One hop Reputations for Peer to Peer File Sharing Workloads

Michael Piatek Tomas Isdal Arvind Krishnamurthy Thomas Anderson
University of Washington

Abstract

An emerging paradigm in peer-to-peer (P2P) networks is to explicitly consider incentives as part of the protocol design in order to promote good (or discourage bad) behavior. However, effective incentives are hampered by the challenges of a P2P environment, e.g. transient users and no central authority. In this paper, we quantify these challenges, reporting the results of a month-long measurement of millions of users of the BitTorrent file sharing system. Surprisingly, given BitTorrent's popularity, we identify widespread performance and availability problems. These measurements motivate the design and implementation of a new, one hop reputation protocol for P2P networks. Unlike digital currency systems, where contribution information is globally visible, or tit-for-tat, where no propagation occurs, one hop reputations limit propagation to at most one intermediary. Through trace-driven analysis and measurements of a deployment on PlanetLab, we find that limited propagation improves performance and incentives relative to BitTorrent.

1 Introduction

Peer-to-peer (P2P) networks have the potential to address long-standing challenges in networked systems. End hosts represent an immense pool of under-utilized bandwidth, storage, and computational resources that, when aggregated by a P2P network, can be used to absorb flash crowds, replicate data intelligently within the network, and externalize bandwidth costs. And unlike network-layer support such as IP multicast, P2P solutions can be deployed without architectural changes to the underlying network.

While significant progress has been made towards addressing the *technical* challenges of building P2P systems, their robustness ultimately depends on convincing users to contribute their resources, a challenge of *incentive design*. Early P2P systems such as Gnutella ignored incentives and were plagued by rampant free-riding, i.e., users consuming resources without contributing them [1]. Free-riding degrades system performance and limits scale. Subsequent systems such as BitTorrent explicitly built user contribution incentives into their design [4], but recent work has exposed methods of circumventing BitTorrent's incentives [11, 12].

Designing robust incentives for P2P networks is challenging due to the constraints of the environment:

- *No central control or trust:* Many practical problems in P2P data sharing become trivial if we can assume a "deus ex machina"—some authority that can mint currency, perform accounting, and penalize miscreants. To date, P2P designs that rely on centralization of these tasks have not been widely adopted.
- *Open implementation:* Users are free to adopt any client implementation, even one that attempts to subvert incentives or strategize. This makes the P2P design challenge harder, as problems like free-riding can be defined away if all users must connect using a particular software release.

This paper concerns how best to design future incentive strategies for P2P networks. We proceed in two steps. First, to ground our work, we conducted a measurement study of BitTorrent. BitTorrent is a widely used P2P system and we were able to study the sharing behavior of tens of thousands of data objects and millions of users for more than one month. Surprisingly given BitTorrent's popularity, we identify widespread performance and availability problems, along with data on why these problems arise in practice. We find that problems cannot be wholly attributed to scarcity of potential data sources and/or capacity limitations. Instead, we argue that ineffective incentives account for the lack of resources, a point underscored by our measurement result that an average user joining an average swarm can get comparable download performance with only 1/100th the contribution.

A key reason for the weakness of current incentives is the duration for which they are active. Current incentives in BitTorrent operate within the context of a single object and only while clients are actively downloading. As a result, users have no reason to contribute once they have satisfied their immediate demands. This weakness implies the need for persistent incentives that operate across data objects and across time. Unfortunately, our measurements also show that most pairs of peers interact with one another in just one swarm, suggesting that long-term incentives will not arise from strategies based on direct interactions and local history alone. But, while most P2P users are transient, our study shows that a small minority of peers participate persistently and across many swarms; these users provide a scaffold for a solution.

The second part of the paper concerns our design of a solution to the problems we found in BitTorrent. We propose a new, *one hop* reputation protocol for P2P networks. Unlike digital currency systems, where contribution information propagates globally, or tit-for-tat, where no propagation occurs, one hop reputations limit propagation to at most one level of indirection. Surprisingly, this limited propagation suffices to provide wide coverage; we find that the majority of peers, while transient, have shared relationships through popular intermediaries one hop removed. We define a protocol that discovers these relationships, enabling a broad range of servicing policies using information beyond direct observations and local history. Through trace-driven analysis and measurements of a deployment on PlanetLab, we show that our default one hop system both improves performance for users in individual swarms and fosters the long-term incentives that are necessary for P2P systems to work well in the long run.

2 Sharing in the wild

To understand the real challenges facing P2P designers, we collected large-scale measurements of BitTorrent in the wild. Over the course of the study we observed more than 14 million peers and 60,000 swarms accounting for thousands of terabytes of transfered data. To measure the strength of contribution incentives, we joined real swarms, exchanged data at varying rates with peers, and collected information to distinguish unique users such as client software and version and IP address. We also tracked the popularity of swarms over time, recording both direct observations of peers and second hand accounts from coordinating tracker servers, membership DHT entries, and peer gossip messages.

Our measurements provide insights into the sharing workload that extend beyond the granularity of performance for a single user or behavior in a single swarm. Specifically, we show the following:

- Performance and availability in BitTorrent is extremely poor. The median download rate in observed swarms is 14 KBps for a peer contributing 100 KBps, and as many as 25% of swarms are unavailable.
- These performance and availability problems are not fundamental. Our measurements show that sufficient capacity is available to provide much better performance than is observed today, and many unpopular objects would see their availability improve if previous downloaders could be offered sufficient incentives to persist as replicas.
- Existing incentives in BitTorrent, while designed to encourage contribution, are largely ineffective. Because of the structure of the workload, BitTorrent incentives permit free-riding and strategic manipulation for the majority of BitTorrent swarms.
- Simple extensions to BitTorrent's incentive strategy, e.g., using direct long-term reciprocation for contributions, will not address the observed problems due to a lack of repeat interactions in the sharing workload. However, the significant disparity in the popularity of P2P users points to the promise of new approaches based on indirect reciprocation.

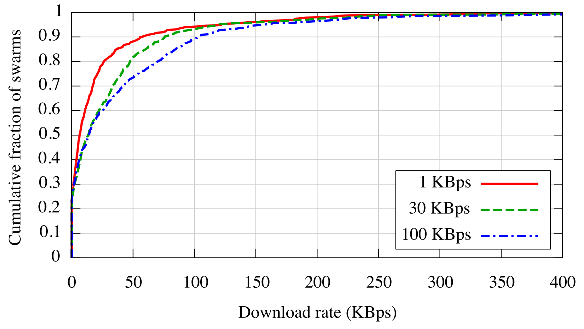


Figure 1: Download performance for different levels of contribution in BitTorrent. Each line gives the distribution of download performance for a contribution level as measured across thousands of real-world swarms in trace `BT-1`. Significant increases in contribution result in slight, if any, improvement in download performance.

2.1 Trace methodology

In BitTorrent, each file is split into blocks. Clients actively downloading a file are randomly matched, with matched peers exchanging data and control information as to which blocks they have and which they need. Ideally, a data source, or *seed*, only needs to provide each data block to a few random clients, and the rest of the work is done by the swarm of peers. Crucially, peers distinguish among competing requests for service according to a tit-for-tat policy: each client preferentially uploads blocks only to those peers that are actively providing data to it (and then, only to those that are providing data at the highest rate). Tit-for-tat is intended to provide better performance for peers that contribute more data.

The reported effectiveness of tit-for-tat has varied widely in existing work. Theoretical analysis, simulation and small testbed studies have pointed to its robustness [2, 10, 15] while more recent studies of performance in the wild have exposed circumstances under which tit-for-tat breaks down [11, 12, 16]. For system builders to design truly robust incentive protocols, a more complete understanding of P2P workloads is required. We collect and analyze BitTorrent trace data with the overarching goal of understanding when tit-for-tat incentives work and when they don't, in the wild.

We make reference to two traces of live BitTorrent swarms collected from a cluster of machines at the University of Washington. Between January 26th and February 3rd, 2007, we measured membership and download performance for instrumented clients participating in 13,353 swarms. We refer to this trace as `BT-1`. We collected a second trace, `BT-2`, from the same cluster over the month of August 2007, providing measurements of 55,523 swarms. In both traces, every hour, a measurement coordinator crawled popular BitTorrent websites that aggregate information about new swarms, downloading all of these. Our instrumented clients joined these swarms periodically during the trace. We include information for only those swarms we successfully connected to at least once. To determine peer download rates, we measured the rate at which new blocks appeared in the peer's list of available blocks and also recorded availability of blocks. Each client contributed resources to the swarms at a rate of either 1, 30, or 100 KBps to examine the performance impact of varying the contribution level.

2.2 BitTorrent performance and availability

The download rate achieved by our measurement clients as a function of contribution rate is summarized in Figure 1 for trace `BT-1`. Even on the well-connected academic network used for our data collection, clients download slowly; contributing 100 KBps yields a median download rate of just 14 KBps, far short of saturating even a modest home broadband connection. Further, 25% of the time swarms were completely unavailable, i.e., delivered no data.

The poor performance of P2P networks cannot be explained by users simply lacking the capacity to offer peers a high average download rate, nor can poor availability be attributed to a long tail of fundamentally unpopular objects. Regarding performance, measurements of more than 100,000 BitTorrent peers in 2006 put average upload capacity at more than 400 KBps [8]. Skew in the capacity distribution is significant; the average value is roughly 10X the median. Regarding availability, our measurement results show that for many seemingly unpopular objects, the *existence* of replicas is not as much a problem as the *persistence* of replicas. The vast majority of swarms would have significantly more replicas if downloaders would simply continue to share after completing. We evaluate this by comparing available replicas assuming peers persist for either one day or one week after their initial observation. For trace `BT-2`, the median increase in available replicas is a factor of 3.

These measurements point to a problem of incentives. If users could be *convinced* to contribute all of their capacity, download performance would increase. If users were *convinced* to persist as object replicas, availability would improve. Realizing these benefits requires understanding the causes of today's weak incentives, the topic we turn to next.

2.3 Workload causes for weak incentives

The strength of a contribution incentive is the return it provides for contribution, i.e., the ratio of uploaded to downloaded bytes. This section details two workload properties that weaken contribution incentives in BitTorrent. First, we examine how the distribution of bandwidth capacity among peers influences the incentive they have to make that capacity available. Second, we quantify the number of swarms for which random, altruistic contributions dominate performance.

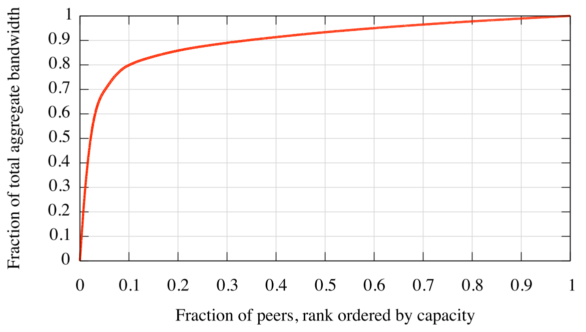


Figure 2: Cumulative fraction of total capacity (y-axis) attributed to the percentage of total peers rank ordered by capacity (x-axis). 80% of the total aggregate capacity of BitTorrent peers comes from the top 10% of users.

Capacity: In BitTorrent, returns are known to diminish as contribution increases [12]. Peers at the low end of the capacity spectrum see large returns on their contributions, i.e., 10 bytes contributed might earn 15 reciprocated. This is balanced by reduced returns for peers with greater capacity. If the disparity between returns for high and low capacity peers were limited, contribution incentives would be only slightly weakened. In BitTorrent, however, the disparity is extreme. In our traces, increasing contributions 100-fold yields a 2-fold median marginal improvement in performance (shown for `BT-1` in Figure 1).

The diminishing returns for contributions is particularly damaging for aggregate P2P resources as the majority of capacity is held by a small minority of users. Figure 2 shows the cumulative fraction of total capacity attributable to peers when ordered by individual capacity. If they were to contribute fully, the top 10% of peers would account for 80% of total capacity. Thus, for the highest capacity peers—those whose increased contribution would most help performance—the contribution incentive is weakest.

Altruism: A user's downloaded bytes come from either other peers actively downloading the object or seeds that have completed their downloads and continue to make data available. Because seeds do not have requests, they have no tit-for-tat basis for making servicing decisions, often doing so randomly in current implementations. An overabundance of seeds weakens contribution incentives as most users receive data regardless of contribution. Conversely, too few seeds also weakens incentives since

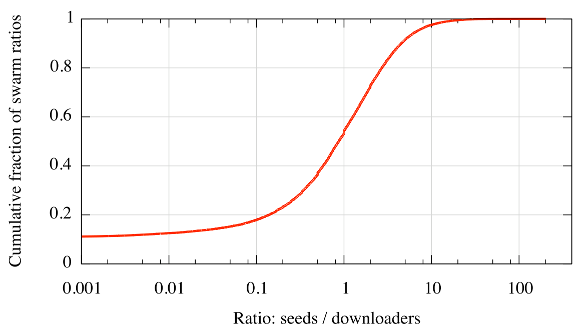


Figure 3: Ratios of seeds to downloading peers for swarms in BT−2. 11% of swarm observations showed no active seeds.

peers quickly run out of data to trade, becoming blocked.

Figure 3 summarizes the amount of seed-based altruism in our BT−2 trace. For each swarm, we compute the ratio of observed seeds and downloaders. This estimates the fraction of data a downloading peer is likely to receive at random, i.e., independent of contribution. The data shows that no one circumstance dominates. 11% of swarm observations show no active seeds (ratio 0) while 50% of swarms have just as many randomly contributing seeds as actively downloading peers.

Given the range of operating conditions we observe in practice, it is unsurprising that the BitTorrent performance picture is unclear. Some swarms enjoy a glut of altruistic donations, weakening contribution incentives and enabling free-riding. Other swarms are starved for data, causing performance to be constrained by availability rather than contribution. For the minority remaining swarms, the strength of the contribution incentive is tied to the bandwidth capacity distribution, with the majority of capacity being held by peers with little reason to contribute, leading to slow download rates.

2.4 A straw-man solution

In contrast to the standard game-theoretic tit-for-tat strategy, BitTorrent's variant is rate-based. Instead of trading with peers byte for byte, reciprocation for a BitTorrent peer is decided only relative to its competitors and is apportioned equally among successfully competing peers. For instance, if a client C with capacity 20 receives data from peers X, Y, and Z at rates 5, 7, and 10 and selects only two peers at a time for reciprocation, C will send data to Y and Z at rate 10 apiece. This approach favors utilization over fairness and stateless operation over stability. Peers simply give away bandwidth if they are poorly matched in terms of rates and maintain only a short-term local history about each peer with which to make servicing decisions, switching peers frequently as short-term status changes.

An alternative to basing tit-for-tat decisions on *rate* is instead basing them on total data *volume*. This is the approach taken by the eDonkey file sharing network, which stores per-peer state recording the amount of data sent and received, using this to rank peers with competing requests. From the perspective of strengthening contribution incentives, a switch from rate to volume seems promising, primarily because it offers the potential for long-term repeat interactions. Seeds might be willing to share files long after completion, improving availability, because in doing so they would contribute to peers whose memory of those contributions would induce reciprocation if the situation were reversed. Similarly, if high capacity peers were mismatched with low capacity peers, the contribution imbalance could be bounded or ignored—assuming repeat interactions would result in eventual repayment.

Unfortunately, volume-based tit-for-tat does not seem to have solved the performance problem in practice. Pucha et al. report a median download rate of 10 Kbps in the eDonkey network [14]—short of our observed median performance for BitTorrent swarms. Although numerous technical differences prohibit an apples-to-apples comparison, we hypothesize that the failure of volume-based tit-for-tat to promote contribution in eDonkey can be traced to a workload property that it likely shares with BitTorrent—a lack of pairwise repeat interactions.

We say that two peers share an interaction if either sends or receives data from the other. Peers exhibit repeat interactions if they exchange data in multiple swarms. Figure 4 reports the frequency of repeat interactions in the BT−2 trace, conditioned on a pair having interacted in at least one swarm. Because our trace data provides only coarse-grained observations of peer membership, i.e., we do not actively probe observed peers repeatedly to determine departure time, we give the distribution of repeat interactions assuming peers persist for either an

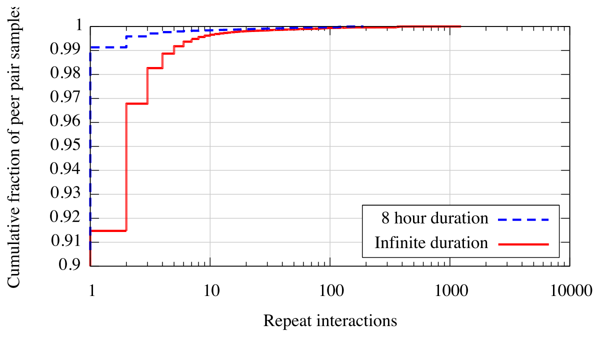


Figure 4: CDF of the frequency of repeat interaction between a pair of peers in the BT−2 trace. Note that the y axis is not zeroed and only pairs interacting at least once are considered. Even assuming infinite duration in swarms, 91.5% of interacting peers will do so only once. Limiting persistence reduces the chance of repeat interaction further to less than 1%.

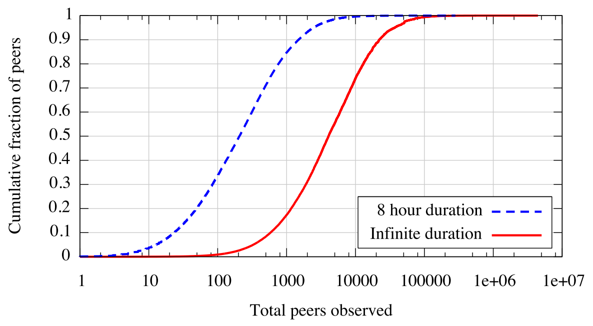


Figure 5: The distributions of peers encountered by BitTorrent users in the BT−2 trace. Whether assuming infinite or limited duration, a small minority of popular peers participates broadly.

infinite duration or an 8 hour interval. Assuming infinite duration overestimates the number of repeat interactions, while assuming an 8 hour duration may underestimate it for some long-lived peers. In either case, however, the chance of enabling long-term incentives via repeat interactions is slim. Even assuming infinite duration, more than 91.5% of peer pairs that occur in a single swarm do not arise in any other swarm over the course of our trace.

The apparent lack of repeat interactions suggests that direct, pairwise exchange based on local history alone will not suffice to enable the long-term contribution incentives needed to address the performance and availability problems we observe in the wild. But, although *direct* interactions appear insufficient, our workload measurements do provide a hint as to the effectiveness of *indirect* reciprocation; i.e., instead of peer A deciding whether to service the requests of B only on the basis of B's contributions to A, indirect reciprocation might see A contributing to B due to B's contributions to C, who has previously contributed to A.

Our data shows that most peers share an indirect relationship of this type. Further, a small number peers account for most of these intermediaries. 97% of all peers observed in trace BT−2 are connected either directly or through an intermediary among the most popular 2000 peers. This is due to a workload characteristic. Although most peers connect to only hundreds of other peers, a small minority is more extensively connected. Figure 5 shows the distribution of peer connectivity in trace BT−2.

The disparity in peer popularity is reflected in the distribution of demand as well. Figure 6 shows the distribution of total demand observed in our trace. We first ordered peers by popularity, i.e., the number of other peers with which they share a swarm. Next, we computed the cumulative fraction of demand attributable to these ordered peers. The top 25% of peers account for 78% of demand.

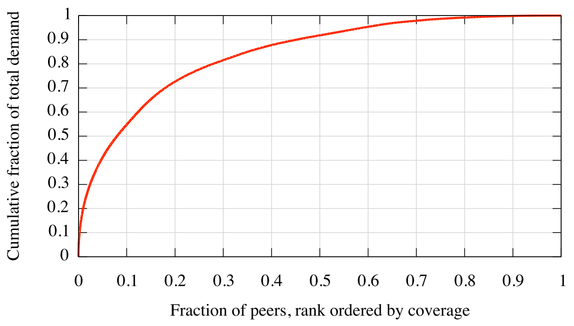


Figure 6: Cumulative fraction of consumption (y-axis) attributed to peers, ordered by popularity (x-axis).

3 One hop indirect reciprocation

Although repeat interactions are rare for the majority of peer pairs, a small minority of popular users have much wider coverage. In this section, we describe a new, *one hop* reputation propagation protocol designed to enable long-term reciprocation beyond this minority via indirect reciprocation. The main goal of one hop reputations is to foster persistent contribution incentives by recognizing and rewarding contributions made by users across swarms and over time. To achieve this, clients maintain a persistent history of interactions and, upon request, serve as intermediaries attesting to the behavior of others.

The key idea behind our scheme is to restrict the amount of indirection between contributing and reciprocating peers to at most one level of intermediaries. This restriction limits the propagation of information, promoting scalability, and allows for local reasoning about the trustworthiness of intermediaries, thereby fostering robustness.

While our measurements show that most peers share a one hop relationship, discovering and using these relationships requires more information than is available through direct observation alone. Peers need to name one another persistently across interactions and exchange messages about third party behavior. In Section 3.1, we define a protocol for exchanging the information required to discover intermediaries and to mediate indirect reciprocation.

Our protocol provides information but does not prescribe how that information must be used, separating the mechanism for exchanging information from the policy for using it. In Section 3.1, we specify a default policy designed to maximize coverage, i.e., the fraction of pairs of peers that can evaluate one another using one hop reputations. We also describe the resistance of our default policy to various forms of strategic manipulation, but we do not claim to be robust to all forms of attack. Instead, our design is intended to allow peers to freely evolve their strategies independently, and we consider several potential alternatives.

Notation	Definition
$n(x \rightarrow y)$	bytes sent directly from x to y
$n(x \leftarrow y)$	bytes received directly from y by x
$n(x \xrightarrow{y} *)$	bytes sent to other peers due to y's recommendation as the intermediary
$n(x \xleftarrow{y} *)$	bytes received by x from other peers with y acting as the intermediary
$n(* \xrightarrow{x} y)$	summation of all bytes from any peer sent to y due to x's referrals
$n(* \xleftarrow{x} y)$	summation of all bytes sent by y to each of x's referrals
$\mathrm{rate}(x \leftarrow y)$	the average rate at which y provided data to x

Table 1: State at client x for each peer y.

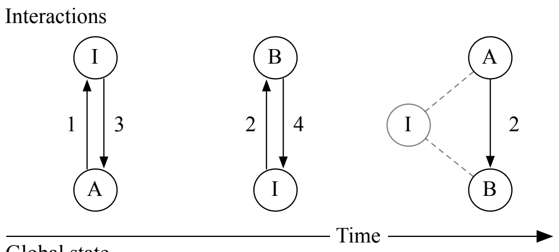
3.1 One hop reputation protocol

Our one hop reputation protocol can be broken down into two facets: the state maintained at each peer and messages used to propagate state between peers.

Per-peer state: One hop reputations extend volume-based tit-for-tat to incorporate reputation intermediaries. Intermediaries serve two purposes: bootstrapping connections between new peer pairs and maintaining accounting information regarding indirect reciprocation. Every client records each peer it has interacted with, either directly during data transfer or indirectly when that peer acts as an intermediary attesting to the behavior of others. Each peer is identified by a self-generated public/private key pair. While a peer can freely create new identities, our default policy rewards long-term persistence and includes provisions for mitigating Sybil attacks [6], creating little incentive to do so. Table 1 lists the state maintained by each client, which is indexed by the public key of its peers.

Figure 7 provides an example of the use of this information to bootstrap a new connection. In this case, a one hop intermediary I bootstraps the interaction between peers A and B who have not previously interacted. In the first two interactions, I exchanges data directly with A and B. These uninformed exchanges are infrequent and serve to bootstrap a reputation. At this point, I can serve as an intermediary between A and B. When they meet, A and B exchange control traffic (defined below) allowing them to recognize their common relationship with I. Because B has contributed to I in the past and A has received prior service from I, A can use its local history regarding I to inform its valuation of B.

Because the interactions between A, B, and C may not occur within the context of a single swarm, peers may need to contact intermediaries across multiple sessions. To aid in this, each client stores its current IP address and TCP port, indexed by its public key, in a DHT. Many popular P2P services already include a DHT, e.g.,



Global state

$n(I \rightarrow A) = 3$	$n(B \rightarrow I) = 4$	$n(A \xrightarrow{I} *) = 2$
$n(I \leftarrow A) = 1$	$n(B \leftarrow I) = 2$	$n(B \xleftarrow{I} *) = 2$
$n(A \leftarrow I) = 3$	$n(I \leftarrow B) = 4$	$n(A \leftarrow B) = 2$
$n(A \rightarrow I) = 1$	$n(I \rightarrow B) = 2$	$n(B \rightarrow A) = 2$

Figure 7: An example of peer state information used for bootstrapping. A recognizes B's standing with intermediary I. Dashed lines indicate prior interactions.

Kademlia is used in BitTorrent and eDonkey. Although existing DHTs are generally robust, we do not evaluate their resistance to strategic or malicious behavior, instead opting to use provided values as hints only. Identity is independently verified using cryptographic keys and key $\rightarrow$ IP mappings are locally cached.

State propagation: In the example of Figure 7, peer pairs A, I and B, I learn about one another directly through data transfer, requiring no explicit signaling. However, when A and B meet, they must exchange messages indicating which peers (possible intermediaries) they share in common and their status with those shared peers. In our protocol, this is a multi-step process.

1. First, peers order their local set of possible intermediaries to form what we refer to as their *top K set*. Inclusion in the top K set is a matter of local policy, but ordering this list by number of observations (our default policy) promotes the exchange of popular peers as intermediaries, increasing one hop coverage. Peers exchange topK messages upon connection.
2. Next, the intersection of local and remote top K sets is computed. This intersection is the set of shared peers that might be used as intermediaries for indirect reciprocation, but more information needs to be exchanged to compute the remote peer's reputation value.
3. For each possible intermediary, peers request attestation *receipts* of contribution by sending a receipt request message, containing the identity of the intermediary, to the remote peer. An attestation receipt for peer B from an intermediary I includes B's local state at I, time stamped and signed by I's private key.
4. Multiple peers can serve as intermediaries to mediate a specific interaction. If so, byte counts in local histories are updated fractionally based on the relative weight of each intermediary's valuation at the data source. This attribution message, a set of {identifier, weight} tuples, is sent to the receiving peer before data is transferred.

5. Once peers begin exchanging data, receipt messages are sent periodically from receiver to sender, to provide proof of received data and the corresponding increments to the sender's valuation. Before transmitting data, the sending peer dispatches a reserve message to mediating intermediaries containing the requesting peer's identifier and request size. These messages serve to preempt attacks based on using the recommendation of an intermediary multiple times and are optional. Periodically, update messages are sent to intermediaries, batching the reporting of transfers attributed to them, documented by received receipts.

In addition to facilitating identification of common intermediaries among peer pairs (Steps 1, 2), top K sets also bootstrap the local histories of new peers in the system. Each entry in the topK message contains a bit indicating whether the entry corresponds to an intermediary that can mediate a direct transfer or a gossip entry for a popular intermediary with whom the sender does not have a direct or indirect relationship. This is an optimization, effectively using two hop propagation to bootstrap one hop reputations. A client relying on direct, local observations alone to derive the coverage of potential intermediaries would have to directly observe them multiple times in distinct swarms before identifying those with high coverage. In the interim, new users would be unable to evaluate the quality of peers in good standing with popular intermediaries. Instead of relying on direct observations alone, peers may incorporate the gossip intermediaries included in the top K sets of their directly connected peers, combining this information with direct observations in their local history. Care must be taken when incorporating this information to prevent strategic manipulation, an issue we will discuss later.

3.2 Policies

Our state exchange protocol provides information about peers that enables a range of valuation policies. In this section, we propose a default policy designed to maximize coverage. However, peers are not required to follow this policy, and we present it as just one plausible design point among several alternatives. Other options are possible, e.g., trading coverage for resistance to strategic manipulation and collusion.

3.2.1 Default policy

Our default policy is the indirection-enabled analogue of volume-based tit-for-tat. When a peer makes servicing decisions, it restricts contribution to only those peers who have a positive or near positive "balance" with the system as a whole. This limits the potential for free-riding, if most peers have a one hop basis for making decisions. In this section, we define precisely how each client ranks the requests of others using one hop information and how membership of possible intermediaries in the top K set is decided.

Default servicing policy of peer L

```
1: for each peer $P$ requesting service
2:     if $P$ and $L$ have interacted directly
3:         $Reputation_P \leftarrow$ Computed using Equation 4
4:     else
5:         $topK_P \leftarrow$ Send topK request
6:         $mediators_P \leftarrow$ random_subset($topK_P \cap topK_L$)
7:         for each intermediary $I \in mediators_P$
8:             Send receipt request to $P$ for $I$
9:         done
10:        $Reputation_P \leftarrow$ Computed using Equation 3
11:    fi
12: done
13: $N \leftarrow \sum_{P|Reputation_P>1.0-\epsilon} Reputation_P$
14: for each peer $P \mid Reputation_P > 1.0 - \epsilon$
15:     $N_P \leftarrow \sum_{I \in mediators_P} w_L(I)$
16:     Send Attribution message, $\forall I \in mediators_P, \{I, \frac{w_L(I)}{N_P}\}$
17:     Send data to $P$ at rate $\sim \frac{Reputation_P}{N}$
18: done
```

Figure 8: The default one hop servicing policy.

Computing reputations: The value of a one hop reputation for peer B from the perspective of a peer A is determined by three factors: 1) the volume of data exchanged between A and B (if any), 2) A's valuation of B's attesting intermediaries, and 3) B's reputation with each attesting intermediary. We denote A's valuation of intermediary I as $w_A(I)$ and the valuation of peer B at intermediary I by $v_I(B)$, defining these precisely in Equations 1 and 2, respectively.

$$w_A(I) = \frac{n(A \leftarrow I) + n(A \overset{I}{\leftarrow} *)}{n(A \rightarrow I) + n(A \overset{I}{\rightarrow} *)} \tag{1}$$

$$v_I(B) = \frac{n(* \overset{I}{\leftarrow} B) + n(I \leftarrow B)}{n(* \overset{I}{\rightarrow} B) + n(I \rightarrow B)} \tag{2}$$

These expressions allow us to define the indirect reputation value of a peer B from the perspective of a peer A, $\text{ivalue}_A(B)$, given a set of mutually recognized intermediaries, $\mathbf{I}$, as:

$$\text{ivalue}_A(B) = \frac{\sum_{I \in \mathbf{I}} w_A(I) \times v_I(B)}{|\mathbf{I}|} \tag{3}$$

If two peers have a bidirectional relationship, the direct reputation value, $\text{dvalue}_A(B)$, is defined as:

$$\text{dvalue}_A(B) = \frac{n(A \leftarrow B)}{n(A \rightarrow B)} \tag{4}$$

Figure 8 shows how servicing decisions are made regarding a set of peers requesting data. Our default policy uses direct observations if they exist, relying on one hop indirection only if local history is unavailable (lines 2–4). This gives peers an incentive to operate as an intermediary since doing so increases their value across all peers directly, removing the need to rely on one hop coverage and preempting other peers that need to compete based on indirect evaluation. If indirection is required, we use a randomly chosen subset[1] of shared intermediaries to mediate transfers (line 6). Attribution receipts are requested for each mediator (lines 7–9) before computing indirect reputation. After reputations have been computed, requests can be serviced. We impose a reputation threshold to limit contribution imbalance (line 14). Selected peers receive Attribution messages indicating the fraction of throughput to account for each mediator (line 16), normalized by the weight of all mediators (line 15). Servicing rates are assigned proportionally based on relative reputation (line 17), normalized (line 13) across the set of selected peers.

Top K membership: Our default policy for populating top K sets is based on the number of direct and indirect observations of each potential intermediary. When a client directly observes a peer, its occurrence count is incremented by one. In addition to direct observation, our default policy also integrates indirect observations in the form of top K sets from peers. In this case, occurrence counts are updated fractionally, weighted by the number of received bytes from the peer reporting an observation relative to others in the recent past.

If an intermediary is unavailable or refuses an update message, its occurrence count is reduced by 20% or 2, whichever is larger. This AIMD policy is intended to promote agreement on intermediaries with wide coverage while quickly pruning popular peers that become overwhelmed or unavailable. Overhead concerns are treated further in Section 4.1.

Liquidity: Peers have an incentive to keep in good standing with intermediaries that have high coverage. Peers gain standing with a popular intermediary by either satisfying its direct requests (direct contribution) or contributing to peers that have satisfied the intermediary's requests one hop removed (indirect contribution). In the former case, there is a net increase in the sum total of reputation values at the intermediary. In the latter case, the reputation of one peer is simply transferred to another. Thus, the sum total of reputation values at an intermediary—the *liquidity* the intermediary provides the system—is limited by the intermediary's demand. This can result in a disabling shortage. Two peers may share many intermediaries that cannot be used because of a lack of standing with those intermediaries, reducing the *effective* coverage of otherwise popular intermediaries. This situation will arise unless popular intermediaries generate enough demand to allow one hop trading in satisfying their requests to cover the remainder of demand in the system.

To address this problem, the demand recorded in attestation receipts obtained for direct contributions is inflated by a fixed amount by all intermediaries. Because the inflation factor depends on the fraction of total demand generated by popular intermediaries, its value is workload dependent. In our traces, the most popular 2000 peers account for 1.6% of total demand, suggesting that an inflation factor of 100 provides sufficient liquidity.

Although the demand of the minority of popular users relative to total user demand varies little over the course of our trace, this may not be a reliable workload characteristic. If fixed, a static inflation factor will suffice to maintain sufficient liquidity even as users join and leave the system. If not, intermediaries will need to adjust their value at the cost of introducing true economic inflation into the system. This requires only a policy change. Intermediaries can mint receipts with higher or lower byte values which peers can recognize and incorporate into their valuation of intermediaries.

Intermediary incentives: In addition to providing sufficient liquidity, inflating the value recorded in attesting receipts also creates an incentive to serve as an intermediary. In the common case that two peers have not directly interacted, their valuation of one another is based on standing with popular intermediaries, trading in indirect attestations 1:1, i.e., 1 byte contributed for 1 byte attested. But, satisfying an intermediary's requests *directly* results in a 1:N exchange, where $N > 1$ is the inflation factor of intermediary receipts. Because of their higher returns, peers prioritize the requests of popular intermediaries. This preferential treatment requires an intermediary to continue mediating transactions: if it stops responding to queries and updates, it will be pruned from the set of preferred intermediaries by peers that it ignores.

3.2.2 Alternate policies

A large body of work on P2P reputation systems has documented a well-known set of challenges for incentive design. These include bootstrapping new users, Sybil attacks, collusion, and free-riding. To date, no comprehensive solution has emerged that addresses all of these issues, nor do we claim that our approach does. Instead, we have explicitly designed our system to separate the protocol mechanisms of reputation propagation and maintenance from the policy for acting on that information. As a result, our scheme supports a range of policies operating at different levels of vulnerability to well-known attacks. Our measurements of BitTorrent suggests that vulnerability to attack is a negative attribute, but not necessarily a fatal one. In this section, we detail several of these policies and the risks they carry.

Direct, deficit 1 block-based tit-for-tat: This is the most conservative policy we consider, ignoring most available information in the interest of (near) strategy-proof operation. Peers make the positive first step in the

[1] Size 10 in our implementation, see Table 2.

traditional tit-for-tat game, sending at most one unreciprocated data block to a peer. For strategic adversaries, attacks are limited. Free-riders obtain at most one block, and while they might collect many blocks by repeating the game with a large number of peers, our measurements show that most swarms have hundreds of peers or fewer while most objects are comprised of thousands of data blocks. Sybil attacks are similarly frustrated. Collusion carries little benefit—the valuation of a peer is based only on the directly observed behavior of that peer. Bootstrapping is straightforward, as adherents to this strategy willingly contribute the single data block needed to start playing the game. Finally, unfairness in data exchanged is sharply bounded per-peer. The strategic robustness of this approach comes primarily at the cost of a lack of long-term incentives; seeds would limit their contribution to just one block, further reducing availability.

Direct, volume-based tit-for-tat: This strategy eliminates the bound on unfairness in block deficit tit-for-tat. Peers contribute their full capacity, realizing that free-riders / Sybil identities might never reciprocate and seed contributions may never be repaid due to the small chance of repeat interactions. Willingness to make such contributions increases utilization while retaining the collusion resistance of local reasoning as in deficit tit-for-tat. However, because repeat interactions are infrequent, long-term contribution incentives remain weak.

Indirect contribution, reputation $> 1.0 - \epsilon$: This is our default policy. The value of ϵ controls the level of indirect imbalance a peer is willing to tolerate. However, because one hop reputations do not provide precise global accounting, peers contributing data due to third-party standing accept the risk that the intermediaries they choose to mediate an exchange may not have wide coverage. But, as we will show in Section 4, most one hop interactions can be mediated by multiple intermediaries with wide coverage for observed workloads.

Indirect / random excess contribution: For many types of strategic behavior, e.g., free-riding, limiting damage depends on reducing contributions when the reputation of a peer cannot be reliably ascertained. Much like the utilization / robustness tradeoff of deficit n tit-for-tat, peers are faced with a choice when considering what to do with any excess capacity that remains after servicing all requests based on one hop information. Continuing to service requests—essentially at random—enables free-riding behavior, with its effectiveness growing as the amount of random contributions increases.

3.3 Attacks and defenses

Permitting indirection greatly expands the range of attacks available to a strategic client or a set of colluding clients. We consider several attacks, but do not claim to make indirect contribution under our default policy fully resistant to all forms of strategic or malicious behavior. For increased robustness, clients are free to adopt an alternate policy that is more conservative.

- *Intermediary collusion to promote peers:* A popular intermediary may collude with peers or with Sybil identities by providing falsely generated attestation receipts. The effectiveness of this attack is ameliorated by the need for the intermediary to have contributed widely to become popular in the first place. In a sense, its good standing with others is its own reputation to lose, and if it does not continue to directly maintain its standing with enough users through continued contributions, the value of good standing with it will diminish, similarly diminishing the value of its falsified receipts.
- *Peer collusion to promote intermediaries:* Because peers prioritize the requests of popular intermediaries in order to gain receipts with high coverage, colluders may attempt to promote a manufactured identity that has not contributed widely. However, our one hop restriction requires directly verifiable contribution to carry out this deception. Because the integration of external top K sets with a peer's local history is weighted by the contributions of the reporting peer, members of the colluding set must "pay" an unknown amount for the promotion of their fraudulent intermediary, balancing the uncertain returns of the scheme against the initial contributions required to carry it out.
- *Peer collusion to defraud intermediaries:* A peer may collude with others or with Sybil identities to report false contributions to inflate standing with popular intermediaries. For example, a peer A that has legitimately contributed 1 MB to popular intermediary I could falsely report contributions of 1 GB to colluding identity B, generating 1 GB worth of indirect contribution through I for peer A. Our default intermediary policy simply disallows the case of negative net contribution enabling this attack, meaning that A can transfer the attribution of its 1 MB worth to B but cannot exceed the amount of data that I can directly verify A has contributed.

4 Evaluation

Comprehensive evaluation of incentive systems is often frustrated by an overabundance of metrics. For example, protocol designers can choose among fairness, utilization, bootstrapping time, overhead, and resistance to strategic or malicious behavior. As we observed in Section 3.3, optimizing for one of these metrics often comes at the expense of another. Further, evaluating the ability of an incentive strategy to promote long-term incentives—a necessity for increasing availability in P2P systems—requires a model of user behavior that is itself

Value	Definition
2000	The number of peers in a top K set
10	The maximum number of potential intermediaries in overlapping top K sets used to mediate transfers
10 MB	Data exchanged before intermediary synchronization updates
100	The multiplicative factor of reported bytes in intermediary receipts
0.1	ϵ bound on reciprocation

Table 2: One hop reputation parameters and values used in our evaluation.

long-term.

In this section, we provide a threefold analysis of our system to partially address these evaluation challenges. First, we describe the key parameters of our prototype implementation. These parameters control overhead, which we evaluate for our observed workload. Second, we use trace data to examine the coverage of one hop reputations; coverage distills many existing metrics for the quality of reputation systems including bootstrapping time, susceptibility to free-riding, and return on investment. Finally, we report experimental results obtained on PlanetLab using a prototype implementation of our system that we have layered on top of the popular Azureus BitTorrent client. Our PlanetLab experiments measure the real-world performance improvement that can be obtained by using the additional information one hop reputations provide.

4.1 Implementation parameters and overhead

Section 3 describes how reputation information is propagated and used by peers, but we have deliberately delayed assigning several workload-dependent parameters to provide context. These parameters and the values assigned in our prototype are listed in Table 2. We consider the influence of each of these parameters.

Top K set size: The exchange of top K sets serves two purposes. First, it allows peers to agree on shared intermediaries for data exchange. Second, it allows peers to quickly learn which intermediaries have wide coverage (and are therefore most valuable). When K is large, peers have a higher chance of discovering shared intermediaries and information about intermediaries is propagated more rapidly. In our implementation, peers exchange top K sets of size 2000. As each entry in the set is a 128 bit public key identifying an intermediary and a gossip bit, bidirectional exchange requires less than 64 kilobytes of data per peer connection, a small fraction of the megabytes of object data typically transferred between peer pairs.

Synchronization, mediating intermediaries: The intersection of top K sets forms a set of possible intermediaries that can facilitate indirect reciprocation. Of these, peers in our implementation select a random subset of size 10 to act as the mediators for the transfer. Clients synchronize state with intermediaries after every 10 MB transferred. This parameter controls the update burden on the most popular intermediaries, a potential scalability bottleneck.

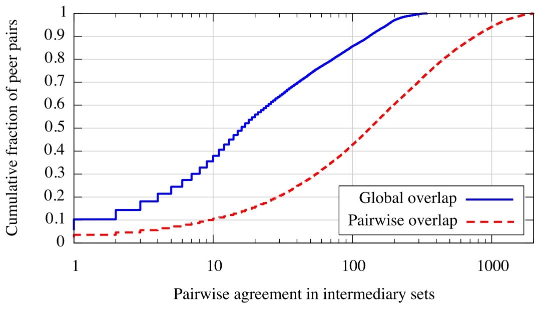


Figure 9: Sizes of overlapping top K sets. Pairwise overlap gives the distribution of overlap sizes for randomly chosen pairs of peers in the `BT-2` trace. Of these intersection sets, global overlap shows the number shared with the top 2000 intermediaries overall.

We measure the number of updates at popular intermediaries using our trace data. Total demand of the 14 million peers in our trace, calculated by counting distinct peers in each swarm and multiplying by swarm file size, is 26,752 terabytes (roughly 2 GB per user). Assuming perfect agreement and static membership in top K sets, each of the most popular intermediaries will need to process 1.4 million updates $\left(\frac{26752\text{ TB}}{2000\times 10\text{ MB}}\right)$. Updates are signed and include a hash (16 bytes), timestamp (4 bytes), 128 bit sender and receiver public keys (32 bytes), and bytes sent and received (16 bytes, 68 bytes total). These updates will be distributed over intermediaries, yielding an overhead of 3 MB per day for each popular intermediary $\left(\frac{1.4\text{ million}\times 68\text{ B}}{31\text{ days}}\right)$. In practice, individual peers will differ in their views of the quality of intermediaries, reducing load, and will also differ in their relative share of total demand and hence update traffic. Also, data exchange between actively downloading peers will be mediated by direct tit-for-tat after the initial exchange, further reducing load. Finally, the size of top K sets can be increased if necessary to further distribute load.

4.2 One hop coverage

For our system to work well, the key factor is whether or not the majority of interactions have a one hop basis for computing reputations. In short, do one hop reputations provide good coverage? We find that they do, arriving at this conclusion using our `BT-2` trace of peer interactions to examine the number of overlapping peers in randomly chosen top K sets, assuming all peers use one hop repu-

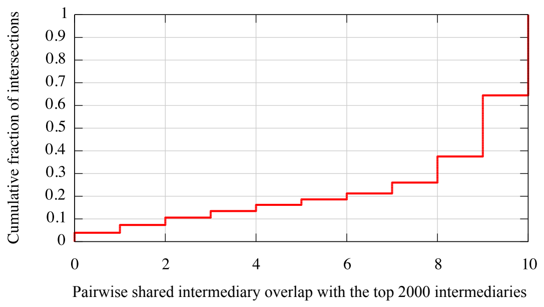


Figure 10: The distribution of intermediary overlap between the top 2000 intermediaries and 10 randomly chosen shared intermediaries between randomly chosen peers. Mediating transfer through a small subset of shared intermediaries suffices to provide wide coverage.

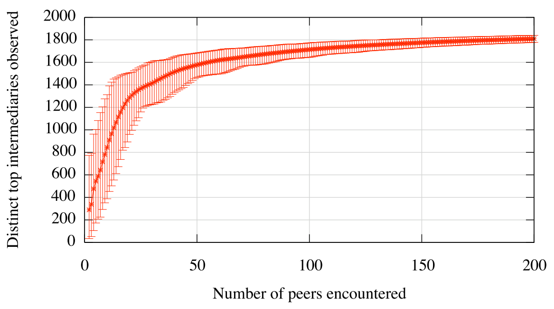


Figure 11: The number of top 2000 intermediaries observed by a new peer as a function of peers directly contacted, averaged over 100 trials with error bars showing the 5th and 95th percentiles.

tations and the parameters of Table 2. Figure 9 shows the number of shared intermediaries between two randomly chosen peers with local histories built up according to our trace. This data indicates the amount of local history that peers can build. Some users participate in only a few swarms, while others participate in hundreds. Applying one hop reputations provides a measure of both coverage and convergence. Coverage is measured by pairwise overlap among the top K sets of randomly matched peers. The median number of shared intermediaries is 83 and more than 99% of peers have at least one common entry in their top K sets. The most useful shared intermediaries are those with wide coverage, and we say that a top K set has converged if it overlaps with the most widely used intermediaries measured over the top K sets of all peers. For some long-lived peers that participate in many swarms, convergence is high, but other peers participate in only a few swarms, limiting their view. When intermediaries with relatively limited coverage mediate transfers, the potential for returns is diminished. Fortunately, our data shows that most randomly matched peers share several intermediaries that are among the 2000 with widest coverage (Global overlap, Figure 9).

Most peers have several choices when deciding which intermediaries to use to mediate their transfers. Using all available intermediaries or only several of the most popular distributes the risk of choosing an intermediary with poor coverage, but contacting potentially hundreds of shared intermediaries per-peer increases overhead. Popular intermediaries that become overloaded may refuse updates from peers that generate too many. To avoid this, we evaluate a default policy of randomly choosing a maximum of ten intermediaries from the shared set for randomly paired peers. Section 4.1 describes how this limit controls overhead. In Figure 10, we examine whether good coverage can be maintained given this limit, finding that even when randomly subsampling shared intermediaries, most interactions will still be mediated through several with wide coverage. 96% of randomly chosen peer pairs share an intermediary that is among the 2000 most popular when randomly subsampling, with a median overlap of size 9.

In the remainder of this section, we describe the implications of high one hop coverage on three properties: bootstrapping time for new users, free-riding, and return on investment for contributions.

Bootstrapping new users: Coverage of popular intermediaries and convergence of top K sets controls the bootstrapping time of one hop reputations. The results of Figures 9 and 10 demonstrate that agreement among top K sets is high, assuming local history built up according to our trace. This includes peers that have participated in the system *at least once*. We next examine how quickly one hop reputations can bootstrap *new* peers that have no local history. Bootstrapping a one hop reputation is a two step process. First, clients need enough observations to ascertain which intermediaries have high coverage. Second, they need to encounter peers that have established relationships with high coverage intermediaries. We consider both aspects.

- *How quickly can a new peer determine intermediary value?* We answer this question statistically using trace data. First, a new identity with an initially empty top K set is created. Next, as in previous experiments, we use our trace data to build up representative top K sets for peers already in the system. We then sample these top K sets randomly, integrating them with that of the newly created identity using randomly assigned weights drawn from our measured end-host capacity distribution of BitTorrent peers. The results of this process are summarized in Figure 11, which gives the number of entries in a new user's top K set that are shared with the 2000 most popular intermediaries globally as a function of the number of peers observed. Data points are averaged over 100 trials with error bars showing the 5th and 95th percentiles. These results demonstrate the rapid bootstrapping of one hop rep-

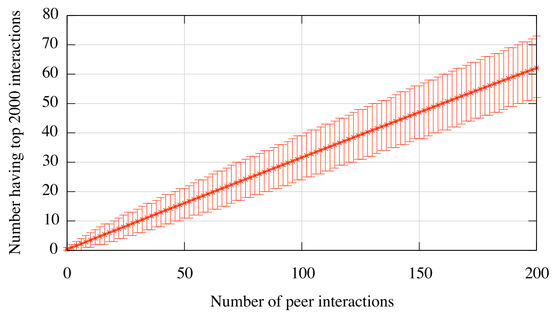


Figure 12: For a new user, the number of peers observed having a direct or one hop relationship with the top 2000 peers as a function of contacted peers.

utations under today's workloads. After only a few dozen interactions with randomly chosen peers, a new user can identify intermediaries with wide coverage.

- *How quickly can peers gain standing with popular intermediaries?* Simply identifying the value of intermediaries does not suffice to enable one hop trading. New peers must encounter and exchange data with popular intermediaries directly or indirectly through others that have directly interacted with them. Figure 12 gives the number of the top 2000 intermediaries in our trace encountered either directly or indirectly by a new user as a function of the number of peers the new user encounters. This data is a conservative bound since we do not model the transfer of standing with the top intermediaries that would occur over time. As with Figure 11, we compute this data statistically, averaging over 100 trials. This data shows that peers observe popular intermediaries either directly or indirectly frequently, allowing a new peer to quickly trade via intermediaries with wide coverage.

Taken together, these results show that new users are likely to both encounter an opportunity to gain standing with a popular intermediary (Figure 12) and recognize that opportunity (Figure 11).

Free-riding: The coverage achieved by one hop reputations for today's workloads suggests that free-riding can be deterred without the significant sacrifices in utilization required by schemes such as deficit 1 block-based tit-for-tat. Because peers usually have a one hop basis for making servicing decisions, the majority of each user's capacity can be allocated to peers that can demonstrate their contributions. However, we do not claim that one hop reputations *prohibit* free-riding, as selfish peers in large swarms may still be able to scavenge enough altruistic excess capacity to complete file downloads. Rather, our goal is simply to limit the opportunities for effective free-riding by providing peers with more information. Under today's reputation systems, the random contributions that enable free-riding are necessary because obtaining information about good peers requires making contributions to all peers, beneficial or not.

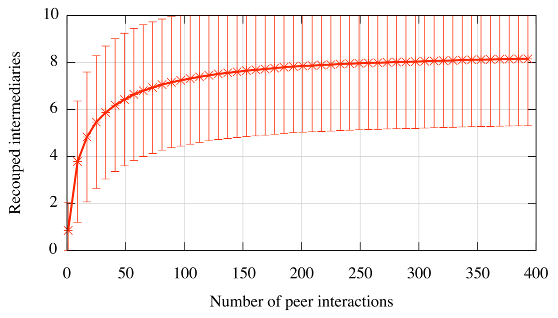


Figure 13: The number of mediating intermediaries from a randomly drawn one hop transfer encountered in subsequent interactions as a function of the number of subsequent interactions. A contributing peers sees higher returns on investment if mediators are chosen that will reappear quickly. Error bars give the standard deviation.

Return on investment: The return on investment for contributed bytes is the amount of reciprocated bytes generated by that contribution. For reputations that persist over short time periods, as in BitTorrent's tit-for-tat, return on investment is immediate and can be measured or computed. For persistent reputation schemes, however, return on investment can be a misleading measure of incentive strength. For instance, volume-based tit-for-tat maintains state regarding each contribution, providing 1:1 returns on all contributions eventually if peers do not leave the system permanently and continue to make requests. But, as we observed in Section 2.4, repeat, direct interactions are extremely rare for today's workloads, suggesting that peers would need to tolerate lengthy delays before receiving reciprocation. Particularly in P2P networks, waiting for reciprocation opportunities dampens returns as some peer departures are permanent.

Because one hop reputations have persistent memory, we evaluate the returns from contribution in terms of the number of interactions required to recoup contributed bytes. Each time a peer contributes data due to indirect, one hop standing, that contribution is mediated through a subset of the shared intermediaries between sender and receiver. The contributing peer earns reciprocation for that contribution only if it later can use some of those intermediaries to mediate another transfer where it acts as receiver. A contributing peer will see poor returns if it selects a mediating intermediary that has poor coverage.

We use the one hop reputations for peers in our `BT-2` trace to compute the number of interactions required to repeat the use of an intermediary from the set mediating a random initial contribution. Figure 13 shows results averaged over 1000 trials with error bars showing standard deviation. For each sample, we compute a size 10 random subset of shared intermediaries between two randomly drawn peers, interpreting this set as the me-

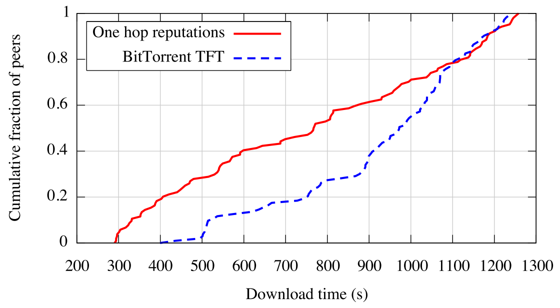


Figure 14: A comparison of performance for bulk file distribution on PlanetLab. Leveraging the historical rate information provided by one hop reputations improves performance.

diating intermediaries in a potential transfer. We then repeat this process, computing the overlap between subsequent mediating sets and the original contribution set. Figure 13 shows that, on average, peers will encounter 8 of the 10 initially used intermediaries within a few hundred peer interactions. Although this implies that returns for one hop contributions are not 1:1 for our default policy, peers do see a higher opportunity for returns on contributions when compared with direct, long-term reciprocation schemes that require tens of thousands of interactions before payback occurs, if at all.

4.3 Deployment on PlanetLab

Our evaluation thus far has focused on the ability of one hop reputations to promote strong contribution incentives through wide coverage and returns on contribution, improving performance by providing users with a reason to contribute more capacity. In this section, we focus on the concrete performance improvement one hop reputations can provide, regardless of strengthened incentives.

Because one hop reputations include not only an accounting of transfers but also the *rate* of those transfers (ref. Table 1), users can make intelligent decisions about which peers are likely to disseminate data rapidly. In BitTorrent today, peers do not maintain historical information about peer capacities and must rely on tit-for-tat to funnel data to high capacity peers. Unfortunately, high capacity goes unnoticed by tit-for-tat when the data available to trade is the limiting factor for performance, as is the case when a file first becomes available or when seed capacity is limited. In both cases, quickly utilizing the full capacity of the swarm depends on high capacity peers receiving data first so they can quickly replicate it, reducing the amount of time required for other peers to gain data to trade.

To measure the performance benefit realized by using rate information, we deployed our prototype one hop reputation implementation on PlanetLab, comparing its performance with the original Azureus BitTorrent implementation on which it is layered. Figure 14 compares the completion times for 100 peers downloading a 25 MB file using BitTorrent in one trial and one hop reputations in the next with simultaneous arrivals in both trials. Before conducting the one hop download trial, we first primed the local histories of participants by distributing a different 25 MB file. We record the download times required to download the second file while using the local history built up during the priming run. To adhere to the skewed bandwidth distribution typical of end-hosts in BitTorrent swarms, we used application level bandwidth capacity limits with values drawn from the percentiles of the end-host capacity distribution for BitTorrent clients given in [12].

One hop reputations improve performance for roughly 75% of PlanetLab hosts, providing a median reduction in download time from 972 seconds to 766 seconds. This performance improvement is attributable to the ability of historical information to allow peers to quickly find good tit-for-tat peerings. This is particularly true for the seed, which distributes data randomly in the reference implementation of BitTorrent. Rather than relying on random selection, a one hop seed can preferentially give data to users it knows have high capacity. These peers amplify the initial contributions of the seed, pumping data into the systems rapidly and increasing utilization relative to that of random selection, which may give initial data to slow peers that cannot quickly replicate it.

5 Related work

Our focus on incentives has led us to build a protocol layer for exchanging peer reputation information, one that can be shared across time and across content distribution applications. While incentives could be added to any content distribution system, it can be quite difficult to design a robust incentive system when participation is ephemeral and identities are not persistent, as we have seen in BitTorrent.

The research community has made considerable progress towards understanding BitTorrent dynamics; we use many of these insights in the design of our one hop reputation system. Qui and Srikant [15] analytically model the BitTorrent protocol, showing that in certain conditions, it achieves a Nash equilibrium. Unfortunately, our measurements show that these conditions are typically not met in practice in live BitTorrent usage. Bharambe et al. [2] use simulation to show that BitTorrent engages in progressive taxation, taking from high capacity peers to give unreciprocated bandwidth to low capacity peers. BitTyrant [12] exploits this observation, showing that clients can strategically deploy their upload bandwidth to significantly improve their local performance, and in the bargain, reduce performance of the swarm. Locher et al. [11] and Sirivianos et al. [16]

made similar points, showing that BitTorrent provides weak protection against free riding clients. Our measurement results of BitTorrent in the wild are compatible with the results of those papers, expanding on previous studies of only a subset of swarms with a large number of active downloaders. Our data is broader, showing that in most BitTorrent swarms incentives are inoperable. Wang et al. [18] and Tribler [13] argue for using third party helpers to improve client performance in BitTorrent. While we show that increased upload contribuion only marginally improves download rates in BitTorrent, instead we generalize the notion of helpers, using one hop reputations to provide an incentive for third parties to do work on behalf of others.

Our work has the most in common with recent work on the design of reputation systems for various P2P applications. Karma [17] focuses on building a robust, incentive compatible distributed hash table as a basis for trading a digital currency. DHTs are a particularly difficult venue for robust incentives, as peers are both ephemeral and have little repeated interaction. To address this, Karma sets up a replicated system of banks on top of the DHT to serve as reputation authorities. In our one hop reputation system, popular nodes serve as a kind of ad-hoc bank without any additional mechanism beyond peer gossip of popular nodes and signed receipts. Our use of indirection is similar in some respects to EigenTrust [9] and multi-level tit-for-tat [21]. EigenTrust focused on the problem of inauthentic files, computing a global reputation for every participant. Reputations in our system are local, and clients are free to evolve their strategy independently over time. Multi-level tit-for-tat demonstrated that much of the value of EigenTrust can be achieved with only a few levels of indirection, an insight we use in the design of one hop reputations.

Finally, we observe that our protocol for propagating reputations allows peers to make their own policy decisions, making it possible for peers to choose among policies for allocating their bandwidth. As such, the one hop protocol may also be able to incorporate a number of previously proposed reputation systems, such as Bayesian estimation [3], PPay [20], PeerTrust [19], among others [5, 7]. However, we must leave the full exploration of these issues to future work.

6 Conclusion

To deliver on their potential benefits, P2P systems need robust contribution incentives. In this paper, we have described the pitfalls undermining currently deployed incentive strategies, finding that decisions based on direct observations and local history will not suffice to overcome the performance and availability problems on which today's P2P networks falter. Our measurements motivate the design of one hop reputations, a protocol for propagating reputations that extends the information peers have available for making servicing decisions. We propose a default policy for the use of this information, finding that for observed workloads, one hop reputations can provide wide coverage and positive, long-term contribution incentives. Through deployment on PlanetLab, we show that one hop reputations can improve short-term download performance for peers as well.

Acknowledgments

We would like to thank the anonymous reviewers and our shepherd, Emin Gün Sirer, for their valuable feedback. This research was partially supported by the National Science Foundation, CSR-PDOS #0720589.

References

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

[2] A. Bharambe, C. Herley, and V. N. Padmanabhan. Analyzing and improving a BitTorrent network's performance mechanisms. In *Proc. of INFOCOM*, 2006.

[3] S. Buchegger and J.-Y. L. Boudec. A robust reputation system for P2P and mobile ad-hoc networks. In *Proc. of IPTPS*, 2004.

[4] B. Cohen. Incentives build robustness in BitTorrent. In *Proc. of P2P-ECON*, 2003.

[5] F. Cornelli, E. Damiani, S. D. C. di Vimercati, S. Paraboschi, and P. Samarati. Choosing reputable servents in a P2P network. In *Proc. of WWW*, 2002.

[6] J. R. Douceur. The Sybil attack. In *Proc. of IPTPS*, 2002.

[7] M. Gupta, P. Judge, and M. Ammar. A reputation system for peer-to-peer networks. In *Proc. of NOSSDAV*, 2003.

[8] T. Isdal, M. Piatek, A. Krishnamurthy, and T. Anderson. Leveraging BitTorrent for end host measurements. In *Proc. of PAM*, 2007.

[9] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina. The eigentrust algorithm for reputation management in P2P networks. In *Proc. of WWW*, 2003.

[10] A. Legout, N. Liogkas, E. Kohler, and L. Zhang. Clustering and sharing incentives in BitTorrent systems. In *SIGMETRICS Perform. Eval. Rev.*, 2007.

[11] T. Locher, P. Moor, S. Schmid, and R. Wattenhofer. Free Riding in BitTorrent is Cheap. In *Proc. of HotNets*, 2006.

[12] M. Piatek, T. Isdal, A. Krishnamurthy, and T. Anderson. Do incentives build robustness in BitTorrent? In *Proc. of NSDI*, 2007.

[13] J. Pouwelse, P. Garbacki, J. Wang, A. Bakker, J. Yang, A. Iosup, D. H. J. Epema, M. Reinders, M. van Steen, and H. Sips. Tribler: A social-based peer-to-peer system. In *Proc. of IPTPS*, 2006.

[14] H. Pucha, D. G. Andersen, and M. Kaminsky. Exploiting similarity for multi-source downloads using file handprints. In *Proc. of NSDI*, 2007.

[15] D. Qiu and R. Srikant. Modeling and performance analysis of BitTorrent-like peer-to-peer networks. In *Proc. of SIGCOMM*, 2004.

[16] M. Sirivianos, J. H. Parka, R. Chen, and X. Yang. Free-riding in BitTorrent networks with the large view exploit. In *Proc. of IPTPS*, 2007.

[17] V. Vishnumurthy, S. Chandrakumar, and E. Sirer. Karma: A secure economic framework for peer-to-peer resource sharing. In *Proc. of P2P-ECON*, 2003.

[18] J. Wang, C. Yeo, V. Prabhakaran, and K. Ramchandran. On the role of helpers in peer-to-peer file download systems: design, analysis, and simulation. In *Proc. of IPTPS*, 2007.

[19] L. Xiong and L. Liu. Peertrust: Supporting reputation-based trust for peer-to-peer electronic communities. *IEEE Trans. On Knowledge and Data Engineering.*, 2004.

[20] B. Yang and H. Garcia-Molina. PPay: Micropayments for peer-to-peer systems. In *Proc of CCS*, 2003.

[21] Q. L. Yu. Robust incentives via multi-level tit-for-tat. In *Proc. of IPTPS*, 2006.