdot C_{up} \end{aligned} \tag{2}$$

According to Lemma 3.3, when a peer downloads twice from a seed the expected number of collisions is $\frac{k^2}{m}$. Therefore, the expected cost of downloading twice and then trading to obtain the rest is:

$$Cost_2 = 2k \cdot C_{dl} + \left(m - 2 \cdot k + \frac{k^2}{m}\right) \cdot (C_{up} + C_{dl}) \tag{3}$$

The difference in costs is:

$$Cost_2 - Cost_1 = \frac{k^2}{m} \cdot C_{dl} + \left(\frac{k^2}{m} - k\right) \cdot C_{up} \tag{4}$$

For this difference to be positive, we must have:

$$k \cdot C_{dl} + k \cdot C_{up} - m \cdot C_{up} \quad > \quad 0 \tag{5}$$

$$k > \frac{C_{up}}{C_{dl} + C_{up}} \cdot m \tag{6}$$

□

not enough, as that peer is able to open many simultaneous connections to other peers and seeds, and download a small amount from each of them, in parallel.

[2] This is very similar in spirit to the famous coupon-collector's problem.

It remains to prove the lemma:

Proof of Lemma 3.3. A peer that has already downloaded once from a seed has exactly $m \cdot \frac{n}{q}$ blocks from the region. When it downloads pieces of this region from some seed a second time, a vector $\vec{s}$ and a set of n values $Vals$ are selected to define the slices of the file it gets. For a given choice of $\vec{s}$, the file is partitioned into equal-sized slices defined by hyperplanes of the form $\vec{s} \cdot \vec{a} = b$. Since we are selecting the set $Vals$ uniformly, each slice has the same chance of getting selected: $\frac{n}{q}$. Let g_b be the number of blocks that have already been obtained from slice b. We know that

$$\sum_{b \in \mathbb{F}_q} g_b = m \cdot \frac{n}{q} \tag{7}$$

The expected number of blocks that will be downloaded twice is then:

$$\sum_{b \in \mathbb{F}_q} \frac{n}{q} \cdot g_b = m \cdot \left(\frac{n}{q}\right)^2 = \frac{k^2}{m} \tag{8}$$

□

3.2 Stronger Properties of Protocol I

Protocol I relies on the random selection of hyperplanes and on the intersection between them to show that there are enough missing blocks after two sessions. In fact, a stronger property holds. Even after multiple sessions with seeds, there is a high probability that blocks will still be missing.

Theorem 3.4. *If m is large, then any peer that has accessed a seed t times, where $t < d/2$, will have exactly $m \cdot \left(\frac{q-n}{q}\right)^t$ missing blocks with probability approaching 1.*

The proof is given below.

The theorem implies that peers who diverge from the protocol and try to download solely from seeds will not be successful even after multiple sessions (in which they download quite a few blocks). The reason that the theorem holds is because randomly selected vectors are highly likely to be independent. We first show the following lemma:

Lemma 3.5. *If a seed is accessed $t < d/2$ times, the vectors $\vec{s}_1, \vec{s}_2, \ldots, \vec{s}_t$ that are chosen at different times during phase 2 of the protocol, are linearly independent with probability approaching 1.*

Proof. The vectors will be linearly dependent only if at some point a vector $\vec{s}_i$ is selected from the span of the previous vectors. If this never happens, they will be independent.

Let us assume that the vectors $\vec{s}_1, \ldots, \vec{s}_{i-1}$ are independent. Since we are dealing with a vector space over $\mathbb{F}_q$, there are q^{i-1} different linear combinations of these vectors, that give q^{i-1} different vectors from which a bad value of $\vec{s}_i$ could be selected. Since we select vectors from the entire space excluding the vector $\vec{0}$ (which is also one of the linear combinations that were counted), we get that the chance of a bad selection is $\frac{q^{i-1}-1}{q^d-1}$.

Since selections of vectors at different sessions are independent, we conclude that the probability of getting t independent vectors is

$$\begin{aligned} Pr\left(\begin{array}{c} \vec{s}_1, \ldots, \vec{s}_t \textit{ are} \\ \textit{independent} \end{array}\right) &= \prod_{i=1}^{t} \left(1 - \frac{q^{i-1}-1}{q^d-1}\right) \\ &> \left(1 - \frac{q^{t-1}-1}{q^d-1}\right)^t > \left(1 - \frac{q^{t-1}}{q^{d-1}}\right)^t \\ &= (1-q^{t-d})^t > (1-q^{-d/2})^{d/2} \end{aligned} \tag{9}$$

Remember that d was defined to fulfill $m = q^d$, so if m is large enough, d is also quite large. Notice in this case that the above bound goes to 1:[3]

$$\forall q \geq 2 \quad \lim_{d \to \infty} \left(1 - q^{-d/2}\right)^{d/2} = 1 \tag{10}$$

□

Armed with Lemma 3.5, we now proceed with the proof of the theorem.

Proof of Theorem 3.4. According to the lemma, with high probability the vectors $\vec{s}_1, \ldots, \vec{s}_t$ that were selected at each stage of the protocol are independent. We denote by $\overline{Vals_i}$ the complementary set of values to the set $Vals_i$ that was selected in the i'th execution of the protocol.

$$\overline{Vals_i} = \{b \in F_q : b \notin Vals_i\}$$

The set of addresses that were not covered at any stage of the protocol is the set for which the following holds:

$$\forall i \in \{1, \ldots, t\} \quad \vec{s}_i \cdot \vec{a} \in \overline{Vals_i} \tag{11}$$

Note that if we select just a single value $b_i \in \overline{Vals_i}$ at every execution, the linear equations

$$\forall i \in \{1, \ldots, t\} \quad \vec{s}_i \cdot \vec{a} = b_i \tag{12}$$

have exactly $\frac{m}{q^t}$ solutions. This is because we assumed the vectors $\vec{s}_i$ are independent, and the system of equations can be described succinctly with a matrix $\mathbb{S}$ (whose rows are the vectors $\vec{s_i}$) of rank t:

$$\mathbb{S} \cdot \vec{a} = \vec{b} \tag{13}$$

However, we do not have just a single set of values $b_1, \ldots, b_t$ but rather $(q-n)^t$ different combinations of values that can be selected. Each selection gives us different equations with a new set of solutions. The addresses that satisfy the equations for each selection are disjoint (for each different selection there exists some linear equation that differentiates among the solutions). We therefore have a total of

$$\frac{m}{q^t} \cdot (q-n)^t \tag{14}$$

different addresses of blocks that were not sent out at any stage. □

From the proof of the theorem we can see that the chances of successfully selecting a set of independent vectors increases if the set is smaller (naturally, two random vectors have a higher chance of being independent than three or more). A further improvement is obtained when the number of blocks is enlarged. The next example shows that the theorem above gives useful probabilities even for typical file sizes that are used today, and not just in the limit.

[3] Example 1 shows that this bound is quite close to 1 even for relatively small values of d, m.

Example 1. *A file of size 2^{24} bytes (16 megabytes) is being shared; the file can be divided into 16 regions. Each region contains 1024 blocks each of 1024 bytes ($r = 16$, $m = 1024$). Now assume that each seed wishes to donate exactly $\frac{1}{2}$ of the file, so we select $n = 1$, $q = 2$. A seed then uploads half a megabyte in each session. Since there are 2^{10} blocks per region, each block has a 10-bit address (in base 2) within the region ($d = 10$). A peer that accesses some seed 5 times will have gotten 5 independent vectors $\vec{s}_i$ with a probability of exactly 0.9747. If this event does occur, that peer will be missing* exactly 1/2 *of the region after accessing the seed once,* 1/4 *after accessing it twice,* 1/8 *after the third time, and so on. Still, at every stage the peer downloads exactly half a region, and so his download efficiency decreases with time. Note that the probabilities stated in this example could be further improved. This can be done by making each block smaller, and thus increasing the number of blocks per region.*

Notice that Theorem 3.4 gives an exact number of missing blocks with high probability, while Lemma 3.3 gives guarantees of an expected number of collisions (even if not all vectors are independent, we still have many expected collisions).

4. AN IMPROVED PROTOCOL

The robustness of a peer-to-peer system depends heavily on the availability of the entire file even in cases where all seeds have vanished. From Theorem 3.4 we know that the number of missing blocks from the previous transmissions of a seed decreases exponentially. That is, our previous protocol sends out a full copy of any given region only after $\Theta\left(m \cdot \ln\left(m\right)\right)$ blocks from that region have been uploaded (in expectation). A full copy of the region is thus quite likely to be missing in case all seeds fail and may thus never be completely obtained by the peers, especially if the number of blocks is large, as we have previously required.

Our second protocol aims to increase the availability of the entire file by judiciously uploading parts of the region that are missing, while still not allowing a single downloader to obtain the entire region from seeds at a low cost. The motivating idea behind this improvement is that it is enough that the set of vectors $\vec{s}_i$ that is selected during different sessions are pairwise independent (rather than independent as a group). That way, any two sets of blocks downloaded from seeds will have at least some guaranteed number of collisions. If this number of collisions is enough to discourage a peer from accessing seeds twice, then it will certainly discourage more attempts.

The protocol will randomly select a small linear subspace $U \subset V$ of dimension 3. A group of pairwise independent vectors $S \subset U$ will be constructed and used to decide which blocks are sent to peers in consecutive sessions. We will show that:
1. If all addresses in U are covered by hyperplanes determined using S alone, then all addresses in V are covered.
2. There are enough vectors in S to cover all addresses in the subspace U.
3. The number of sessions required to cover a single region is dramatically decreased.
4. Similarly to the previous simpler protocol, a peer that attempts to access the seed twice for the same region will download at a higher cost.

Because the subspace U is randomly selected, a peer that downloads twice from different seeds will most likely encounter addresses that were determined according to independent vectors from non-overlapping subspaces, i.e., it will not gain a downloaded segment without collisions.

The protocol again operates in sessions, although this time sessions will not be entirely independent from one another.

Protocol II:

Region Initialization:
1. Randomly and uniformly select 3 independent vectors $\vec{v}_1, \vec{v}_2, \vec{v}_3 \in V \setminus \{\vec{0}\}$. Let U be the linear subspace $U = \text{span}\{\vec{v}_1, \vec{v}_2, \vec{v}_3\}$.
2. Select a group of vectors $S \subset U$ such that S is a maximal group of pairwise independent vectors.
3. Set $C = U$ to be the set of addresses from U that have yet to be sent to any seed.

During the i'th time in which a certain region is being requested, do the following:
Session Actions:
1. Randomly and uniformly select a vector $\vec{s}_i \in S$ and remove it from the group.
2. Select n values $Vals = \{b_1, \ldots, b_n\}$ such that $b_i \in \mathbb{F}_q$ without repetitions, and such that the number of uncovered elements in $\vec{a} \in C$ for which $\vec{s}_i \cdot \vec{a} \in Vals$ is maximized. If there are several possible choices, randomize among them.
3. Send to the peer all blocks in the requested region that have an address $\vec{a}$ for which $\vec{s}_i \cdot \vec{a} \in Vals$.
4. Remove the newly covered block addresses from C. If C is empty, repeat the initialization for the region.

4.1 Analysis of Protocol II

We now prove the protocol's properties. The next lemma shows that it is enough to check that we cover all block addresses in U; the remaining addresses will also be covered.

Lemma 4.1. *Let V be a vector space over $\mathbb{F}_q$. Let $U \subset V$ be a subspace of V. If block addresses are selected using hyperplanes of the form defined by $\vec{s}_i \cdot \vec{a} = b_i$ where all vectors $\vec{s}_i$ are from U, and all addresses in U have been covered by some hyperplane, then the entire vector space V has been covered by these hyperplanes.*

Proof. Let us assume to the contrary that there exists some $\vec{v} \in V$ that has not been covered by any hyperplane. Let $\vec{u} \in U$ be the projection of v on U:

$$(\vec{v} - \vec{u}) \perp U \tag{15}$$

Since $\vec{u}$ is in U, it has been covered by some hyperplane that is defined by $\vec{s}, b$. We shall now show that this hyperplane also covers v:

$$\vec{s} \cdot \vec{v} = \vec{s} \cdot (\vec{v} - \vec{u}) + \vec{s} \cdot \vec{u} = 0 + b = b \tag{16}$$

which implies that v is indeed on the hyperplane. □

The next lemma gives some idea of the number of pairwise independent vectors that we can use:

Lemma 4.2. *The vector space U of dimension d over $\mathbb{F}_q$ contains a set S of pairwise independent vectors such that $|S| = \frac{q^d-1}{q-1}$*

Proof. We prove the lemma by induction on d. The base case will be $d = 1$. In this case $\frac{q^d-1}{q-1} = 1$ and the set that contains only one (non-zero) vector is trivially pairwise independent (in fact, in dimension 1 this is only a scalar).

Now, we will assume that the lemma is correct up to $d = d' - 1$ for some $d' \geq 2$ and show how to extend the proof to the case $d = d'$. Let $S_{d'-1}$ be a set of pairwise independent vectors of dimension $(d' - 1)$. We know that $|S_{d'-1}| = \frac{q^{d'-1}-1}{q-1}$.

We construct $S_{d'}$ by extending the vectors in $S_{d'-1}$ with an extra coordinate. We denote by $(b, \vec{v})$ the vector that has b in the first coordinate and the values of vector v in coordinates 2 to d'. We then define:

$$S_{d'} = \left\{(b, \vec{v}) \quad : \quad b \in \mathbb{F}_q, \quad \vec{v} \in S_{d'-1}\right\} \cup \{(1, \vec{0})\} \tag{17}$$

Note that the size of the set $S_{d'}$ is as required:

$$|S_{d'}| = q \cdot |S_{d'-1}| + 1 = \frac{q^{d'} - 1}{q - 1} \tag{18}$$

It now remains to show that every pair of vectors in $S_{d'}$ is linearly independent. Let $\vec{v}_1, \vec{v}_2$ be a pair of vectors from $S_{d'}$ such that $\vec{v}_1 \neq \vec{v}_2$. There are 3 cases. If the pair consists of 2 vectors that have different non-zero values in coordinates 2 to d' then the linear independence is derived from the linear independence of every pair of vectors in $S_{d'-1}$. If one of the vectors is $(1, \vec{0})$, then it only linearly depends on vectors of the form $(b, \vec{0})$ for some $b \in \mathbb{F}_q$ which are not present in the set. The third case is when both vectors are identical in coordinates 2 through d (some of which must be non-zero). In this case, the only linear combination that would have a chance to demonstrate dependence would have to be

$$b \cdot \vec{v}_1 + (-b) \cdot \vec{v}_2 \tag{19}$$

for some $b \in F_q$. However, the first coordinate of the vectors must be different, which means this combination cannot work (the result will be non-zero). The set therefore contains pairwise independent vectors only. □

We now show that the number of unsent blocks whose addresses are in U is reduced quickly.

Lemma 4.3. *For a given seed, and a given region, the number of unsent blocks from the subspace U is reduced at least by a factor of $\frac{q-n}{q}$ in every session in which blocks from that region are sent.*

Proof. When a vector $\vec{s}_i$ is selected during session i, it defines a partition on the space of all addresses into hyperplanes of the form $\vec{s}_i \cdot \vec{a} = b$ for different values of $b \in \mathbb{F}_q$. Each of these hyperplanes contains exactly $1/q$ of all addresses, and together they cover the entire space. Therefore, when selecting n such hyperplanes (without repetitions) it is possible to cover at least n/q out of the entire group of uncovered addresses that is contained within the space. Since the protocol maximizes the number of covered addresses in U, we are left with less than $\frac{q-n}{n}$ uncovered ones. □

Finally, we can prove that the protocol will successfully send out all blocks in the region quickly:

Theorem 4.4. *Each seed that acts according to Protocol II sends out a full copy of the region requested in under $\frac{q}{n}3 \cdot \ln(q) + 1$ sessions.*

Proof. According to Lemma 4.3 the number of uncovered addresses in U is reduced by at least a factor of $\frac{q-n}{q}$ at every step. There are q^3 different vectors in U, and so the number of sessions to cover U is bounded from above by:

$$\text{Num Sessions} \leq \frac{\ln\left(q^3\right)}{\ln\left(\frac{q}{q-n}\right)} + 1 = \tag{20}$$

$$\frac{3 \cdot \ln(q)}{-\ln\left(1 - \frac{n}{q}\right)} + 1 = \frac{q \cdot 3 \cdot \ln(q)}{-\ln\left(\left(1 - \frac{n}{q}\right)^q\right)} + 1 \leq \tag{21}$$

$$\frac{q}{n} \cdot 3 \cdot \ln(q) + 1 \tag{22}$$

where the last transition is due to the fact that

$$\left(1 - \frac{n}{q}\right)^q \leq e^{-n} \tag{23}$$

Note that we have used $\dim(U) = 3$ for which we have enough pairwise independent vectors as per Lemma 4.2:

$$\forall q \geq 2 \quad \forall q > n \geq 1 \quad 3\frac{q}{n} \cdot \ln(q) + 1 < \frac{q^3 - 1}{q - 1} \tag{24}$$

which implies that the protocol will not run out of pairwise independent vectors from the set S before covering all addresses in U. From Lemma 4.1 we learn that once this has happened, we also have covered the entire region. □

Combining Theorem 4.4 with the fact that $m \cdot \frac{n}{q}$ blocks are sent in every session gives us a total of fewer than $m \cdot (3 \cdot \ln(q) + 1)$ blocks that need to be sent by a seed to get an entire copy out. This is a significant improvement over the number of transmitted blocks in Protocol I: $\Theta(m \ln(m))$, since usually $m >> q$.

Still, peers do not gain from accessing seeds twice because they have just as many missing blocks as in Lemma 3.3:

Lemma 4.5. *A peer that contacts any of the seeds for 2 sessions will have with high probability exactly $\left(\frac{q-n}{q}\right)^2 \cdot m$ missing blocks from the region.*

The proof is similar to that of Lemma 3.3. Notice that if the peer accessed the same seed twice in sessions that did not have an initialization action between them, then surely the pieces of the region that the peer got were determined using two independent vectors. On the other hand, if an initialization did occur, or if the peer accessed two different seeds, then the two vectors that were selected are in fact independent with high probability (as in Theorem 3.4) which implies that they had some blocks in common.

The following example demonstrates Protocol II, and also shows that in some cases the construction in Theorem 4.4 is not tight, and it may be sufficient to work with a subspace U of dimension 2.

Example 2. *Let us assume that the file being shared contains 2^4 blocks per region.*[4] *Seeds wish to donate half of the file to peers ($q = 2$, $n = 1$, $d = 4$). For simplicity, let us assume that the vectors $(1, 0, 0, 0)$ and $(0, 1, 0, 0)$ were randomly selected to span the subspace U by some seed for some arbitrary region. The following set is a maximal pairwise independent set of vectors in that subspace: $S = \{(1, 0, 0, 0), (0, 1, 0, 0), (1, 1, 0, 0)\}$*

[4] This is just to keep the example simple. The number of blocks would usually be much larger.

The first time the region is requested by some peer, a value of $b = 1$ is selected by chance as the offset of the hyperplane, and addresses matching $(1, 0, 0, 0) \cdot \vec{a} = 1$ are sent. The second time blocks that match $(0, 1, 0, 0) \cdot \vec{a} = 0$ are sent (with $b = 0$ that is arbitrarily selected).

At this point, only one address in U has been left uncovered: the address for which $a_1 = 0, a_2 = 1$. Therefore, the third time the region is requested, with regard to the third vector $(1, 1, 0, 0)$ only one value of b covers any of the unsent blocks: $b = 1$. So blocks with addresses matching $(1, 1, 0, 0) \cdot \vec{a} = 1$ are sent.

Note that at every session the seed transmits $n/q = 1/2$ of the blocks in the region, and that after only 3 sessions, the entire region has been transmitted. This is only slightly slower than transmitting the entire region without using the protocol (which would take the equivalent of two sessions).

5. FAIR TRADING AMONG PEERS

We previously assumed that peers can trade blocks in a 1-to-1 ratio and that no peer can manipulate the protocol to improve this ratio. The strategy of sending out a block only after receiving one is deficient in several ways. First, it is too slow, since a peer has to wait to receive a full block before it can start transmitting one in return. Furthermore, it is unclear who sends the first block. If a block of data is sent by one peer, and the other peer does not upload in return, then he has managed to gain a block without uploading and can repeat this process to gain more blocks from other peers (possibly under a new identity). Here, we show that there exists a different strategy for trading blocks that guarantees a good trading ratio without impeding the speed of transfer.

We claim that the following strategy achieves these goals:
1. In the first round, send one block of junk.
2. At every consecutive round, send the other peer a new block from the file but only if he has sent more than 9/10 of the blocks you have sent him, and you did not send 100 blocks more than he has.

A similar strategy has been suggested by [11] without the initial upload of junk data. With the initial upload of junk data, a peer that drops the connection early is at a disadvantage. If it has sent junk data, it has already expended effort, and dropping the link will only mean expending this effort again to establish a connection later. If it has not sent any data, then all it will receive is the initial junk block from the other peer—which is worthless.

As the interaction between peers progresses, it is less and less beneficial to drop the connection, as the price of establishing a new link with some other peer is large compared to the current overhead of trading data. The restriction on a difference of 100 blocks implies that as the interaction goes on, the trade ratio approaches 1-to-1.

5.1 Additional Improvements

Here we discuss several other modifications that augment the protocols we presented above.

Rarest-first dissemination of regions: The BitTorrent protocol attempts to improve the robustness of the system by first distributing rare blocks. This helps in maintaining a full copy of the file in the network even if all seeds are down, and in speeding up the download (some peers may be waiting for the rare blocks alone after easily obtaining all commonly found ones). A similar approach can be combined with our protocol as well (although in our case regions play the role of blocks). Seeds should try to distribute full copies of regions that have not been fully sent out to the population of downloaders. Peers should attempt to complete rare regions with a higher priority than commonly found ones.

Thwarting connection dropping: A possible manipulation by peers that wish to download a larger portion of the file from seeds is to drop the connection to the seed if they ever realize that they are being given a block they already possess. That way, the peer can avoid the added cost of downloading useless information. To prevent peers from taking this approach, seeds that are interacting with a new peer for the first time should deterministically send the few first blocks in a region. Thus, if the seed is contacting them for a second time regarding that region, it will suffer a loss by having to always download a portion of the region it already possesses. Note however, that honest peers that only access the seed once are not harmed, since they do not possess those blocks to begin with.

Lowering the seed donation ratio: The idea of a deterministic initial donation by seeds can also be used to reduce the amount of donated effort by seeds even below the threshold given in Theorem 3.2. Setting a constant part of the region that is deterministically (or with higher probability) sent to requesting peers increases the number of redundant blocks a misbehaving peer will download, and makes it less worthwhile to access peers several times. The following example demonstrates this:

Example 3. *Let us assume that seeds donate the first k_1 blocks from the region deterministically, and then randomly select k_2 blocks from the remaining $m-k_1$ blocks in the region (according to one of the protocols we have presented). The expected cost incurred by a peer that downloads only once from a seed and then trades for the rest of the file is then:*

$$C_1 = (k_1 + k_2) \cdot C_{dl} + (m - k_1 - k_2) \cdot (C_{dl} + C_{up}) \qquad (25)$$

A peer that downloads twice pays $2(k_1+k_2)\cdot C_{dl}$ for downloads, and has $k_1 + \frac{(k_2)^2}{m-k_1}$ redundant blocks that he downloads (in expectation). He thus has an expected cost of:

$$\begin{aligned} C_2 = \quad & 2(k_1 + k_2) \cdot C_{dl} + \\ & + \left(m - k_1 - 2k_2 + \frac{(k_2)^2}{m - k_1}\right) \cdot (C_{dl} + C_{up}) \end{aligned} \qquad (26)$$

The difference in costs is

$$C_2 - C_1 = (k_1+k_2)\cdot C_{dl} + \left(\frac{(k_2)^2}{m - k_1} - k_2\right)\cdot(C_{dl}+C_{up}) \quad (27)$$

For the sake of simplicity, let us assume that $C_{up} = C_{dl}$. The difference in costs is then

$$C_2 - C_1 = \left(k_1 - k_2 + 2\frac{(k_2)^2}{m - k_1}\right) \cdot C_{dl} \qquad (28)$$

Now, notice that we can set $k_1 = k_2$ for example, and always get a positive difference in costs:

$$C_2 - C_1 = 2\frac{(k_2)^2}{m - k_1} \cdot C_{dl} \qquad (29)$$

A seed can therefore choose to upload only one quarter of the file, by selecting $k_1 = k_2 = \frac{m}{8}$. This naturally comes at a cost to the robustness of the system, since half of the donation of seeds (the k_1 blocks that are sent deterministically) goes towards blocks that are very frequent, instead of promoting diversity and sending blocks that are more rare.

Proof of effort: If peers have only a very small piece of a region missing it may be hard for them to obtain it. Seeds can give out this small missing piece, and in return request information to be sent to them from the downloading peer. This sent information (which can be junk, and does not have to be part of the file) will impose a small cost on the peer that will prevent him from downloading entire regions in this manner without incurring an appropriate cost. This mechanism should be used sparingly, as it wastes bandwidth for a transmission that is not really beneficial.

6. DISCUSSION AND FUTURE WORK

We have presented a file-sharing protocol that adds incentives to increase the level of donation by downloading peers. In particular, our protocol is resistant to the *Large-View exploit* [15], since peers cannot gain a higher utility by connecting to many seeds—each seed gives out information somewhat randomly, and so sessions with many seeds are equivalent to sessions with only one seed. We have further shown how the protocol could be modified to increase the robustness of the system when the number of seeds is low.

It is interesting to note that when the number of seeds is small (e.g., when the download starts and only a single seed exists in the system) then the problem of incentives in the regular BitTorrent protocol is not too pressing. Since the download bandwidth provided by seeds at this point is low, peers who trade blocks get more downloads, and selfish peers have a harder time competing for the scarce bandwidth of the seeds. Therefore, uploading peers obtain a copy of the file much faster—almost as soon as a single copy of the file is uploaded. As the number of seeds grows, free-riders can again download quickly. We therefore suggest a hybrid approach: if the system is deemed to be in a "risky" state, then seeds should allow requests for specific blocks. However, if there are plenty of seeds, our protocol would hurt free riders and can be used without worrying about robustness. This hybrid approach is common in several BitTorrent clients, e.g., those that use the *Super-Seeding* mechanism when the number of seeds is low, and resume their regular behavior after the number increases.

Another related challenge that still remains is to make the protocol more secure against malicious activity. Modern peer-to-peer protocols usually employ some method of verification (usually via hashing) to check blocks they have obtained, so that blocks that contain errors or have been planted by a malicious agent are not propagated to the entire system. This presents a challenge, since peers may be holding only partial slices of a region, which may be harder to verify (and we would not want to propagate even this partial information in case it is wrong). A similar challenge exists in file-sharing systems that use network coding, and some solutions do exist (e.g., those mentioned briefly in [8]).

Seeds in our model were considered to be agents who donate without expecting anything in return. Often, someone that is uploading one file would be interested in downloading another. The question of how to design systems that consider more than one file at a time naturally arises, especially in a protocol that does not depend on a monetary or reputation system. Perhaps trading pieces of different files can prove to be effective, provided that trading partners that can form beneficial deals are found and matched efficiently.

Acknowledgment

Both authors were supported in part by Israel Science Foundation grant #898/05.

7. REFERENCES

[1] E. Adar and B. A. Huberman. Free riding on gnutella. *First Monday*, September 2000.

[2] A. Cheng and E. Friedman. Sybilproof reputation mechanisms. In *P2PECON '05: Proceedings of the 2005 Workshop on Economics of Peer-to-Peer Systems*, pages 128–132, NY, 2005. ACM.

[3] B. Cohen. Incentives build robustness in BitTorrent. In *The 1st Workshop on the Economics of Peer-to-Peer Systems*, Berkeley, California, June 2003.

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

[5] M. Feldman, K. Lai, I. Stoica, and J. Chuang. Robust incentive techniques for peer-to-peer networks. In *EC '04: Proceedings of the 5th ACM Conference on Electronic Commerce*, pages 102–111, NY, 2004. ACM.

[6] M. Feldman, C. Papadimitriou, J. Chuang, and I. Stoica. Free-riding and whitewashing in peer-to-peer systems. In *Proceedings of the ACM SIGCOMM Workshop on Practice and Theory of Incentives in Networked Systems*, pages 228–236, NY, 2004. ACM.

[7] Z. Ge, D. R. Figueiredo, S. Jaiswal, J. Kurose, and D. Towsley. Modeling peer-peer file sharing systems. In *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies. IEEE*, volume 3, pages 2188–2198, March 2003.

[8] C. Gkantsidis and P. R. Rodriguez. Network coding for large scale content distribution. In *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, volume 4, pages 2235–2245 vol. 4, 2005.

[9] P. Golle, K. Leyton-Brown, and I. Mironov. Incentives for sharing in peer-to-peer networks. In *EC '01: Proceedings of the 3rd ACM conference on Electronic Commerce*, pages 264–267, NY, 2001. ACM.

[10] Ipoque. Internet study 2007, October 2007. www.ipoque.com/media/internet_studies/internet_study_2007.

[11] S. Jun and M. Ahamad. Incentives in BitTorrent induce free riding. In *P2PECON '05: Proceeding of the 2005 ACM SIGCOMM workshop on Economics of peer-to-peer systems*, pages 116–121, NY, 2005. ACM.

[12] D. Levin, K. LaCurts, N. Spring, and B. Bhattacharjee. BitTorrent is an auction: analyzing and improving BitTorrent's incentives. *SIGCOMM Comput. Commun. Rev.*, 38(4):243–254, 2008.

[13] M. Li, J. Yu, and J. Wu. Free-riding on BitTorrent-like peer-to-peer file sharing systems: Modeling analysis and improvement. *IEEE Transactions on Parallel and Distributed Systems*, September 2007.

[14] T. Locher, P. Moor, S. Schmid, and R. Wattenhofer. Free riding in BitTorrent is cheap. In *5th Workshop on Hot Topics in Networks*, Irvine, CA, US, Nov. 2006.

[15] M. Sirivianos, J. H. Park, R. Chen, and X. Yang. Free-riding in BitTorrent with the large view exploit. In *6th Int. Workshop on Peer-to-Peer Systems*, 2007.