# A Performance Analysis of Incentive Mechanisms for Cooperative Computing

Carlee Joe-Wong,[*] Youngbin Im,[†] Kyuyong Shin,[‡] and Sangtae Ha[†]
[*]Department of Electrical and Computer Engineering, Carnegie Mellon University
[†]Department of Computer Science, University of Colorado at Boulder
[‡]Department of Computer Science, Korea Military Academy
Emails: cjoe@princeton.edu, {youngbin.im,sangtae.ha}@colorado.edu, kyshin@kma.ac.kr

***Abstract*—As more devices gain Internet connectivity, more information needs to be exchanged between them. For instance, cloud servers might disseminate instructions to clients, or sensors in the Internet of Things might send measurements to each other. In such scenarios, information spreads faster when users have an incentive to contribute data to others. While many works have considered this problem in peer-to-peer scenarios, none have rigorously theorized the performance of different design choices for the incentive mechanisms. In particular, different designs have different ways of "bootstrapping" new users (distributing information to them) and preventing "free-riding" (receiving information without uploading any in return). We classify incentive mechanisms in terms of reciprocity-, altruism-, and reputation-based algorithms, and then analyze the performance of these three basic and three hybrid algorithms. We show that the algorithms lie along a tradeoff between fairness and efficiency, with altruism and reciprocity at the two extremes. The three hybrids all leverage their component algorithms to achieve similar efficiency. The reputation hybrids are the most fair and can nearly match altruism's bootstrapping speed, but only the reciprocity/reputation hybrid can match reciprocity's zero-tolerance for free-riding. It therefore yields better fairness and efficiency when free-riders are present. We validate these comparisons with extensive experimental results.**

## I. Introduction

As more devices join the Internet of Things, more information is being exchanged between users and devices embedded in a network [1]. For example, a central server might distribute instructions among multiple devices, or devices could exchange measurement data with each other. While a central server can send this information to all users, it is more efficient for users to participate as well: when a user receives information, (s)he can pass it on to other users [2]. User participation is particularly important when there are many users and a large amount of information to be sent, e.g., distributing large software updates.

Operators of information exchange networks face a challenge in deciding how users should be incentivized to exchange information. Indeed, many works have proposed different incentive mechanisms [3]–[8]. Yet these incentive mechanisms differ in their emphases on various desirable properties, e.g., some prioritize efficiency, or receiving information quickly. Others emphasize fairness, or ensuring that users receive as much information as they contribute [9], [10].[1] Prior works on information exchange have even shown that there are fundamental tradeoffs between desirable mechanism properties like fairness and efficiency [10]–[12]. Thus, in designing an incentive mechanism for their users, operators often need to prioritize possibly conflicting objectives, and then employ the incentive mechanism that best satisfies their priorities.

Existing works on performance tradeoffs for incentive mechanisms generally focus on the limits of possible performance tradeoffs, instead of comparing the performance of specific mechanisms in the design space [10]–[12]. Other works have evaluated these mechanisms empirically [13]–[15], but do not provide a rigorous analytical comparison. Our work thus fills a gap in the literature by *characterizing the design space of incentive mechanisms* and *rigorously comparing mechanisms within this design space*. We therefore provide a guide that operators can use to choose the incentive mechanisms that achieve their desired performance tradeoffs in real systems. To do so, we classify and compare six representative incentive mechanisms across several performance metrics.

Two of the most important metrics that we consider are efficiency and fairness. While efficiency ensures that the system in general performs well (i.e., information spreads quickly), fairness incentivizes users to contribute to the system in the first place: otherwise, users could passively receive data from others without contributing any data to the system. Without a well-designed incentive mechanism, the energy and processing costs of contributing data give individuals an incentive to *free-ride* [16]–[19], which can cause a system collapse if no users contribute data to each other.

Strictly enforcing fairness, however, often reduces efficiency and can also cause a system collapse [10]. In particular, new users are often unable to initially contribute to the system since they do not yet have information to send to others. In order to "bootstrap" these users, many incentive mechanisms use a form of altruism, or allowing newcomers to receive pieces of data without requiring them to reciprocate these downloads. Unsurprisingly, this bootstrapping can lead to free-riding and a loss of fairness [12]. Thus, different incentive mechanisms are generally designed to achieve different levels of fairness, efficiency, bootstrapping speed, and free-riding susceptibility.

We rigorously quantify the performance of six specific incentive mechanisms. We consider the metrics of fairness, efficiency, bootstrapping, and free-riding; and *prove* that a few

[1]We take "contribute" to mean the amount of information that users send to others, regardless of whether they originally collected this information.

fundamental classes of incentive mechanisms yield different performance tradeoffs. Hybrid mechanisms can improve these tradeoffs, but most hybrids inherit some free-riding susceptibility from their component algorithms. We make the following contributions:

- In Section III, we classify different incentive mechanisms based on three classes of exchange algorithms: altruism, reciprocity, and reputation. We illustrate the classification by describing three hybrid algorithms that combine elements of the basic classes.
- In Section IV, we model the performance of the three basic and three hybrid exchange algorithms. We use our model to compare and explain the algorithms' expected fairness, efficiency, bootstrapping, and free-riding. While altruism is the most efficient, it is also less fair and more susceptible to free-riding. At the other extreme, direct reciprocity is fair and does not permit free-riding, but also very inefficient. The hybrid algorithms leverage their component algorithms to be both fair and efficient, but only the reciprocity/reputation hybrid can match reciprocity's resilience to free-riding.
- We validate Section IV's results with extensive experiments in Section V. We show that the algorithms' performance matches our model's predictions, and that free-riding compromises fairness and efficiency for susceptible algorithms. The reciprocity/reputation hybrid's resilience to free-riding allows it to maintain high fairness and efficiency in the presence of free-riders.

We conclude the paper in Section VI.

## II. Related Work

Most works on incentive mechanisms for exchanging information consider either theoretical limits of performance tradeoffs or propose specific exchange algorithms. We instead classify, evaluate, and compare *existing* incentive mechanisms at the limits of the incentive mechanism design space.

**Metrics for information spread.** Many works on information dissemination focus on efficiency, i.e., the speed of propagation through various network topologies [20] in different contexts [21], [22]. The particular context of sensor networks considers tradeoffs between energy usage, efficiency [23], [24], and fairness, i.e., ensuring that nodes contribute equally to the system [9]. Works on peer-to-peer (P2P) algorithms consider tradeoffs between efficiency, as measured by file download times, and fairness, or the amount of data that users receive, relative to the amount they contribute [10], [11]. Enforcing fairness incentivizes users to contribute to the system [2], but can severely reduce efficiency due to the need to bootstrap new users [12]. Bootstrapping, on the other hand, often opens up an opportunity to free-ride [16]–[19].

**Comparing incentive mechanisms.** BitTorrent and BitTorrent-like systems are compared empirically in [13]–[15], but without a rigorous analysis of their expected performance in general scenarios. Other works compare a wider range of existing P2P algorithms, e.g., through taxonomies and simulations [25], but again without an analytical performance comparison. A descriptive survey of trust- or reputation-based P2P algorithms is presented in [26].

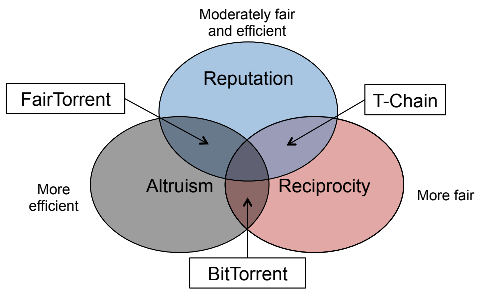


Fig. 1: Classification and expected performance of incentive mechanisms for exchanging information.

**Designing incentive mechanisms.** One of the most popular incentive mechanisms for exchanging information is BitTorrent [3], whose performance has been well-studied both theoretically [11], [27], [28] and empirically [13]–[15]. While BitTorrent is generally fair and efficient, it suffers from free-riding [18], [19]. Variants of BitTorrent, e.g., PropShare [5] and BitTyrant [6], attempt to reduce this free-riding.

BitTorrent is based on reciprocity, i.e., users upload more data to those users that have given them the most data. Thus, BitTorrent enforces fairness: users receive approximately as much as they have contributed. Many other P2P incentive mechanisms take a more indirect approach to fairness. For instance, in T-Chain, users employ encryption to ensure that data uploads are reciprocated [8], allowing them to trust each other's good behavior. In reputation systems, history is used to establish trust [4], [29]–[32], e.g., Give-to-Get allows users to preferentially upload to those who have given them more data in the past [29]. However, reputation-based mechanisms tend to have difficulty bootstrapping new users. In the next section, we introduce six representative incentive mechanisms, whose performance we analyze in the rest of the paper.

## III. Algorithms Compared

We consider a network of users, each of which can upload to or download from each other user. Each user wishes to collect a full set of data from other users and a central "seeder," and this data is divided into discrete *pieces*. Users download missing data from other users and upload data in return. We focus in this paper on the algorithms that determine to which user(s) each user sends data. These double as incentive mechanisms that encourage users to contribute.

### A. Algorithm Descriptions

We consider three classes of exchange algorithms: reciprocity-, altruism-, and reputation-based [8]. *Reciprocity* algorithms require that users reciprocate whenever they receive data; thus, they ensure that users upload exactly as much data as they download, enforcing fairness. *Altruism* requires users to upload to randomly selected users, with no attempt at reciprocity. Finally, *reputation* algorithms indirectly enforce reciprocity by requiring users to upload to those with the highest reputations, based on past (global) behavior. We interpret this preference probabilistically [4]: the probability of uploading to another user is proportional to the total number of pieces uploaded by that user to any other user. Bootstrapping in the reputation algorithm, as in EigenTrust [4], is accomplished by reserving a small fraction of bandwidth for altruism. Figure 1 visualizes this classification and introduces three hybrid algorithms that we describe below.

**Reciprocity/altruism:** We consider a hybrid of these algorithms in which a fixed amount (e.g., 80%) of users' upload bandwidth is reserved for reciprocity, which is enforced in a series of discrete timeslots. In each timeslot, this bandwidth is used to upload data to a given number of users from which the user has received the most data in the previous timeslot. The remaining bandwidth is used for altruism, allowing existing users to bootstrap newcomers. Once they receive pieces, newcomers can begin to participate in direct reciprocity. BitTorrent [3] is an example of this class of algorithm.

**Reputation/altruism:** In this hybrid algorithm, each user maintains a deficit counter of the total number of pieces uploaded to, less those received from, each other user. These counters function as local reputation scores: users always upload to the client with the smallest deficit counter, i.e., from whom they have received the most pieces without reciprocation. However, if all deficit counters are nonnegative, users upload to randomly chosen users with zero reputations, including newcomers. Thus, the contributing users effectively engage in altruism, uploading pieces with no expectation of reciprocity. FairTorrent [7] is an example of such an algorithm.

**Reciprocity/reputation:** Users in this hybrid algorithm can reciprocate uploads by uploading a piece to any user. If the receiving user reciprocates to the uploading user, we refer to the exchange as *direct reciprocity*; reciprocating to another user is called *indirect reciprocity*. Through indirect reciprocity, newcomers can receive a piece from one user and reciprocate by uploading the received piece to another user. T-Chain [8] uses this class of hybrid algorithm. T-Chain users upload encrypted pieces to others to ensure that uploads are reciprocated, and only release the decryption keys after confirming that the receiving user has reciprocated.

## B. Expected Algorithm Performance

We consider four different performance metrics: fairness, efficiency, bootstrapping speed, and susceptibility to free-riding. Figure 1 shows our qualitative expectations for the different algorithms' performance, which we explain below. We analyze their performance in detail in Section IV.

**Fairness and efficiency:** A fair algorithm ensures that each user receives as much as it contributes, while an efficient one minimizes the average time required to finish downloading the file. Efficiency will generally decrease as fairness increases ( [10], cf. Lemma 1). We thus expect that reciprocity will be the most fair, though in practice it is so inefficient that fairness cannot be defined (cf. Section IV-A). Altruism will likely be the most efficient and least fair: altruistic uploads ensure that each user receives a similar download rate regardless of its upload capacity, decreasing fairness. Reputation systems will likely lie between these extremes: users increase their reputations by uploading more file pieces, making them more likely to receive downloads from others. Fairness is thus encouraged, though not enforced as strictly as in reciprocity.

**Bootstrapping speed:** Bootstrapping is another form of efficiency: more efficient systems will distribute resources to newcomers faster, allowing these newcomers to begin exchanging pieces and contributing to the system sooner. Thus, we expect that more efficient algorithms (altruism, followed by reputation and reciprocity) will also bootstrap users faster.

**Susceptibility to free-riding:** More susceptible algorithms allow free-riders to receive data from others without uploading any themselves, i.e., they will likely be less fair. Free-riders can either collect free resources from other users or collude to trick legitimate users into uploading data to them. We expect that altruism will be more vulnerable to non-collusive free-riding, since altruistic uploads are given freely with no reciprocity expectation. Reputation algorithms will likely be more vulnerable to collusion, as colluders can falsely inflate each others' ratings to raise their reputations. Reciprocity algorithms do not allow free-riding.

## IV. Performance Analysis

We consider the algorithms' fairness and efficiency in Section IV-A, bootstrapping speed in Section IV-B, and susceptibility to free-riding in Section IV-C. We assume that there are $N$ users in total, each of which can upload to each other user, and let $U_i$ denote the upload bandwidth capacity of the $i$th user, with $U_1 \geq U_2 \geq \ldots \geq U_N$. We assume that $U_i \leq \sum_{j \neq i} U_j$ for all users $i$, so that no single user possesses a disproportionate amount of the total capacity. This constraint allows us to guarantee fairness for some of the algorithms. We use $u_i \leq U_i$ to denote the actual upload bandwidth of the $i$th user with the different algorithms, and $d_i$ to denote the download rate (i.e., bandwidth); each user also receives an expected bandwidth $u_S/N$ from a seeder(s). Note that the total upload and download rates over all users must be equal:

$$u_S + \sum_{i=1}^{N} u_i = \sum_{i=1}^{N} d_i. \tag{1}$$

## A. Fairness and Efficiency

We first consider the fairness and efficiency of each algorithm. We measure **efficiency** by the average download time over all users, which we can approximate as

$$E = \sum_{i=1}^{N} \frac{1}{N d_i} \tag{2}$$

assuming that the download rates are in equilibrium over time and normalizing to a unit file size.[2] **Fairness** measures the discrepancy between the numbers of pieces uploaded and received by each user. Thus, we define the fairness for each

[2]While we could instead define "efficiency" in terms of the upload bandwidths, i.e., $E = \sum_i u_i$, doing so may not maximize the download times (2). We show below that five of the six algorithms maximize the total upload bandwidth, but yield suboptimal download times (Corollary 1).

TABLE I: Expected download rate for user $i$ in equilibrium with perfect piece availability and no free-riders. BitTorrent and the reputation algorithm are respectively assumed to allocate fractions $\alpha_{BT}$ and $\alpha_R$ of their bandwidth for altruism.

| **Algorithm** | **Download utilization** $(d_i - u_s/N)$ |
|---|---|
| Reciprocity | $0$ |
| T-Chain | $U_i$ |
| BitTorrent | $\frac{1-\alpha_{BT}}{n_{BT}} \sum_{j=\lfloor \mathrm{mod}(i,n_{BT})\rfloor+1}^{\mathrm{mod}(i,n_{BT})+n_{BT}} U_j + \alpha_{BT}\frac{\sum_{k=1,k\neq i}^{N} U_k}{N-1}$ |
| FairTorrent | $U_i$ |
| Reputation | $U_i \sum_{j=1,j\neq i}^{N} \frac{(1-\alpha_R)U_j}{\sum_{k=1,k\neq j}^{N} U_k} + \alpha_R \frac{\sum_{k=1,k\neq i}^{N} U_k}{N-1}$ |
| Altruism | $\frac{\sum_{k=1,k\neq i}^{N} U_k}{N-1}$ |

user $i$ as $f_i = d_i/u_i$, i.e., the ratio of download to upload rates; more fair algorithms have $f_i$ near 1. We define the system-wide fairness as the average absolute values of the $\log(f_i)$:

$$F = \frac{1}{N}\sum_{i=1}^{N}\left|\log\left(\frac{d_i}{u_i}\right)\right|. \qquad (3)$$

The closer the $d_i/u_i$ are to 1 and $F$ is to 0, the more fair the system; $F = 0$ if and only if $d_i = u_i$ for all $i$. A positive $F$ indicates that some users are receiving or uploading disproportionate amounts of bandwidth. We thus see that there is a fundamental tradeoff between fairness and efficiency:

*Lemma 1 (Optimal fairness and efficiency):* To achieve $F = 0$, we should have $u_i = d_i$ for each user $i$. However, the maximum value of $E$ subject to (1) is achieved when all users upload with full capacity, $u_i = U_i$, and have the same download rates $d_i = \sum_i U_i/N + u_S/N$.

We now analyze the algorithms' fairness and efficiency in two scenarios: first an idealized equilibrium in which all users need pieces from each other (i.e., perfect piece availability), and then a more realistic scenario that considers the probability that a given user needs a piece from another user.

*1) Idealized Equilibria:* We first characterize the algorithms in terms of their upload and download rates for each user and then use these to compare the fairness and efficiency.

*Lemma 2:* All users upload with their full capacity $U_i$ in equilibrium with perfect piece availability, except for reciprocity users, who do not upload anything.

We now use the results of Lemma 2 to find the equilibrium download rates of each algorithm:

*Proposition 1:* Table I shows the download rates of each user $i$ in equilibrium with perfect piece availability, assuming no free-riders.

To derive these results, we suppose that users' reputations, which are determined by the numbers of pieces that they upload, are proportional to their upload bandwidth; this assumption likely holds with perfect piece availability. We also define $n_{BT}$ as the number of users to which BitTorrent reciprocally uploads data at any given time, and we assume that altruistic users are equally likely to upload to any other user.

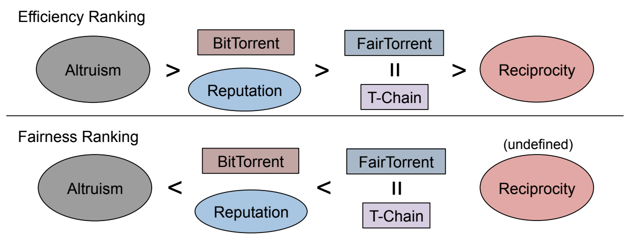


Fig. 2: Fairness and efficiency ranking of the six algorithms in the idealized scenario (Corollary 1).

From Table I, we find that most algorithms emphasize fairness in ideal conditions, though some put more emphasis on efficiency. Reciprocity in particular emphasizes fairness to such an extreme degree that users do not upload to each other at all: no upload can be initiated because a reciprocal download is not guaranteed. BitTorrent, altruism, and reputation are the only algorithms that are not perfectly fair, but they are more efficient. However, no algorithm is perfectly efficient:

*Corollary 1 (Comparing fairness and efficiency):* In equilibrium with perfect piece availability, only T-Chain and FairTorrent achieve the optimal fairness.

When all users' upload rates are sufficiently similar, i.e., $\sum_{j=1}^{N} U_j \gg U_i$ and $U_i \approx U_{i+n_{BT}}$ for any user $i$, altruism achieves the highest (but still sub-optimal) efficiency. BitTorrent and the reputation algorithm are more efficient than T-Chain or FairTorrent.

Figure 2 summarizes Corollary 1's results. Comparing them to Figure 1, we see that the results generally match our predictions in Section III-B: altruism is the most efficient and least fair algorithm, and reciprocity the least efficient. The reputation and hybrid algorithms lie in between. Surprisingly, BitTorrent is more efficient than FairTorrent, despite having some elements of reciprocity, and FairTorrent is just as fair as T-Chain, despite FairTorrent's use of some altruism.

*2) Piece Availability:* Since perfect piece availability does not occur in real systems, users' actual download rates will differ from those in Table I. Efficiency is particularly affected: users' download rates will suffer if they cannot find others with the pieces they need. We model the piece availability with two factors: the number of pieces held by each user and the distribution of the pieces among the users (i.e., which pieces are held by more users). We suppose that users are equally likely to have a given piece, e.g., as achieved in local-rarest-first piece selection [27]. We use $p_k$ to denote the probability that a given user has exactly $k$ pieces.

To evaluate the impact of piece availability on the different algorithms, we find the probability that a user $j$ can upload a piece to user $i$ under each algorithm. A lower probability that two users will exchange data generally implies a lower efficiency, as defined in (2): even if all users utilize their total upload capacity, users that are less able to exchange data with each other will experience lower overall download rates.[3] We consider the effect of these constraints on the reciprocity and altruism algorithms, as well as their hybrids, before showing that the reputation algorithm can achieve poor fairness and efficiency when piece availability is considered.

We first consider the reciprocity algorithm. With reciprocity alone, user $j$ cannot upload to user $i$, as neither can initiate an exchange (cf. Proposition 1's proof in the Appendix).

Users can initiate piece exchanges in the reciprocity algorithm's hybrids, BitTorrent and T-Chain. If such an exchange has been initiated, user $j$ continues to upload to user $i$ with direct reciprocity only if both users each need at least one of the other's pieces. Letting $m_j$ and $m_i$ denote the numbers of pieces that users $j$ and $i$ have respectively, the probability that $j$ and $i$ can exchange pieces with direct reciprocation is

$$\pi_{DR}(j,i) = q(i,j)q(j,i) = 1 - \frac{\binom{M-\min(m_i,m_j)}{\max(m_i,m_j)-\min(m_i,m_j)}}{\binom{M}{\max(m_i,m_j)}}, \tag{4}$$

where $M$ denotes the total number of pieces and $q(i,j)$ the probability that user $i$ needs at least one piece from user $j$:

$$q(i,j) = \begin{cases} 1 & \text{if } m_i < m_j \\ 1 - \frac{\binom{M-m_j}{m_i-m_j}}{\binom{M}{m_j}} & \text{if } m_i \geq m_j \end{cases} \tag{5}$$

If many users arrive simultaneously in a flash crowd, most users will have few pieces and reciprocity may perform poorly. In the extreme case of $m_i$ or $m_j = 0$, $\pi_{DR}(j,i) = 1 - 1/\binom{M}{\max(m_i,m_j)} = 0$: users cannot exchange pieces unless each has at least one piece. We elaborate on this restriction in Section IV-B's discussion of bootstrapping newcomers.

T-Chain and BitTorrent supplement the probability of direct reciprocity occurring with options for indirect reciprocity and altruism respectively, leading us to conclude the following:

*Proposition 2 (Piece exchange probabilities):* The probabilities that user $j$ can upload to user $i$ for T-Chain and BitTorrent are respectively

$$\pi_{TC}(j,i) = q(i,j)q(j,i) + q(i,j)(1-q(j,i)) \times \left(1 - \left(1 - \sum_{l=1}^{M} p_l q(j,l)\left(1-q(l,j)\right)\right)^{N-2}\right) \tag{6}$$

$$\pi_{BT}(j,i) = q(i,j)\left((1-\alpha_{BT})q(j,i) + \alpha_{BT}\right) \tag{7}$$

where $\alpha_{BT}$ denotes the fraction of BitTorrent's bandwidth allocated for optimistic unchoking. Also, $\pi_{TC} \geq \pi_{BT}$ if

$$\alpha_{BT} \leq 1 - \left(1 - \sum_{l=1}^{M} p_l q(j,l)\left(1-q(l,j)\right)\right)^{N-2}. \tag{8}$$

[3]This quantification of piece availability is inspired by the quantification of file sharing effectiveness in [27].

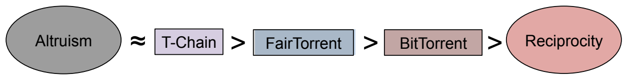


Fig. 3: Efficiency comparison with piece availability.

Note that these probabilities do not necessarily represent the probability that user $j$ *will* upload to user $i$; they represent the *feasibility* of doing so, upload capacity permitting. We see that as $N$ increases, $\pi_{TC} \geq \pi_{BT}$ for sufficiently large $\alpha_{BT}$. Thus, BitTorrent has the lowest probability of piece exchange:

*Corollary 2:* Altruism has the highest probability of piece exchange between given users $j$ and $k$. As $N \to \infty$, T-Chain has the same probability of piece exchange as altruism.

FairTorrent suffers less from piece availability since it does not explicitly require reciprocation. Thus, any user $j$ can, piece deficits permitting, upload to user $i$ if $i$ needs a piece from $j$.

**Effect on efficiency:** Figure 3 shows our efficiency predictions in this scenario. Given the exchange probabilities in Corollary 2, we expect altruism to be the most efficient algorithm, followed closely by T-Chain. Indeed, since piece availability is the only restriction on users exchanging pieces in T-Chain, we expect that it will be nearly as efficient as altruism for a large number of users. FairTorrent is similarly unconstrained by piece availability, but FairTorrent users are required to upload to users with the lowest piece deficits; thus, we expect FairTorrent to be less efficient than T-Chain.

Comparing Figures 2 and 3, we see that T-Chain's, FairTorrent's, and BitTorrent's efficiencies have reversed compared to the ideal scenario. Since real systems likely reflect a mix of both scenarios, we expect these three hybrid algorithms to attain comparable efficiencies in practice. Section V's experiments validate this prediction.

**Effect on fairness:** Since FairTorrent and T-Chain are the least constrained by piece availability (Corollary 2) and the most fair in the idealized case (Figure 2), we expect that they will continue to be the most fair algorithms with piece availability constraints. We expect BitTorrent to be the next most fair, with altruism remaining the least fair.

**Reputation algorithm:** The download rate of each user with the reputation algorithm depends on its reputation score. While we would expect that users with higher bandwidth would have higher reputation scores, they may have lower reputation scores due to receiving few pieces in the initial stages. The system fairness and efficiency can then suffer:

*Proposition 3:* Let $r_i$ denote each user $i$'s reputation, with $\sum_{k=1}^{N} r_k \gg r_i$. Once the system reaches an equilibrium with perfect piece availability, its fairness and efficiency are

$$F = \frac{1}{N}\sum_{i=1}^{N}\left|\log\left(\frac{d_i}{u_i}\right)\right| = \sum_{i=1}^{N}\left|\log\left(\frac{r_i \sum_{k=1}^{N} U_k}{NU_i \sum_{k=1}^{N} r_k}\right)\right|$$

$$E = \sum_{i=1}^{N}\frac{\sum_{k=1}^{N} r_k}{N r_i}. \tag{9}$$

For instance, if one user has very low reputation but moderate upload bandwidth, its efficiency and fairness will both suffer. Thus, in contrast to our prediction in Figure 1 and analysis for the idealized scenario, reputation algorithms can in practice achieve poor fairness and efficiency. We empirically demonstrate this result for realistic scenarios in Section V.

## B. Bootstrapping Speed

Since more efficient algorithms tend to bootstrap users faster, we expect the algorithms' relative bootstrapping speeds to be similar to their efficiency. We define the **bootstrapping time** $T_B(P)$ as the time for $P$ newcomers who arrive in a flash crowd to each receive at least one file piece, assuming no free-riders in the system.[4] We assume that there is one seeder, and we let $z(t)$ denote the number of bootstrapped users at time $t$. We consider a series of discrete timeslots $t = 1, 2, \ldots$, and let $K$ denote the average number of pieces that each user can upload within a single timeslot.

To compare the algorithms' bootstrapping times, we first note that $T_B(P)$ is a function of the bootstrapping probability:

*Lemma 3:* The expected amount of time until $P$ users are bootstrapped is

$$\mathbb{E}\left[T_B(P)\right] = \sum_{n=1}^{\infty}\left(1 - \left(1 - \prod_{t=1}^{n}\left(1 - p_B(t)\right)\right)^P\right), \quad (10)$$

where $p_B(t)$ denotes the probability that a single newcomer is bootstrapped at time $t$.

Since (10) decreases as each $p_B(t)$ increases, algorithms with a higher probability of bootstrapping yield lower bootstrapping times. Instead of comparing the algorithm bootstrapping times, we thus compare the bootstrapping probabilities. Table II gives this probability for each algorithm, which we derive below.

To derive Table II, we first note that all probabilities have the form $1-(N-n_S)x/N$, where $x$ is the probability that a given user $i$ is *not* bootstrapped by any other user, and $(N-n_S)/N$ is the probability that a user is not bootstrapped by the seeder (we assume the seeder bootstraps $n_S$ users per timeslot). It therefore suffices to find $x$ for each of the six algorithms.

For reciprocity, we have $x = x_R = 1$, since users never send data to each other. For T-Chain, we find that $x = x_{TC}$ is

$$\left(\pi_{DR} + (1-\pi_{DR})\frac{N-2}{N-1}\right)^{Kz(t)} = \left(\frac{N-2+\pi_{DR}}{N-1}\right)^{Kz(t)}, \quad (11)$$

i.e., the probability $\pi_{DR}$ that a user $j$ engages in direct reciprocity with another user $i$, plus the probability that $j$ engages in indirect reciprocity with a user $k \neq i$. We assume that $j$ can always find a user who needs one of its pieces, i.e., that indirect reciprocity can always occur. Note that $\pi_{DR}$ will change as the pieces become more dispersed among users.

[4]While flash crowds are an extreme scenario, they can occur, e.g., if a cloud server acts as a seeder to distribute urgent instructions to a network of clients.

To derive BitTorrent's probability $x = x_{BT}$, we assume that user $j$ uploads to $n_{BT}$ users with reciprocity and altruistically uploads to one other user in each timeslot. Thus, user $j$ does not bootstrap user $i$ with probability $(N - n_{BT} - 2)/(N - n_{BT} - 1)$, yielding

$$x_{BT} = \left(\frac{N - n_{BT} - 2}{N - n_{BT} - 1}\right)^{z(t)},$$

as in Table II.

To derive FairTorrent's bootstrap probabilities, we note that a user $j$ will only bootstrap another user $i$ if all of $j$'s deficits with other users are $\geq 0$. We use $\omega$ to denote the probability that this does not happen (users have a negative deficit with at least one other user) and suppose that the user has a zero deficit with at least $K$ users (e.g., newcomers). Then the probability that a user $i$ will not be bootstrapped is

$$x_{FT} = \left(\omega + (1-\omega)\frac{n_{FT} - K - 1}{n_{FT} - 1}\right)^{z(t)}, \quad (12)$$

where $n_{FT}$ denotes the number of users with zero deficits and we suppose that the user randomly chooses among the users with zero deficits. The probability $\omega$ is related to $\pi_{DR}$, which we used to analyze T-Chain's bootstrapping speed: if users can directly reciprocate to each other, their piece deficits for each other will flip back and forth from 0 to $\pm 1$. However, we expect that $\pi_{DR} \leq \omega$: $\omega$ measures the probability that a user has a negative deficit with *at least one* of the $K$ users with which it exchanges pieces, which is likely to happen if it receives pieces from multiple users.

Users of a reputation algorithm generally will not bootstrap newcomers, as these users have zero reputation. Thus, newcomers can only be bootstrapped by altruism. We suppose that half of the users altruistically upload to one other user in each timeslot, as suggested by [4].

Finally, when users upload only altruistically, each bootstrapped user uploads $K$ pieces to users chosen uniformly at random. Thus, the probability that a given user $i$ is not bootstrapped is

$$x_A = \left(1 - \frac{1}{N-1}\right)^{Kz(t)}. \quad (13)$$

We can compare the bootstrapping probabilities as follows:

*Proposition 4 (Comparing bootstrapping speeds):* Altruism has the largest bootstrapping probability when $K \geq 2$, $N \gg K$, and $\omega$ is large enough so that

$$(1-\omega)\frac{N-1}{n_{FT}-1} \leq \left(1 - \frac{1}{N-1}\right)^{K-1}. \quad (14)$$

T-Chain and FairTorrent bootstrap as fast as altruism when $\pi_{DR} = \omega = 0$. If $N$ and $n_{FT} \gg n_{BT}$ and $\pi_{DR}$ and $\omega$ are sufficiently small, T-Chain and FairTorrent bootstrap faster than BitTorrent, while reputation is slower than BitTorrent.

As $K$ increases, the conditions on $\pi_{DR}$ and $\omega$ for which T-Chain and FairTorrent bootstrap faster than BitTorrent become less strict; for instance, if $K = 2$, it is sufficient for

TABLE II: Bootstrap probabilities for each algorithm when a flash crowd arrives. Sample probabilities assume that $N = 1000$, $n_S = 1$, $K = 5$, $z(t) = 500$, $\pi_{DR} = 0.5$, $n_{BT} = 4$, $\omega = 0.75$, $n_{FT} = 500$.

| Algorithm | Probability of Bootstrapping | Example |
|---|---|---|
| Reciprocity | $\frac{n_S}{N}$ | 0.1% |
| T-Chain | $1 - \frac{N-n_S}{N}\left(\frac{N-2}{N-1} + \frac{\pi_{DR}}{N-1}\right)^{Kz(t)}$ | 71.4% |
| BitTorrent | $1 - \frac{N-n_S}{N}\left(\frac{N-n_{BT}-2}{N-n_{BT}-1}\right)^{z(t)}$ | 39.6% |
| FairTorrent | $1 - \frac{N-n_S}{N}\left(\omega + (1-\omega)\frac{n_{FT}-K-1}{n_{FT}-1}\right)^{z(t)}$ | 71.4% |
| Reputation | $1 - \frac{N-n_S}{N}\left(\frac{N-2}{N-1}\right)^{z(t)/2}$ | 22.2% |
| Altruism | $1 - \frac{N-n_S}{N}\left(\frac{N-2}{N-1}\right)^{Kz(t)}$ | 91.8% |

$\pi_{DR}, \omega \leq 1/2$. Proposition 4 holds for the example scenario in Table II, where we assume that 500 of 1000 users have been bootstrapped; thus, the probability of direct reciprocity in T-Chain, $\pi_{DR}$, is at most 0.5, and even smaller if fewer users are bootstrapped soon after the flash crowd arrives. The probability $\omega$ will be similarly small when users initially arrive, but will grow as users exchange pieces with more users.

Comparing Proposition 4's and Table II's results with our efficiency predictions in Figure 3, we see that more efficient algorithms generally bootstrap faster. Surprisingly, FairTorrent bootstraps just as fast as T-Chain, though it is slightly less efficient due to users' requirement to upload to those with the lowest piece deficits. When a flash crowd arrives, most users have similar piece deficits, so FairTorrent is relatively unconstrained. Thus, combining our results here with those for fairness and efficiency in the previous section, we would expect T-Chain and FairTorrent to outperform BitTorrent in real systems, though all three achieve comparable efficiency. Section V's results verify these expectations.

## C. Susceptibility to Free-Riding

We finally examine the algorithms' susceptibility to free-riding. Though the altruism hybrids are less susceptible than altruism alone, which makes all upload bandwidth available to free-riders, they still inherit some susceptibility. However, while we would expect more fair algorithms to be less susceptible to free-riding, we show that this is not always the case. We show in Section V that free-riding affects both the fairness and efficiency of the different incentive mechanisms, compared to the results of [10], which do not consider free-riding.

We can roughly quantify the potential for free-riding through two metrics: the amount of available "free" resources and the potential for acquiring resources through collusion or Sybil attacks. Attacks such as exploiting altruism [6], cheating [33], the large-view-exploit [18], [19], and whitewashing [4], [34] all take advantage of freely available resources, while collusion enables users to obtain resources by falsely convincing other users that they are behaving legitimately [8], [16].

TABLE III: Resources available for free-riding.

| Algorithm | Exploitable resources | Collusion probability |
|---|---|---|
| Reciprocity | 0 | none |
| T-Chain | 0 | $\pi_{IR}\frac{(m-1)m}{(N-1)N} \ll 1$ |
| BitTorrent | $\alpha_{BT}\sum_{i=1}^{N} U_i$ | none |
| FairTorrent | $(1-\omega)\sum_{i=1}^{N} U_i$ | none |
| Reputation | $\alpha_R\sum_{i=1}^{N} U_i$ | 1 |
| Altruism | $\sum_{i=1}^{N} U_i$ | n/a |

**Non-collusive free-riding:** Table III gives the amount of resources directly exploitable by free-riders in the absence of collusive attacks. We use the same notation as in Section IV-A's fairness-efficiency analysis; thus, the total system resources are $\sum_{i=1}^{N} U_i$, or the total upload bandwidth. Reciprocity and T-Chain are less susceptible than the other algorithms; this conclusion is consistent with Section III-B's observation that reciprocity algorithms more strictly enforce fairness, making it difficult to free-ride. BitTorrent also has elements of reciprocity, but its use of altruism makes $\alpha_{BT}$ of its upload resources available for free-riders. The reputation algorithm similarly makes $\alpha_R$ of its upload capacity available.

Free-riders in FairTorrent can receive pieces from a user $i$ only when user $i$ has nonnegative piece deficits from all other users. Thus, the amount of exploitable resources is $(1 - \omega)\sum_i U_i$, where $\omega$, as in Section IV-B, is the probability that a user has a negative deficit with at least one other user. In the most favorable scenario when $\omega = 0$, $m$ free-riders could obtain $m/N$ expected pieces per timeslot. From [7], users have a deficit of at most $O(\log N)$ pieces over time, which bounds the total number of pieces a free-rider can receive.[5] This bound applies even if users employ a whitewashing attack [4], [34], i.e., continually creating new user IDs so that they do not accumulate positive piece deficits.

**Free-rider collusion:** Collusive attacks are more effective than individual free-riding when user transactions involve third parties. For instance, T-Chain is vulnerable to some collusion when indirect reciprocity occurs: if a user $\mathbb{S}$ uploads to a free-rider $\mathbb{R}$ and asks $\mathbb{R}$ to reciprocate to a colluding free-rider $\mathbb{P}$, then $\mathbb{P}$ may falsely report receipt of a piece from $\mathbb{R}$, thus triggering $\mathbb{S}$ to send the decryption key to $\mathbb{R}$ and allowing $\mathbb{R}$ to receive a piece for free. However, the probability of indirect reciprocity $\pi_{IR}$ is

$$q(i,j)(1-q(j,i))\left(1-\left(1-\sum_{l=1}^{M} p_l q(j,l)\left(1-q(l,j)\right)\right)^{N-2}\right),$$

which is generally quite low (cf. Proposition 2), so collusions rarely occur [8]. The reputation algorithm is very vulnerable to collusion since free-riders can use false praise to artificially raise each others' reputations and receive more uploads.[6]

[5]Usually, $\omega > 0$: most users have a negative deficit with some other user.

[6]More sophisticated reputation schemes that consider users' trustworthiness [4] can circumvent such false praise to some extent; however, if legitimate users collude with many free-riders, then users can still game the system.

Non-reputation and local reputation algorithms like FairTorrent do not suffer from collusion. However, since they are vulnerable to other attacks, we expect T-Chain to perform better than the other hybrid algorithms in the presence of free-riders, as opposed to comparable performance with no free-riders. Our experiments in Section V validate this prediction.

## V. Experimental Results

We evaluate the six information exchange algorithms using an event-driven simulator adapted from the simulator used for TBeT [33]. We compare the algorithms' fairness, efficiency, bootstrapping speed, and susceptibility to free-riding, as in Section IV. We define free-riding susceptibility as the fraction of upload bandwidth received by the free-riders. For convenience, we use the average fairness, $\left(\sum_i u_i/d_i\right)/N$, to measure the system fairness in our experiments, instead of the statistic $F$ defined in (3).

### A. Simulation Setup

We initiate each experimental run with one seeder. One thousand users arrive in a flash crowd within the first 10 seconds, as considered in the previous section's performance analysis, and start downloading a data file of size 128 MB from other users and the seeder. Users exit the swarm immediately after finishing the download.

For the reciprocity algorithm, we assume that users upload only to the neighbor that has contributed the most to them. With altruism, users instead upload to random neighbors at their full upload capacity. With BitTorrent, users upload to random neighbors with a 20% probability, and otherwise to neighbors with the highest contributions. For the reputation algorithm, we assume that all users know the amount of data that each user uploads to all other users; users' reputations are proportional to this amount of data. We run the simulations without and with different types of free-riding attacks, which are chosen to maximize each algorithm's vulnerability. We then add the large-view exploit attack [18], [19] to all algorithms and evaluate its effect on the system performance.

### B. Algorithm Comparisons

*1) No Free-Riding:* Figure 4 compares the algorithm performance when all users are compliant (no free-riding). As predicted in Figures 2 and 3, altruism is the most efficient, i.e., it has the shortest download completion times in Figure 4a. Reciprocity is the least efficient; no user completes the download at all. T-Chain, BitTorrent, and FairTorrent show comparable efficiency, which is in between the results of the idealized (Figure 2) and piece availability (Figure 3) scenarios in Section IV-A: in the first, idealized, scenario, we would expect both T-Chain and FairTorrent to be slower than BitTorrent, but in the second we would expect T-Chain to be the most efficient. Thus, we conjecture that the idealized scenario can model the middle of the simulation, when pieces are well-distributed among users, but that we need to account for piece availability at the beginning and end of the simulation.

The fairness in Figure 4b also matches our predictions in Figure 2: as the system stabilizes, T-Chain's and FairTorrent's fairness values approach 1, with BitTorrent's nearly at 1. Altruism and reputation are much less fair, as predicted by Proposition 3's reputation analysis. The reputation algorithm's fairness drops after 300 seconds due to users with higher upload capacities (and thus, higher reputations) completing their downloads and leaving the system.

The bootstrapping speeds in Figure 4c are consistent with Section IV-B's analysis. Altruism and FairTorrent are the fastest, as expected from Proposition 4, while T-Chain can nearly equal altruism's bootstrapping speed. Reciprocity, reputation, and BitTorrent are respectively the slowest to third-slowest algorithms, as expected from the proposition and the example probabilities given in Table II.

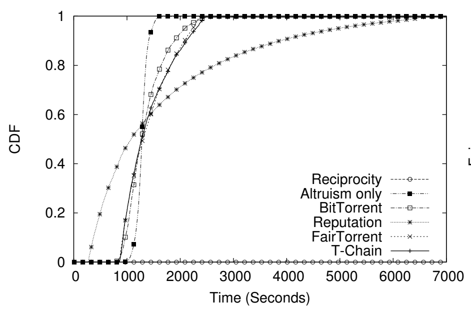
*2) Effect of Free-Riding:* Figure 5 compares the algorithm performance when 20% of the users free-ride.[7] We assume that free-riders use the most effective attack for each algorithm, i.e., simple, non-collusive free-riding for most algorithms, with additional collusion for T-Chain [8], [16] and whitewashing for FairTorrent [4], [34] as discussed in Section IV-C.

Figure 5a shows the susceptibility of each algorithm. Reciprocity's and T-Chain's susceptibility values are almost zero, since they have no exploitable resources, as in Table III. BitTorrent and the reputation algorithm are more susceptible, as predicted in the table, and altruism is the most susceptible. FairTorrent is the second-most susceptible algorithm. As discussed in Section IV-C, FairTorrent is susceptible to free-riders if users have nonnegative piece deficits with their legitimate neighbors; they will then upload to new free-riders with zero deficits. We would expect this scenario to occur at equilibrium (Table I) and right after a flash crowd arrives.

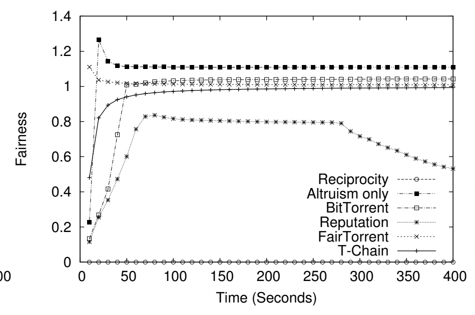
Free-riding also affects the algorithms' efficiency (Figure 5b) and fairness (Figure 5c). Compared to Figure 4a's results without free-riding, most algorithms' efficiency decreases: free-riders now receive some of the upload bandwidth. T-Chain and BitTorrent, which are less efficient than altruism without free-riders, are now more efficient as they are less susceptible to free-riding. They are also the most fair; while FairTorrent achieved comparable fairness without free-riding, its fairness is more affected by free-riding.

We finally add another free-riding attack, the large-view exploit [18], [19], in which free-riders connect to, and therefore receive free resources from, more neighbors than before. The results are shown in Figure 6. All algorithms' susceptibility approximately doubles compared to without the large-view exploit, as shown in Figure 6a. Thus, the algorithms become less efficient and less fair, as can be seen by comparing Figure 6b's efficiency and Figure 6c's fairness results with Figures 5b and 5c. T-Chain is now visibly more efficient and more fair than BitTorrent, as free-riders receive 10% of BitTorrent's upload capacity but less than 1% of T-Chain's.

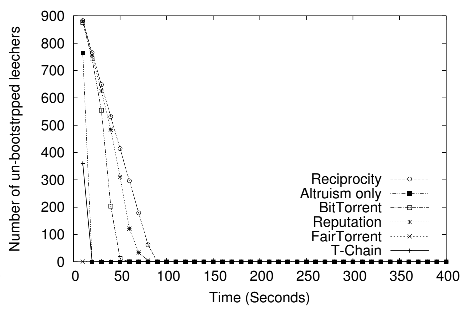
[7] We omit the bootstrapping results due to space, but they are similar to those in the compliant scenario (Figure 4c).



(a) Distribution of download completion times.

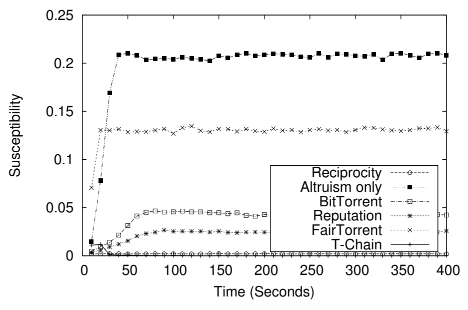


(b) Average fairness.

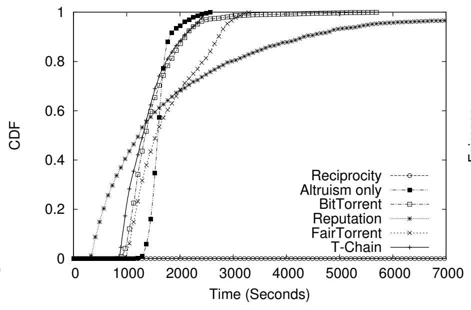


(c) Number of un-bootstrapped users.

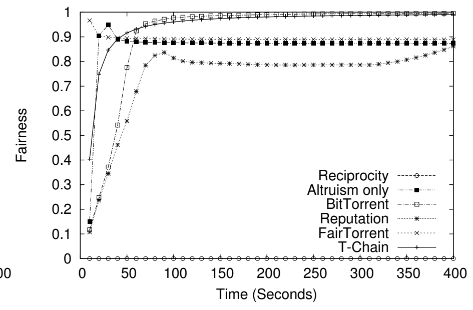
Fig. 4: Performance results when all users are compliant. The results are consistent with our analysis in Section IV.



(a) Susceptibility to free-riding.

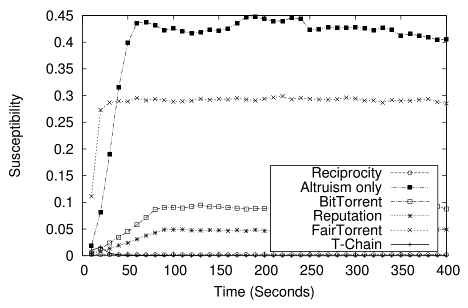


(b) Distribution of download completion times.

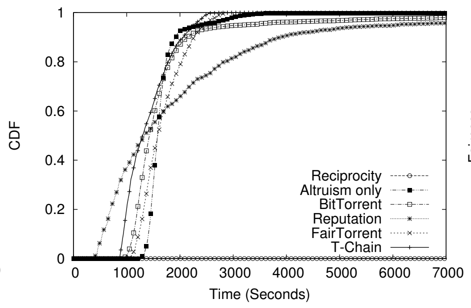


(c) Average fairness.

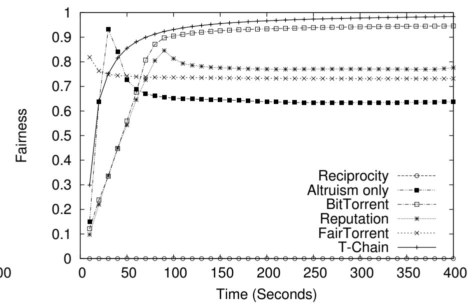
Fig. 5: Performance results for compliant users when 20% of users are free-riders. Free-riding attacks target each algorithm's vulnerabilities, as specified in Section V-B2.



(a) Susceptibility to free-riding.



(b) Distribution of download completion times.



(c) Average fairness.

Fig. 6: Performance results for compliant users when 20% of users are free-riders. We add the large-view exploit to the attacks used for Figure 5.

## VI. Conclusion

In this work, we classify information exchange algorithms in terms of three fundamental classes. We rigorously compare the three basic and three hybrid algorithms, modeling their efficiency, fairness, bootstrapping speed, and susceptibility to free-riding. We find that the three basic algorithms lie at different points along a fairness-efficiency tradeoff, with altruism the least fair and reciprocity the least efficient. However, T-Chain, a hybrid of the reciprocity and reputation algorithms, can exceed the fairness and efficiency of both the reputation and reciprocity algorithms, while nearly matching reciprocity's zero-tolerance for free-riding and altruism's bootstrapping speed. The other hybrids are also fair and efficient, but more susceptible to free-riding.

We validate our results with extensive experiments on an event-driven simulator. The experimental results match with our model's predictions for the algorithms' relative fairness, efficiency, bootstrapping speed, and susceptibility to free-riding. When free-riders are present and target each algorithm's vulnerabilities, the fairness and efficiency of all but the least susceptible algorithms suffer. Since T-Chain is the least susceptible algorithm and exhibits good efficiency and fairness even without free-riders, it performs especially well when free-riders are present. Thus, our work formally classifies and then explains the differences in performance observed for different incentive mechanisms. We intend it to guide practitioners in using these incentive mechanisms to achieve different performance tradeoffs in real systems.

## References

[1] J. Manyika, M. Chui, P. Bisson, J. Woetzel, R. Dobbs, J. Bughin, and D. Aharon, "Unlocking the potential of the Internet of Things," McKinsey Global Institute, 2015, http://www.mckinsey.com/insights/business_technology/the_internet_of_things_the_value_of_digitizing_the_physical_world.

[2] X. Yang and G. De Veciana, "Performance of peer-to-peer networks: Service capacity and role of resource sharing policies," *Performance Evaluation*, vol. 63, no. 3, pp. 175–194, 2006.

[3] B. Cohen, "Incentives build robustness in bittorrent," in *Workshop on Economics of Peer-to-Peer systems*, vol. 6, 2003, pp. 68–72.

[4] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina, "The eigentrust algorithm for reputation management in p2p networks," in *Proc. of WWW*. ACM, 2003, pp. 640–651.

[5] D. Levin, K. LaCurts, N. Spring, and B. Bhattacharjee, "Bittorrent is an auction: analyzing and improving bittorrent's incentives," in *ACM SIGCOMM Com. Comm. Rev.*, vol. 38, no. 4. ACM, 2008, pp. 243–254.

[6] M. Piatek, T. Isdal, T. Anderson, A. Krishnamurthy, and A. Venkataramani, "Do incentives build robustness in bittorrent," in *Proc. of USENIX NSDI*, vol. 7, 2007.

[7] A. Sherman, J. Nieh, and C. Stein, "Fairtorrent: bringing fairness to peer-to-peer systems," in *Proc. of ACM CoNEXT*. ACM, 2009, pp. 133–144.

[8] K. Shin, C. Joe-Wong, S. Ha, Y. Yi, I. Rhee, and D. Reeves, "T-chain: A general incentive scheme for cooperative computing," in *Proc. of IEEE ICDCS*, 2015.

[9] C. T. Ee and R. Bajcsy, "Congestion control and fairness for many-to-one routing in sensor networks," in *Proc. of ACM SenSys*. ACM, 2004, pp. 148–161.

[10] B. Fan, J. Lui, and D.-M. Chiu, "The design trade-offs of bittorrent-like file sharing protocols," *IEEE/ACM Transactions on Networking*, vol. 17, no. 2, pp. 365–376, 2009.

[11] Y. Yue, C. Lin, and Z. Tan, "Analyzing the performance and fairness of bittorrent-like networks using a general fluid model," *Computer Communications*, vol. 29, no. 18, pp. 3946–3956, 2006.

[12] Y. Li, D. Gruenbacher, and C. Scoglio, "Evaluating stranger policies in p2p file-sharing systems with reciprocity mechanisms," *Computer Networks*, vol. 56, no. 4, pp. 1470–1485, 2012.

[13] J. Pouwelse, P. Garbacki, D. Epema, and H. Sips, "The bittorrent p2p file-sharing system: Measurements and analysis," in *Peer-to-Peer Systems IV*. Springer, 2005, pp. 205–216.

[14] R. L. Xia and J. K. Muppala, "A survey of bittorrent performance," *IEEE Communications Surveys & Tutorials*, vol. 12, no. 2, pp. 140–158, 2010.

[15] L. Guo, S. Chen, Z. Xiao, E. Tan, X. Ding, and X. Zhang, "A performance study of bittorrent-like peer-to-peer systems," *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 1, pp. 155–169, 2007.

[16] M. Feldman and J. Chuang, "Overcoming free-riding behavior in peer-to-peer systems," *ACM SIGecom Exchanges*, vol. 5, no. 4, pp. 41–50, 2005.

[17] M. Li, J. Yu, and J. Wu, "Free-riding on bittorrent-like peer-to-peer file sharing systems: Modeling analysis and improvement," *IEEE Trans. on Parallel and Distributed Systems*, vol. 19, no. 7, pp. 954–966, 2008.

[18] T. Locher, P. Moor, S. Schmid, and R. Wattenhofer, "Free riding in bittorrent is cheap," in *Proc. of ACM HotNets*. ACM, 2006, pp. 85–90.

[19] M. Sirivianos, J. H. Park, R. Chen, and X. Yang, "Free-riding in bittorrent networks with the large view exploit." in *IPTPS*, 2007.

[20] P. Jacquet, B. Mans, and G. Rodolakis, "Information propagation speed in mobile and delay tolerant networks," *IEEE Transactions on Information Theory*, vol. 56, no. 10, pp. 5001–5015, 2010.

[21] M. Cha, A. Mislove, and K. P. Gummadi, "A measurement-driven analysis of information propagation in the Flickr social network," in *Proc. of WWW*. ACM, 2009, pp. 721–730.

[22] E. B. Sudderth, A. T. Ihler, M. Isard, W. T. Freeman, and A. S. Willsky, "Nonparametric belief propagation," *Communications of the ACM*, vol. 53, no. 10, pp. 95–103, 2010.

[23] J. Liu, H. Nishiyama, and N. Kato, "A framework for information propagation in mobile sensor networks," in *Proc. of IEEE MASS*. IEEE, 2013, pp. 214–221.

[24] Y. Xu and W. Wang, "The speed of information propagation in large wireless networks," in *Proc. of IEEE INFOCOM*. IEEE, 2008.

[25] W. Youming, "A taxonomy-based perspective of the design trade-offs for BitTorrent-like protocols," Master's thesis, National University of Singapore, 2013.

[26] Y. Tang, H. Wang, and W. Dou, "Trust based incentive in p2p network," in *Proc. of the IEEE Intl. Conf. on E-Commerce Tech. for Dynamic E-Business*. IEEE, 2004, pp. 302–305.

[27] D. Qiu and R. Srikant, "Modeling and performance analysis of bittorrent-like peer-to-peer networks," in *Proc. of ACM SIGCOMM*, vol. 34, no. 4. ACM, 2004, pp. 367–378.

[28] W.-C. Liao, F. Papadopoulos, K. Psounis, and C. Psomas, "Modeling bittorrent-like systems with many classes of users," *ACM Transactions on Modeling and Computer Simulation*, vol. 23, no. 2, p. 13, 2013.

[29] J. J.-D. Mol, J. A. Pouwelse, M. Meulpolder, D. H. Epema, and H. J. Sips, "Give-to-get: free-riding resilient video-on-demand in p2p systems," in *Electronic Imaging 2008*. International Society for Optics and Photonics, 2008, pp. 681 804–681 804.

[30] M. Sirivianos, J. H. Park, X. Yang, and S. Jarecki, "Dandelion: Cooperative content distribution with robust incentives." in *USENIX Annual Technical Conference*, vol. 7, 2007.

[31] M. Piatek, T. Isdal, A. Krishnamurthy, and T. E. Anderson, "One hop reputations for peer to peer file sharing workloads." in *NSDI*, vol. 8, no. 1, 2008, pp. 1–14.

[32] R. Landa, D. Griffin, R. G. Clegg, E. Mykoniati, and M. Rio, "A sybilproof indirect reciprocity mechanism for peer-to-peer networks," in *Proc. of IEEE INFOCOM*. IEEE, 2009, pp. 343–351.

[33] K. Shin, D. S. Reeves, and I. Rhee, "Treat-before-trick: Free-riding prevention for bittorrent-like peer-to-peer networks," in *Proc. of IEEE IPDPS*. IEEE, 2009, pp. 1–12.

[34] J. R. Douceur, "The sybil attack," in *Peer-to-peer Systems*. Springer, 2002, pp. 251–260.

[35] T. Lan, D. Kao, M. Chiang, and A. Sabharwal, "An axiomatic theory of fairness in network resource allocation," in *Proc. of IEEE INFOCOM*, 2010.

## Appendix

### A. Proof of Lemma 1

*Proof:* The download rates $d_i$ should minimize $\sum_i 1/(Nd_i)$k, i.e., the average download time (2), or equivalently $\sum_i 1/d_i$, subject to the (linear) feasibility constraint (1). Since this is a convex optimization problem, we use $\lambda$ to denote the Lagrange multiplier for the feasibility constraint and derive the KKT optimality conditions $\lambda = 1/d_i^2$, i.e., $d_i = \sqrt{\lambda^{-1}}$. We then have $\sum_i \sqrt{\lambda^{-1}} = \sum_i U_i$, i.e.,

$$\lambda = \left(\frac{N}{\sum_i U_i}\right)^2, \quad d_i = \sum_{i=1}^{N} \frac{U_i}{N}, \tag{15}$$

which no algorithm achieves. ■

### B. Proof of Lemma 2

*Proof:* In the reciprocity algorithm, users can only reciprocate pieces given to them; they cannot initiate piece exchanges. Thus, no uploads take place, since users can never initiate uploads.

T-Chain achieves full utilization as users can opportunistically initiate as many exchanges as possible until their upload capacity is saturated; this "opportunistic seeding" is advantageous for users as their uploads are required to be reciprocated.

BitTorrent, FairTorrent, and reputation systems all require users to use their full upload capacity, even in non-equilibrium scenarios. Similarly, fully altruistic users will always donate their full upload bandwidth to other users. ■

## C. Proof of Proposition 1

*Proof:* With perfect piece availability, T-Chain effectively becomes the same as reciprocity: all users can always reciprocate to each other. Thus, we have $d_i = u_i = U_i$.

The download rates for BitTorrent's tit-for-tat algorithm, used for $1-\alpha$ fraction of the upload rate allocation, are taken from [10]. The remaining $\alpha$ fraction comes from altruism.

We calculate the equilibrium download rates for FairTorrent and the reputation system by defining $u(j,k)$ to be the upload rate from user $j$ to user $k$. Then the download rate of user $j$ is $\sum_k u(k,j)$, where we assume that $u(j,j) = 0$, i.e., no user uploads to itself. FairTorrent requires that all users $k$ contributing to a given user $j$ have the same deficit, which we define as $\alpha_j$. Then we have $u(j,k) = \alpha_j + u_{k,j} = \alpha_j + \alpha_k + u(j,k)$, i.e., $\alpha_j = -\alpha_k$. Since this relation holds for all users $j$ and $k$, we have $\alpha_j = 0$ and $u(j,k) = u(k,j)$. Thus, the download rate $\sum_k u(k,j) = \sum_k u(j,k) = U_j$ for all users $j$.

For our reputation-based system in equilibrium, the number of pieces uploaded by each user is proportional to the user's upload capacity $U_i$; thus, we take the reputation of each user $i$ at user $j$ to be $U_i / \sum_{k\neq j} U_k$. Our reputation system requires that absent altruism,

$$u(j,i) = U_j \left( \frac{U_i}{\sum_{k=1,k\neq j}^N U_k} \right).$$

We then sum over all users $j$ to find that

$$\sum_{j=1,j\neq i}^N u(j,i) = U_i \sum_{j=1,j\neq i}^N \frac{U_j}{\sum_{k=1,k\neq j}^N U_k} \approx U_i,$$

absent altruism. Since users devote the remainder of their bandwidth to altruism their expected download bandwidth from others can be calculated from the fact that each user $j$ uses a fraction $\alpha$ of its bandwidth for altruism.

When all pieces are perfectly available, it is reasonable to assume that altruistic users randomly choose the users to which they upload, with each user having the same chance of being chosen by another user. Thus, the download rate of each user $i$ is simply the average upload rates of all users except for user $i$. ■

## D. Proof of Corollary 1

*Proof:* We can see immediately that T-Chain and FairTorrent all achieve the optimal fairness, since their download and upload rates are equal. To see that no algorithm achieves the optimal efficiency, we use the download rates found from Lemma 1,

$$d_i^\star = \sum_{i=1}^N \frac{U_i}{N} \tag{16}$$

for each user $i$, which no algorithm achieves.

To show that altruism achieves the highest efficiency, we first note that the aggregate download rate $\sum_i d_i = \sum_i U_i$ is the same for all algorithms. Thus, since the download time (2) is simply $\alpha$-fairness with $\alpha = 2$, it suffices to show that the download rates from all other algorithms can be transformed into those for altruism by a series of Robin-Hood operations [35]. It is therefore sufficient to show that the largest download rate over all users with altruism is smaller than the largest download rate for any other algorithm over all users. We show the result for the algorithms with $U_i = D_i$; the proof for BitTorrent is analogous. The proof for reputation systems is similar, since under the assumption $\sum_{j=1,j\neq k}^N U_j \approx \sum_{j=1,j\neq k}^N U_j$, users' download rates are equal to their upload rates.

We must show that

$$U_1 \geq \frac{\sum_{k=1}^{N-1} U_k}{N-1},$$

i.e., that $NU_1 + U_N \geq \sum_k U_k + U_1$. Subtracting $U_n$ from both sides, we find that $NU_1 \geq U_1 + \sum_{k=1}^{N-1} U_k$, which holds since $U_1 \geq U_k$ for all $k > 1$.

To show that BitTorrent is more efficient than T-Chain or FairTorrent when $U_i \approx U_{i+n_{BT}}$, we note that this implies that $\sum_{j=\lfloor \mathrm{mod}(i,n_{BT}) \rfloor +1}^{\mathrm{mod}(i,n_{BT})+n_{BT}} U_j \approx U_i$, and thus that BitTorrent achieves download rates $\approx (1-\alpha_R)U_i + \alpha_R \frac{\sum_{k=1,k\neq i}^N U_k}{N-1}$. Thus, the vector of the download rates for all users is a convex combination of the download rates for altruism and T-Chain/FairTorrent. Since the average download time is a concave function of the download rates, we therefore find that BitTorrent's efficiency is between that of altruism and T-Chain/FairTorrent. The proof for the reputation algorithm is analogous. ■

## E. Proof of Proposition 2

*Proof:* For any algorithm, user $j$ can only upload to user $k$ if user $k$ requires a piece from user $j$, which occurs with probability $q(k,j)$. With T-Chain, two possibilities can occur under which user $j$ can upload to user $k$. First, direct reciprocity can occur between the two users, which only occurs if user $j$ requires a piece from user $k$, i.e., with probability $q(j,k)$. Second, if user $j$ does not require a piece from user $k$, indirect reciprocity can occur. For indirect reciprocity to occur, we must have a user $l$ for which user $l$ requires a piece from user $j$ but user $j$ does not require one from user $l$; user $l$ thus redirects user $j$ to reciprocate to user $k$. The probability that at least one user $l$ exists satisfying these conditions is exactly the second summand in (6).

User $j$ will only upload to user $k$ through tit-for-tat with BitTorrent if user $j$ also requires at least one piece from user $k$, which occurs with probability $q(j,k)$; however, any user $k$ that requires a piece from user $j$ can receive one with altruism. Thus, interpreting $\alpha_{BT}$ as the probability of altruism, we obtain (7). The condition on $\alpha_{BT}$ for which $\pi_{TC} \geq \pi_{BT}$ follows directly from (6–7). ■

## F. Proof of Corollary 2

*Proof:* From the proof of Prop. 2, altruism has a higher probability $\pi_A(j,k) = q(k,j)$ of piece exchange than does

BitTorrent and PropShare. It thus suffices to show that $q(k,j) \geq \pi_{TC}(j,k)$, which follows from the fact that

$$0 \leq 1 - \left(1 - \sum_{l=1}^{M} p_l q(j,l)\left(1 - q(l,j)\right)\right)^{N-2} \leq 1$$

in (6). As $N \to \infty$, we see that $\pi_{TC}(j,i) \to \pi_A(j,i)$ as desired. ■

## G. *Proof of Proposition 3*

*Proof:* We suppose that all users request pieces from all other users; thus, we have $u(i,j) = U_i r_j / \sum_{k=1}^{N} r_k$, and for each user $j$,

$$\frac{d_j}{U_j} = \frac{r_j \sum_{k=1}^{N} U_k}{U_j \sum_{k=1}^{N} r_k};$$

adding across all users gives the desired result. To find the efficiency, we have

$$\frac{1}{d_i} = \frac{\sum_{k=1}^{N} r_k}{r_j \sum_{k=1}^{N} U_k}.$$

■

## H. *Proof of Lemma 3*

*Proof:* We first note that the probability that a single user will be bootstrapped within $n$ periods is $1 - \prod_{t=1}^{n}\left(1 - p_B(t)\right)$. Moreover, each user's bootstrapping occurs independently, i.e., the probability of being bootstrapped at a given time does not depend on whether other users are bootstrapped at that time. We see that this is true by noting that both indirect reciprocity and altruism occur randomly: *any* user can be chosen to receive pieces, possibly multiple pieces from the same user in the same timeslot, and bootstrapping occurs if the chosen user happens to not have any pieces. Thus,

$$\mathbb{P}\left[T_B(P) \leq n\right] = \sum_{n=1}^{\infty}\left(1 - \prod_{t=1}^{n}\left(1 - p_B(t)\right)\right)^{P},$$

and (10) follows by using the fact that $\mathbb{E}\left[T_B(P)\right] = \sum_{n=1}^{\infty} \mathbb{P}\left[T_B(P) \geq n\right]$. ■

## I. *Proof of Proposition 4*

*Proof:* We first show that altruism has a faster bootstrapping speed than T-Chain by writing

$$\pi_{DR} + (1 - \pi_{DR})\frac{N-2}{N-1} = \frac{N-2}{N-1} + \frac{\pi_{DR}}{N-1} \geq \frac{N-2}{N-1},$$

so

$$1 - \frac{N-1}{N}\left(\pi_{DR} + (1-\pi_{DR})\frac{N-2}{N-1}\right)^{Kz(t)} \leq 1 - \frac{N-1}{N}\left(\frac{N-2}{N-1}\right)^{Kz(t)}.$$

Substituting $\pi_{DR} = 0$ into T-Chain's bootstrapping probability immediately yields the same bootstrap probability as altruism.

We can similarly show the result for FairTorrent, with $\omega$ replacing $\pi_{DR}$.

When $\omega = 1$ for FairTorrent and $\pi_{DR} = 1$ for T-Chain, both FairTorrent and T-Chain have a $1/N$ bootstrapping probability; they then have the slowest bootstrapping speeds.

To show that altruism is faster than BItTorrent, it suffices to show that

$$\left(\frac{N-2}{N-1}\right)^{K} \leq \frac{N - n_{BT} - 2}{N - n_{BT} - 1},$$

which is equivalent to $(N - k_{PS} - 1)\left((N-1)^K - (N-2)^K\right) \geq (N-1)^K$. Dividing both sids by $(N-1)^K$, we see that it suffices to prove this relation for $K = 2$. We then find the equivalent condition $(2N-3)(N - n_{BT} - 1) \geq (N-1)^2$, which holds for $N >> K \geq n_{BT}$.

To show that altruism is faster than FairTorrent, it suffices to show that

$$\frac{n_{FT} - K - 1}{n_{FT} - 1} + \frac{\omega K}{n_{FT} - 1} \geq \left(\frac{N-2}{N-1}\right)^{K}.$$

We thus simplify to find the equivalent condition

$$1 - \frac{K}{n_{FT} - 1} \geq \left(1 - \frac{1}{N-1}\right)^{K}.$$

We therefore need to show that $1 - K\beta\alpha \geq (1-\alpha)^K$ for $\alpha \in [0,1]$, where $\beta = (N-1)/(n_{FT}-1)$. Since the two sides of the inequality are equal when $\alpha = 0$, it suffices to show that the slope of the LHS is less negative than that of the RHS, i.e., $K(1-\omega)\beta \leq K(1-\alpha)^{K-1}$, yielding the condition (14).

To show that altruism is faster than the reputation system, it suffices to note that $(N-2)/(N-1) < 1$ and $Kz(t) > z(t)/2$.

To show that BitTorrent is slower than T-Chain, we must show that

$$\left(\frac{N - 2 + \pi_{DR}}{N-1}\right)^{Kz(t)} \leq \left(\frac{N - n_{BT} - 2}{N - n_{BT} - 1}\right)^{z(t)}$$

for $K \geq 2$, so it suffices to show the result for $K = 2$. We find the equivalent condition

$$\begin{aligned}&\left(N^2 + (2\pi_{DR} - 4)N + (\pi_{DR} - 2)^2\right)(N - n_{BT} - 1)\\ &\leq \left(N^2 - 2N + 1\right)(N - n_{BT} - 2),\end{aligned}$$

which after gathering terms becomes

$$\begin{aligned}&(N - n_{BT})\left((2\pi_{DR} - 2)N + (\pi_{DR} - 2)^2 - 1\right)\\ &\leq -N^2 + 2\pi_{DR}N + (\pi_{DR} - 2)^2 - 2,\end{aligned}$$

or

$$\begin{aligned}&(2\pi_{DR} - 1)N^2 + \left(\pi_{DR}^2 - 6\pi_{DR} + 3 - 2\pi_{DR}n_{BT} + 2n_{BT}\right)N\\ &+ n_{BT} + 2 - (n_{BT} + 1)(\pi_{DR} - 2)^2 \leq 0.\end{aligned}$$

This relationship clearly holds if $N$ is sufficiently large and $\pi_{DR} \leq 1/2$. Note that for larger $K$, we can have a larger threshold for $\pi_{DR}$.

To show that BItTorrent is slower than FairTorrent, we must show that

$$\frac{N - n_{BT} - 2}{N - n_{BT} - 1} \geq \frac{n_{FT} - K - 1 + \omega K}{n_{FT} - 1},$$

which holds if $n_{FT} \geq N - n_{BT}$ and $(1-\omega)K \geq 1$, i.e., $\omega \leq 1 - 1/K$.

To show that the reputation system is slower than BitTorrent, we must show that

$$\frac{N-2}{N-1} \geq \left(\frac{N-n_{BT}-2}{N-n_{BT}-1}\right)^2,$$

which can be readily seen upon cross-multiplying and canceling summation terms. ■