Systemic risk and user-level performance in private P2P communities

Jia, AL; Rahman, R; Vinko, T; Pouwelse, JA; Epema, DHJ

DOI
10.1109/TPDS.2012.332

Publication date
2013

Document Version
Accepted author manuscript

Published in
IEEE Transactions on Parallel and Distributed Systems

Systemic risk and user-level performance in private P2P communities

Adele L. Jia, Rameez Rahman, Tamás Vinkó, Johan A. Pouwelse, and Dick H. J. Epema.

Abstract—Many peer-to-peer communities, including private BitTorrent Communities that serve hundreds of thousands of users, utilize credit-based or sharing ratio enforcement schemes to incentivize their members to contribute. In this paper, we analyze the performance of such communities from both the system-level and the user-level perspectives. We show that both credit-based and sharing ratio enforcement policies can lead to system-wide "crunches" or "crashes" where the system seizes completely due to too little or to too much credit, respectively. We explore the conditions that lead to these system pathologies and present a theoretical model that predicts if a community will eventually crunch or crash. We apply this analysis to design an adaptive credit system that automatically adjusts credit policies to maintain sustainability. Given private communities that are sustainable, it has been demonstrated that they are greatly oversupplied in terms of excessively high seeder-to-leecher ratios. We further analyze the user-level performance by studying the effects of oversupply. We show that although achieving an increase in the average downloading speed, the phenomenon of oversupply has three undesired effects: long seeding times, low upload capacity utilizations, and an unfair playing field for late entrants into swarms. To alleviate these problems, we propose four different strategies, which have been inspired by ideas in social sciences and economics. We evaluate these strategies through simulations and demonstrate their positive effects.

1 Introduction

In decentralized collaborative systems, including peer-to-peer (P2P) systems, providing incentives for user to contribute is essential. The well-known P2P file-sharing protocol BitTorrent owes its success to its Tit-For-Tat (TFT) incentive policy, which works reasonably well in fostering cooperation among downloaders (also known as *leechers*). However, TFT does not provide incentives for peers to remain in the system after their downloads are complete in order to *seed* the entire file. Therefore, peers are free to engage in "Hit and Run" behavior, leaving immediately upon completing their downloads.

To provide an incentive for seeding, in recent years there has been a proliferation of so-called *private* BitTorrent communities. These communities employ private *trackers* that maintain centralized accounts and record the download and upload activity of each user. They apply policies to incentivize good overall upload / download behavior. One such well-known policy is *Sharing Ratio Enforcement* (SRE), in which each member is required to keep its *sharing ratio* (the ratio between its total amounts of upload and download) at least equal to a threshold called the *SRE threshold*, which is set by the community administrator. Community members whose sharing ratios drop below the threshold are first warned and then banned from downloading, or even expelled from the community. Another such policy is the *credit-based* policy, which requires each member to maintain a positive *credit* (its total amount of upload minus its total amount of download). In this paper we explore both the system-level dynamics and the user-level performance in communities adopting such policies.

Considering a private community as an economic system, we analyze its system-level dynamics by studying its potential systemic risk. In economics, systemic risk is the risk of a collapse of an entire economic system or market [15]. We find that in private communities, too much credit distributed too evenly leads to a *crash* in which peers hold abundant credit and are not willing to contribute. Hence, the system seizes to zero throughput containing only leechers. Conversely, too little credit distributed over the peers leads to a *crunch* in which peers do not have enough credit to download, leading to a seized system containing only seeders[1].

Even when crashes or crunches do nxot occur, i.e., when the system is *sustainable*, this only ensures that the system is able to function, but not how well it functions. Though many measurement studies [7], [17], [18], [24] have shown that the SRE-based and credit-based policies are very effective in boosting contribution levels in terms of high seeder-to-leecher ratios and the corresponding high downloading speeds, we argue that the abundant supply of bandwidth also has several negative effects such as excessively long seeding times that are often unproductive. To explore this, we analyze the user-level performance in sustainable private communities.

- *A. L. Jia, T. Vinkó, J. A. Pouwelse and D. H. J. Epema are with the Parallel and Distributed Systems Group, Delft University of Technology, the Netherlands.*
  E-mail: adele.lu.jia@gmail.com
- *R. Rahman is with the Department of Computer Science, Iqra University, Karachi, Pakistan.*

1. A real world example of the crash and crunch is the story of the Capitol Hill Baby Sitting Co-op [13], which was a group of parents who agreed to cooperate to babysit. A crunch happened when most people wanted to save up coupons: they looked for an opportunity to babysit but there was little demand. Later when more coupons were issued a crash happened: most people felt they had enough coupons so they didn't want to babysit, leaving the system with huge demand but no supply.

The main contributions of this paper are:

1. We demonstrate using simulations that in private communities credit crashes and crunches can occur, and we identify the conditions that lead to these extreme outcomes (Section 4);

2. We present a theoretical model that predicts whether a system will crash, crunch, or be sustainable over a defined time horizon (Section 5). Based on this model we propose an adaptive credit policy that helps the system to avoid crashes and crunches (Section 6);

3. We show that the users in sustainable private communities, while achieving high system-wide downloading speeds, are forced to seed for excessively long times, during which their upload capacity utilizations are quite low (Section 7). Further, when the popularity of a swarm decreases over time, peers that join the swarm not early enough will have to seed for much longer durations than peers who join (strategically) at the beginning of the swarm (Section 10);

4. We propose and evaluate by means of simulations four new strategies that alleviate these problems while still maintaining a reasonable system-wide downloading speed (Sections 8, 9, and 10).

We use private BitTorrent communities as an example, but our analysis is applicable to any P2P system that adopts contribution enforcement policies, by generalizing the metrics for determining the credit and the sharing ratio from the upload and download amounts in a P2P file sharing system to any metrics representing contribution and consumption. To the best of our knowledge, this paper is the most comprehensive effort aimed at studying the various issues with existing incentive mechanisms employed by private P2P communities.

2 Support from Real World Observations

To support our later analysis, we first present real world observations of two private communities, CHDBits.org [1] and Bitsoup.org [2]. CHDBits and Bitsoup both require the users to maintain sharing ratios larger than the threshold of 0.7. The trackers of CHDBits collect information that is periodically reported by the BitTorrent clients of its users, which is displayed in the form of HTML pages available to only its users. We crawled these trackers in May 2011. For each user in CHDBits, we collected the information on its user profile page including the upload and download amount, the seeding time, and the sharing ratio. For each torrent, we collected the information of its published date, and its numbers of seeders and leechers at the time of snapshot. In total, information on all the 31,547 registered users and 40,040 torrents was obtained. For Bitsoup, we use the traces published in [5] that report the user activity of 84,007 users in 13,741 torrents during a period of two months.

2.1 The existence of over-seeding behavior

In previous work [10] we have shown that users who always seed can, counter-intuitively, lead private communities to poor performance due to a credit crunch, in which a few peers accumulate much of the credit and deprive others of the opportunity of downloading. This implies that the user behavior can significantly influence the system performance. Inspired by this finding, we first demonstrate the user behavior as observed in the real world, based on which we later analyze the system-level and user-level performance of private communities.

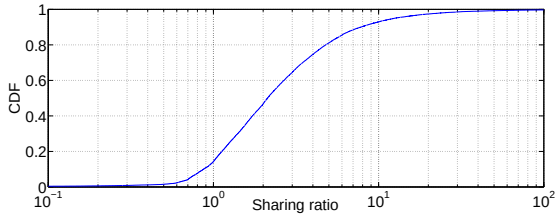
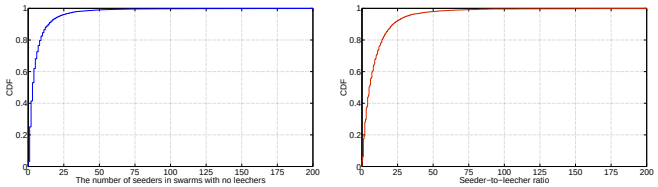


Fig. 1. Over-seeding behavior: the CDF of the sharing ratios of peers in CHDBits.org.



(a) The CDF of the number of seeders in swarms with no leechers

(b) The CDF of the seeder-to-leecher ratio in swarms with at least one leecher

Fig. 2. Oversupply in CHDBits swarms.

In CHDBits, maintaining a sharing ratio slightly above the SRE threshold is sufficient for a user to start downloading a new file. However, we observe that not all the users behave like this. As shown in Fig. 1, more than 95% of the users in CHDBits keep sharing ratios higher than 0.7 and more than 50% of the users keep them higher than 2. This phenomenon of peers seeding more than required and achieving sharing ratios that are (much) higher than the SRE threshold has also been observed in many other communities [17].

From the above observation we abstract two user behaviors for our later analysis, *lazy-seeding* and *over-seeding*. Lazy-seeding peers seed the minimum amount required by the enforcement policies. They represent the users who are download-oriented, i.e., who only seed enough to maintain adequate sharing ratios or credit to be able to start new downloads. On the other hand, over-seeding peers are deposit-oriented, and always maintain sharing ratios (much) higher than required. The behavior of such peers may be triggered by various motivations such as altruism, a desire to be part of the rich elite of the community, or a habit of storing credit for the future. In line with the terminology used in economics, over-seeding peers can be understood as *hoarders* as their behavior essentially amounts to hoarding credit.

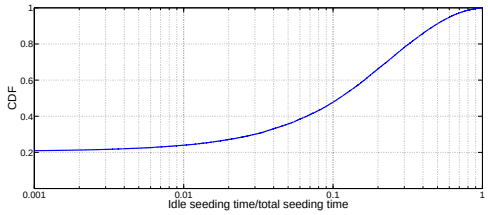


Fig. 3. Unproductive seeding: The CDF of the fraction of idle seeding time of peers with sharing ratios smaller than 1 in BitSoup.org.

2.2 The oversupply

The main motivation for implementing credit or SRE policies is to close the gap between bandwidth demand and supply as observed in public BitTorrent communities, where there is significantly more demand than supply [18]. However, the presence of over-seeding peers completely reverses the situation and in private communities, swarms tend to be extremely oversupplied.

At the time of the crawling, CHDBits had 33,041 active swarms (with at least one leecher or one seeder), among which 26,402 swarms (79.9%) had no leechers at all! As shown in Fig. 2(a), 40% of the swarms with no leechers still had at least 5 seeders, and 5% of these swarms even had more than 20 seeders. For swarms with at least 1 leecher, the *seeder-to-leecher ratio* (SLR) is quite high: as shown in Fig. 2(b), 50% (5%) of these swarms had an SLR of at least 6 (30). We see clearly that a majority of the swarms are heavily oversupplied. In such swarms, intuitively it is difficult for seeders to perform any actual uploads due to the insufficient demand and unsatisfied supply. We validate our speculation through the following observation.

2.3 Unproductive seeding

It is clear that in order to achieve high sharing ratios, peers need to spend considerable amount of seeding time. In the case of over-seeding peers, long seeding times are to be expected. However, we observe that even many peers with small sharing ratios suffer from excessively long seeding times, and a significant part of their seeding time is spent idle without being able to upload anything to others. As a consequence, they have to wait for a long period until their sharing ratios are high enough to start new downloads.

Fig. 3 shows the CDF of the fraction of idle seeding time of peers with sharing ratios smaller than 1 in BitSoup. We see that 10% of these peers spend at least half of their seeding time idle. Note that Fig. 3 only shows the fraction of idle seeding time. It can be conjectured that the fraction of seeding time that is not completely idle yet still yields very low upload speed, would be much higher. We term this situation as *unproductive seeding* and we hypothesize that it is due to the oversupply under credit-based or SRE-based schemes.

Based on these observations, in later sections we analyze the system-level credit dynamics and user-level performance in private communities. Before that, we first introduce the basic model in the following section.

3 Model Description

In this section we will explain the credit-based and SRE-based incentive policies, and our model of communities that employ one of these policies.

3.1 Credit-based versus SRE-based policies

The credit-based and SRE-based policies are essentially very similar, in a way that they can be understood as variations of each other. The idea behind both policies is that every peer has to maintain at all times t a certain relation between the total amount $u(t)$ it has uploaded and the total amount $d(t)$ it has downloaded since it entered the community until time t. The credit-based policy requires users to keep non-negative credit, i.e., to ensure that $u(t) - d(t) \geq 0$, while the SRE-based policy requires users to keep a minimum sharing ratio $SR(t) = u(t)/d(t)$, i.e., to ensure that $SR(t) \geq \alpha$, where α is the SRE threshold. Throughout this paper we assume $\alpha \leq 1$, as most private communities do [2], [3]. When $\alpha = 1$ in the SRE policy, the SRE-based and credit-based policies coincide.

By enforcing non-negative credit in the credit-based policy, the exchanging of data by peers does not generate new credit, and the total amount of credit in the community is always equal to zero (or to the sum of the initial credits allocated to the peers by the community administrator). In contrast, an SRE-based policy allows users to have negative credit (i.e., to have $u(t)-d(t) < 0$, which means that $SR(t) < 1$). Holding negative credit increases the amount of credit among the peers with positive credit in the system—in other words, by holding negative credit a user is essentially minting credit. More precisely, the total credit minted by a user in an SRE-based community with $SR(t) < 1$ until time t is:

$$d(t) - u(t) = (1 - SR(t))d(t), \tag{1}$$

which is bounded by $(1-\alpha)d(t)$. As the sharing ratios of peers fluctuate, SRE-based communities hold a dynamic amount of credit circulating in the system.

3.2 The basic model

We consider a community that is either credit-based or SRE-based. The community comprises a set of s swarms[2] each associated with a file of size F (expressed in number of pieces or units), and a set of N peers

2. We assume the number of swarms to be large enough that even with no injection of new swarms, users still have enough swarms to download from.

each with upload capacity U.[3] We assume no limit on the download capacity of peers. The download model follows the TFT mechanism in BitTorrent, with seeders uploading units to leechers and leechers exchanging units with each other. In reality, a peer can participate in multiple swarms simultaneously, with its bandwidth shared among all the swarms. However, since the sharing ratio is aggregated over all the swarms, we assume that at any time a peer only participates in one swarm, either as a leecher or a seeder.

The operation of the model is based on *cycles* representing units of time. In every cycle, a peer either uploads and/or downloads data or is idle, and at the end of every cycle, it may switch swarms. Peers attempt to download all s files in random order.

In a credit-based community, every peer p is initialized with an amount C_p of credit, and in an SRE-based community, every peer is initialized with a download amount equal to F and a sharing ratio that is a uniformly random number between 0 and 2. A peer can and will only start leeching its next file if its credit or its sharing ratio is at least equal to its *target threshold*, otherwise it continues seeding the current file.

Based on real-world observations, we implement two user behaviors: lazy-seeding and over-seeding (see details in Section 2.1). The target threshold of a lazy-seeding peer is an amount F of credit in a credit-based community (enough to start and complete leeching a new file) and a sharing ratio equal to the SRE threshold in an SRE-based community. The condition for a lazy-seeding peer p at time t to stop seeding is $c_p(t) \geq 0$, with:

$$c_p(t) = \begin{cases} u_p(t) - d_p(t) + C_p - F & \text{credit-based,} \\ u_p(t) - \alpha d_p(t) & \text{SRE-based,} \end{cases} \quad (2)$$

where $u_p(t)$ and $d_p(t)$ represent the total amounts of upload and download of peer p. Over-seeding peers behave in a similar way, but in both credit-based and SRE-based communities they aim at large sharing ratios. Throughout this paper we choose a sharing ratio of 2 as the default target threshold for over-seeding peers. We have run several tests using different values for the threshold and the results show that the tendency of the problem is the same.

4 System-level performance: the crash and crunch

In this section, we perform a number of simulations to explore the credit dynamics in private communities, focusing on analyzing the conditions under which a community will crash, crunch, or be sustainable. We define a *crash* as a situation in which due to credit abundance, peers are not incentivized to contribute and the system completely seizes up, providing no upload or download to any peers. We define a *crunch* as a situation in which due to credit shortages, peers cannot afford new downloads and the system seizes providing no upload or download to any peers. We define a community to be *sustainable* if it does not crash or crunch.

3. Since crash and crunch are due to credit abundance and shortage, respectively, bandwidth heterogeneity does not change whether a system will crash or crunch. However, it does influence the user-level performance, for which we examine both bandwidth homogeneous and heterogeneous systems in Section 7.

TABLE 1
Sustainability of the credit-based system.

frac.of **rich** at start	avg.**throughput** (std.dev)	avg.frac.of **seeders** (std.dev)	final **state**
0.1	0.000 (0.000)	1.000 (0.000)	crunch
0.3	0.218 (0.001)	0.953 (0.005)	sustain
0.5	0.777 (0.002)	0.769 (0.018)	sustain
0.7	0.968 (0.004)	0.506 (0.018)	sustain
0.8	0.587 (0.478)	0.249 (0.204)	sustain/crash
0.9	0.001 (0.000)	0.000 (0.000)	crash

4.1 Experimental setup

We consider a closed system without new peer arrivals. Peer arrivals bring credit into the system and make it difficult to identify whether the underlying credit dynamics is due to the enforcement policy or to the new credit. In fact, in reality many private communities are (nearly) closed [1], [3]. For example, CHDBits hardly has any open registration and new members can only be admitted by extremely restricted invitation.

The simulation is based on the basic model introduced in Section 3, with $N = 1000$, $s = 100$, $F = 10$ units, and $U = 4$ units per cycle. The small file size means the simulation runs produce results at a large scale of granularity. We also performed runs with $F = 100$ and found no significant difference in results. We choose $\alpha = 0.7$ as the default value of the SRE threshold[4], as this value is used in many private communities, e.g., [2], [3]. For each experiment we perform 10 independent runs, and each run is executed for 2000 cycles.

We consider three performance metrics, namely the average throughput, the fraction of seeders, and the state of the system at the end of the simulation. The throughput is expressed as the total amounts of units of data exchanged in the system over an entire run, normalized to the highest one observed in all the experiments. The state of the system indicates whether the system crunches, crashes or sustains.

4.2 Credit-based: constant credit

As discussed in Section 3, the amount of credit in a credit-based community is always equal to the initial credit allocated by the community administrators. In this experiment, we vary the fraction of peers who are

4. We have run several tests using different values for α. Results show that the tendency of the problem stays the same, but with different speeds of entering crash or crunch.

given an initial credit of F (and other peers are given zero credit), which we call rich peers, thus generating different levels of credit in the system.

4.2.1 Populations of lazy-seeding peers

We first show the results of the system containing only lazy-seeding peers in Table 1. When the fraction of rich peers is initialized to 0.3, 0.5, and 0.7, we see sustainable outcomes with increasing throughput and a smaller number of seeders. We have run extended runs up to 20,000 cycles and find that the sustainable outcomes are maintained. This is intuitive since as the amount of credit in the system increases, fewer peers are poor, and hence more exchange of data can occur.

In the crunch state, where only 10% of peers are initialized as rich, the system is composed of all seeders by the end of the run, and hence, no exchange of data can occur. Conversely, in the crash state, where 90% of peers are initialized as rich, all peers are leechers by the end of the run, which again means no exchange of data. Inspection of individual runs shows that crunches and crashes happen quickly—within the first ten cycles or so. This is reflected in the low (almost zero) throughput under crash and crunch states.

It is interesting to see that when the initial fraction of rich peers is set to 0.8, both sustain and crash outcomes can occur. This is reflected in the high variance of the throughput. Here we are very close to the threshold leading to a crash and we find path dependency based on initial random conditions leading to either a high sustainable throughput, or a sudden crash otherwise.

4.2.2 Populations containing over-seeding peers

We find that introducing *any* number of over-seeding peers into the system eventually leads to a crunch, and the speed of the crunch depends on the number of over-seeding peers. This is intuitive since in our experiments, over-seeding peers seed (to hoard credit) until they have a sharing ratio larger than 2. This means that as the simulation progresses, the over-seeding peers eventually hold all the credit in the system and a crunch is inevitable.

4.3 SRE-based: dynamic credit

As discussed above, a credit-based community keeps a delicate constant amount of credit which, if not properly set, will lead the system to crunch or crash. On the other hand, as stated in Section 3, an SRE-based system keeps dynamic credit by allowing peers with sharing ratios less than one to mint some credit. Hence, essentially lazy-seeding peers in an SRE-based community inject credit into circulation, and as in a credit-based community, over-seeding peers absorb credit from circulation.

Intuitively, an SRE-based system cannot be sustainable if all peers are lazy-seeding: soon they will inject too much credit into circulation, which eventually leads the system to a crash. However, as we have shown in Section 2.1, in private communities over-seeding peers always exist and they absorb credit from circulation. Hence, in an SRE-based system with over-seeding peers, the effect of credit-injecting by lazy-seeding peers can be alleviated and the system might eventually be sustainable.

TABLE 2
Sustainability of the SRE-based system.

frac.of **over -seeding peers**	avg.**throughput** (std.dev)	avg.frac.of **seeders** (std.dev)	final **state**
0.1	0.0093 (0.0012)	0.0000 (0.0000)	crash
0.2	0.2046 (0.2103)	0.0037 (0.0082)	crash/sustain
0.3	0.8910 (0.0041)	0.1487 (0.0141)	sustain
0.4	0.9865 (0.0090)	0.4212 (0.0243)	sustain
0.5	0.1436 (0.0083)	1.0000 (0.0000)	crunch

We run several simulations to validate the above hypotheses. We consider an SRE-based system in which we vary the fraction of over-seeding peers to assess their influence on the credit dynamics. Table 2 shows the simulation results. Consistent with our intuition, a certain fraction of over-seeding peers (0.3 and 0.4 in our experimental settings) does lead the SRE-based community to be sustainable. A too small or a too large fraction of over-seeding peers (0.1 and 0.5 in our experimental settings), on the other hand, eventually leads the system to crash or crunch.

5 Predicting Crashes and Crunches

In this section, we will derive (approximate) conditions for predicting whether the system will crunch or crash.

In the model introduced in Section 3, suppose that at time t, a swarm ℓ has $x^\ell(t)$ leechers and $y^\ell(t)$ seeders. Denoting the fraction of the file that a leecher x_i^ℓ still has to download by $p_i^\ell(t)$, x_i^ℓ needs to spend an amount $\alpha(1-p_i^\ell(t))F$ of credit to finish its download. We define $L_i^\ell(t)$ and $R_i^\ell(t)$ as the sets of peers that have fewer or more pieces of the file than peer i, respectively (for a seeder $y^\ell(t)$, $L_i^\ell(t)$ consists of all leechers and $R_i^\ell(t)$ is empty). We assume that peer i only downloads from peers in $R_i^\ell(t)$, and only uploads to peers in $L_i^\ell(t)$ (this is not quite true in BitTorrent, which makes the conditions that we will derive approximations). We further assume that the credit paid by peer i is equally shared by all peers in $R_i^\ell(t)$. Hence, if the situation (in terms of $L_i(t)$ and $R_i(t)$) does not change from time t onward, peer i can earn an amount $Q_i^\ell(t)$ of credit from the peers in $L_i^\ell(t)$, where

$$Q_i^\ell(t) := \sum_{j \in L_i^\ell(t)} \frac{(1-p_j^\ell(t))F}{|R_j^\ell(t)|}. \tag{3}$$

Let $X_\ell(t)$ and $Y_\ell(t)$ respectively represent the sets of leechers and seeders that, assuming that the situation does not change from time t, are able to achieve their target thresholds and start new downloads. Together with Eq. (2), we have:

$$\begin{aligned} X_\ell(t) &:= \big\{x_i^\ell : c_{x_i^\ell}(t) + Q_i^\ell(t) - \alpha(1-p_i^\ell(t))F \geq 0\big\}, \\ Y_\ell(t) &:= \big\{y_j^\ell : c_{y_j^\ell}(t) + Q_j^\ell(t) \geq 0\big\}. \end{aligned} \tag{4}$$

Now we estimate the remaining download time of leechers and the remaining seeding time of seeders for the current file. Here we assume that during the upload process, leechers and seeders alike upload with their full capacity U and distribute their upload capacity equally across all the leechers they are uploading to. The estimated remaining download time $T_i^\ell(t)$ of leecher x_i^ℓ can be expressed as

$$T_{x_i^\ell(t)} := \frac{(1-p_i^\ell(t))F}{\sum_{k\in R_i^\ell(t)} \frac{U}{|L_k^\ell(t)|}}. \quad (5)$$

Similarly, the estimated remaining time for a seeder y_j^ℓ to achieve its target threshold and stop seeding is:

$$T_{y_j^\ell}(t) := \max\{0, -c_{y_j^\ell}(t)\}/U, \quad (6)$$

where $-c_{y_j^\ell}(t)$ (if positive) represents the credit y_j^ℓ still needs to earn to achieve its target threshold.

We can now formulate the *condition for a crunch* to happen in the system as the condition that the sets $X_\ell(t)$ and $Y_\ell(t)$ are both empty for all swarms ℓ at some time t, because that no leechers or seeders are able to earn enough credit to leave their swarms. As a consequence, by the time that the last leecher finishes its download, there will be no exchange of credit in the whole system.

In order to formulate the condition for a crash to happen, let $P_\ell(t) := \{x_i^\ell \ : \ x_i^\ell \notin X_\ell(t)\}$ be the set of leechers in swarm ℓ who will need to seed after finishing their current downloads in order to achieve their target thresholds. Then the *condition for a crash* to happen is

$$|Y_\ell(t)| = y^\ell(t) \text{ and } \min_{k\in P_\ell(t)} T_{x_k^\ell}(t) > \max_{j\in Y_\ell(t)} T_{y_j^\ell}(t), \quad (7)$$

for all swarms ℓ at some time t. To see this, note that the system crashes if there are no seeders. The first part of the condition above says that all the seeders in the system will be able to earn enough credit to leave their swarms. If in addition none of the leechers in $P_\ell(t)$ can finish its download before the last existing seeder leaves the swarm and if this happens to all the swarms (the second part of the condition), then the whole system will end up with no seeders and seize completely, i.e., a credit crash will occur.

6 Adaptive Credit for Sustainability

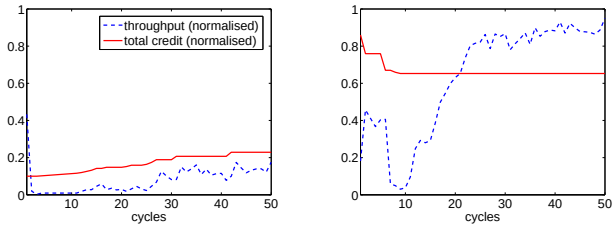
Based on the experimental and theoretical results of Sections 4 and 5, we have designed a novel *adaptive credit intervention mechanism* to avoid crashes and crunches. At each cycle, we check the conditions for crunches and crashes derived in Section 5, thus obtaining early warnings for potential crunches or crashes. When we find that the system is destined for a crunch, a new credit policy called *freeleech*[5] will be applied. As a consequence, leechers do not pay any credit for downloading, but seeders and other uploaders are still credited for uploading. Hence, new credit is injected into the system. Credit injection for stimulating the economy has often been used successfully in real world situations [22]. When we find that the system is destined for a crash, it applies a *freeseed* policy in which seeding peers (and uploading leechers) do not receive any credit for uploading, but leechers still pay credit for downloading. Hence, credit is removed from the system.

We use the credit-based system as an example to evaluate our strategies, but the same analysis can be applied to an SRE-based system. The experimental setup is the same as in Section 4.

5. Freeleech is sometimes also used in existing private communities such as CHDBits, but in a more empirical manner.

TABLE 3
System sustainability with adaptive credit.

frac.of **rich** at start	avg.**throughput** (std.dev)	avg.frac.of **seeders** (std.dev)	final **state**
0.1	0.234 (0.026)	0.948 (0.012)	sustain
0.3	0.311 (0.004)	0.941 (0.005)	sustain
0.5	0.782 (0.002)	0.769 (0.009)	sustain
0.7	0.968 (0.001)	0.512 (0.024)	sustain
0.8	0.976 (0.001)	0.535 (0.015)	sustain
0.9	0.995 (0.002)	0.575 (0.027)	sustain



(a) Initially 10% Rich Peers. (b) Initially 90% Rich Peers.

Fig. 4. The normalized throughput and credit in the system with the adaptive credit intervention mechanism.

6.1 Populations of lazy-seeding peers

Table 3 shows the performance of our adaptive credit intervention mechanism in a credit-based system containing only lazy-seeding peers. All runs produce a sustainable outcome, including those initialized with fractions of 0.1 and 0.9 of rich peers, which previously led to crunches and crashes when the mechanism is not applied (see Section 4). This indicates that the model in Section 5 gives early enough warning for the adaptive credit policy to avoid crashes and crunches.

Fig. 4 shows the results of two runs initialized with fractions of 0.1 and 0.9 of rich peers. A crunch is avoided in Fig. 4.(a) via the activation of freeleech at several cycles—note the increase in credit over time. A crash is avoided in Fig. 4.(b) via the activation of freeseed in the initial cycles—note the decreasing credit over time.

6.2 Populations containing over-seeding peers

As stated in Section 4, any number of over-seeding peers in a credit-based community will eventually lead to a

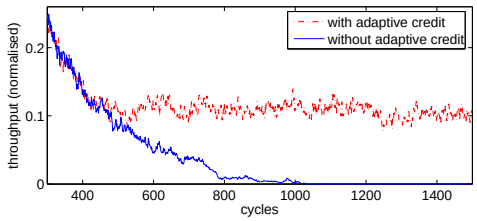


Fig. 5. The normalized throughput with and without the adaptive credit intervention mechanism (50% Rich Peers and 1% over-seeding peers).

crunch, due to the increasing amounts of credit they hoard. In order to test whether our adaptive credit intervention mechanism can deal with this extreme condition, we ran several simulations in which a small subset (1%) of the population are over-seeding peers.

Fig. 5 shows the throughput in the system with and without the adaptive credit intervention mechanism. As can be seen, without the mechanism, the system eventually crunches, whereas with the mechanism, the system is sustainable. However, the throughput of the system in the latter case is still very low. Although new credit is injected each time a crunch is predicted, this additional credit is eventually collected by the over-seeding peers and the process repeats. We believe that this is due to the fact that the adaptive credit intervention mechanism does not attempt to optimize the system, but rather only to avoid a crunch. In later sections we provide a more thorough analysis on optimizing the system, i.e., improving the user-level performance.

6.3 Discussion

The aim of adopting the freeleech and freeseed policies is to avoid crunches and crashes, which is actually achieved, but at the potential cost that the original incentive for contributions is temporarily suspended. It could be argued that this could lead to reduced performance if users learn to game the system by only downloading during freeleech periods and not seeding during freeseed periods.

A refinement that will help preserve incentives even during freeseed and freeleech periods, is to reduce the freeseed and freeleech "tax" amount. Then, rather than having leechers not pay anything at all for downloading and seeders not being credited for uploading, they can be charged or credited for a fraction, say 50%. Any value more than 0% still provides incentives for contribution. Furthermore, the taxation amount can also be variable, and can be applied in a continuous fashion, rather than getting triggered at the extreme conditions of crash and crunch. We explore this later in Section 8.

Until now, we have analyzed the sustainability of a P2P community that adopts a credit-based or SRE-based policy. However, the sustainability of the system only ensures that the system is able to function, but does not guarantee it will function well (recall Fig. 5 for an example of a sustainable system with low performance). To explore this, in the following sections we analyze and improve the user-level performance in sustainable P2P communities. There, we take sharing ratio enforcement as an example, but our analysis is also applicable to the credit-based policy, since it is only a special case of SRE with a threshold equal to one.

7 User-level performance: the positive and negative effects of SRE

In this section we show the user-level performance under SRE. Based on simulations we examine the influence of several parameters and we exhibit the main reasons for the positive and negative effects of SRE.

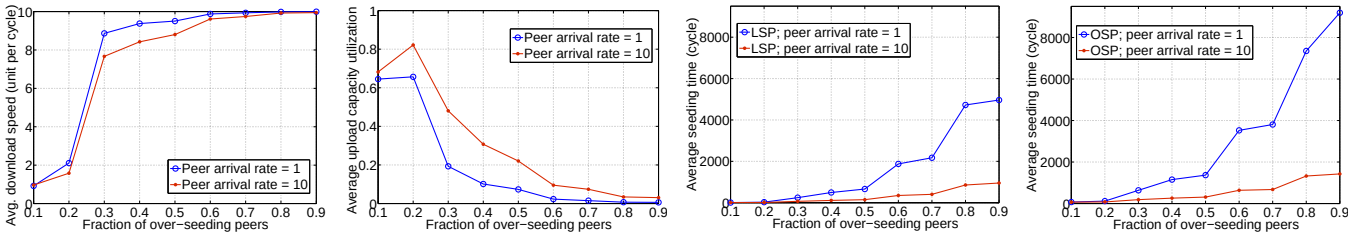
7.1 Experimental setup

In Section 4 we have shown that in closed private communities crashes or crunches easily happen. It is not worthwhile to analyze the user-level performance in an unsustainable system. Hence, in this section we consider an open system with peer arrivals. As stated in Section 3.1 and 4.3, new peers bring credit into the system and the increase of the credit level alleviates the potential systemic risk. In reality, there are many private communities with open registration, e.g., BitSoup [2], and they can be considered as open systems.

We use the same simulator and consider the same initial settings as in Section 3, except that now we consider 100 initial peers and 5 swarms in the system. In each cycle, new peers arrive according to a certain arrival rate (1 or 10 peers per cycle in our simulations) and they join a random swarm to download. After the first download, they maintain a sharing ratio above their target thresholds. Each peer (with upload capacity 1 unit per cycle) attempts to download all the 5 files (with size of 10 units) in the system, in random order. We consider a bandwidth-homogeneous BitTorrent system unless otherwise indicated. We run the simulation for 2000 cycles and keep a record of peers who finish downloading all the files by the end of the simulation. The results represent the average of 5 runs.

7.2 The imbalance of bandwidth supply and demand

In our first experiment we vary the fraction of over-seeding peers, thus generating different levels of over-supply. As shown in Fig. 6(a), with the fraction of over-seeding peers increasing from 0.1 to 0.9, the average downloading speed is increased nearly 10 times. However, the average upload capacity utilization is significantly deteriorated and the seeding time is increased dramatically. With 50% over-seeding peers, on average each peer can only utilize less than 20% of its upload capacity (Fig. 6(b)). With this low upload capacity utilization, all peers have to stay for extremely long times (compared to their downloading times) to achieve the sharing ratio required by SRE (Figs. 6(c) and 6(d)). In our



(a) Downloading speed (b) Upload capacity utilization (c) Seeding time of LSP (d) Seeding time of OSP

Fig. 6. User-level performance under different fractions of lazy-seeding peers (LSP) and over-seeding peers (OSP), and different peer arrival rates.

experiment with 50% over-seeding peers, the seeding time of a lazy-seeding peer is nearly 200 times more than its downloading time, and for over-seeding peers, it even increases to over 400 times.

On the other hand, with a smaller peer arrival rate (which means a smaller demand) the imbalance and hence the performance, are even worse. As shown in Fig. 6, when the peer arrival rate decreases from 10 to 1 peer per cycle, with the same fraction of over-seeding peers, the average upload capacity utilization is decreased 2-3 times and the average seeding time is increased 2-5 times.

Our simulation results show that, under SRE, the existence of over-seeding peers makes the swarms oversupplied. As a consequence, with a relatively large fraction of over-seeding peers and a small peer arrival rate, peers have to seed for extremely long times, though their seedings are not very productive. This is consistent with the theoretical results in our previous work [12].

7.3 The influence of the SRE threshold

Many communities [2], [3] use 0.7 as the default value of the SRE threshold, empirically or intuitively. This section complements the necessary analysis behind the choice.

Fig. 7 shows that, which is consistent with our intuition, when the SRE threshold is increased from 0.2 to 0.9, the upload capacity utilization decreases while the average seeding time is increases. Further, the effect SRE is limited when the fraction of over-seeding peers is small. Surprisingly, in Fig. 7(a) we see that when there are 10% over-seeding peers, the upload capacity utilization is increased when the SRE threshold increases from 0.2 to 0.8, and then drops when it further increases to 0.9. We believe this is due to, what we term as, the *seeder's dilemma*: with either a very small or a very large number of seeders, peers cannot well-utilize their upload capacities. The former case is due to the *piece availability problem*: When there are not enough seeders, leechers have to exchange data with each other, which is not always possible since they only hold a part of the entire file. The latter case is due to the insufficient download demand. Without enough demand, though seeders have the will, they cannot find enough leechers to upload to.

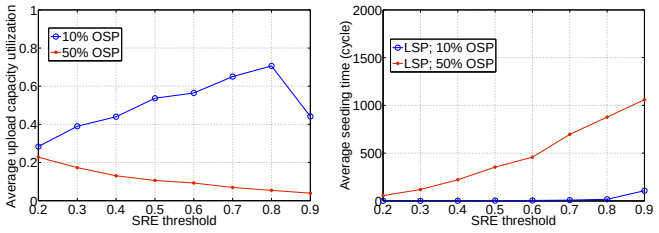
7.4 The discrimination against peers with limited capacities

In this subsection, we analyze SRE's effects in bandwidth-heterogeneous systems. More specifically, we simulate a system with two classes of peers, namely slow and fast peers. All the other settings are the same as in previous experiments, except that the upload capacity of slow peers is 1 unit per cycle and for fast peers it is 4 units per cycle. We consider two scenarios in our simulation, i.e., without and with 30% over-seeding peers. As we show previously, 30% over-seeding peers is typical to demonstrate the effects of SRE. We change the fraction of fast peers from 0.1 to 0.9 and the results are shown in Fig. 8.

We see that when there is no over-seeding peers fast peers barely need to do any seeding work, but their existence increases the seeding times of slow peers (Fig. 8(a)). This result is consistent with our previous work [11] where we show that high-capacity peers manage to upload considerably more during the leeching process, and thus need to seed for shorter times. When the fraction of over-seeding peers is increased from 0 to 30%, slow peers need to seed 200 to 500 cycles more than fast peers, while originally they only needed to seed 20 cycles more. In general, slow peers need to seed 4 times as long as fast peers, which is the same as the ratio between the upload capacity of a fast and a slow peer. This result is also consistent with our previous theoretical results [12].

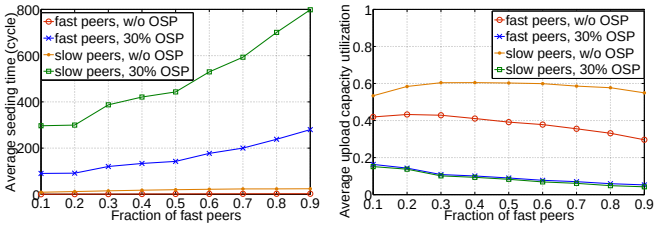
Meanwhile, Fig. 8(b) shows that the upload capacity utilizations of both fast and slow peers do not change much with the fraction of fast peers. However, when there is no over-seeding peer, slow peers have better upload capacity utilizations. We believe this is due to the fact that slow peers stay as seeders longer than fast peers. Normally seeders can achieve better upload capacity utilizations, since they are not influenced by the piece availability problem.

While fast and slow peers both put all their effort in participating in the community, slow peers need to seed longer. We term this as SRE's *discrimination* against low-capacity peers. Clearly, the long seeding time, the low upload capacity utilization, and the discrimination against low-capacity peers severely deteriorate the user-level performance in private communities. In the following sections, we propose several strategies to alleviate



(a) Upload capacity utilization (b) Seeding time of LSP

Fig. 7. Influence of the SRE threshold under different fractions of lazy-seeding peers (LSP) and over-seeding peers (OSP).



(a) Seeding time (b) Upload capacity utilization

Fig. 8. SRE's discrimination under different fractions of over-seeding peers (OSP).

these problems.

8 Description of Proposed Strategies

Inspired by ideas in social sciences and economics, in this section we propose four strategies aimed at alleviating the negative effects of incentive policies used in private BitTorrent communities, which require only a minor revision of those policies.

8.1 Negative taxation

The idea of *negative taxation* is that people earning below a certain amount receive supplemental pay from the government [9]. We take inspiration from the concept of negative taxation and devise a new strategy in which the upload amount of a peer is calculated as its actual upload amount multiplied by coefficient $\mathcal{T}$ defined as:

$$\mathcal{T} = \max\{\min\{1/SR,\ \theta\},\ 1\}, \tag{8}$$

where SR represents the sharing ratio of a peer and $\theta > 1$ represents the *maximum negative taxation degree*.

It is easy to see that a) when $SR \geq 1$, $\mathcal{T} = 1$, b) when $1/\theta \leq SR < 1$, $\mathcal{T} = 1/SR > 1$, and c) when $SR \leq 1/\theta$, $\mathcal{T} = \theta > 1$. By using this new strategy, to gain the same sharing ratio, poor peers ($SR < 1$) seed less and rich peers ($SR \geq 1$) seed the same amount as when using the original SRE. The maximum negative taxation degree controls the maximum negative taxation a peer can get, which alleviates the threat of free-riding.

8.2 Welfare for the rich

The term *welfare for the rich* is used to describe the bestowal of grants and tax-breaks to the wealthy [16]. Taking inspiration from this concept, we devise another strategy to alleviate the long seeding time, i.e., accelerating the seeding process of an over-seeding peer by giving welfare to it. The upload amount of a peer is calculated as its actual upload amount multiplied by coefficient $\mathcal{W}$ defined as:

$$\mathcal{W} = \max\{\min\{SR,\ \varphi\},\ 1\}, \tag{9}$$

where $\varphi > 1$ represents the *maximum welfare degree*.

By using this strategy, to gain the same sharing ratio, poor peers ($SR < 1$) seed the same amount and rich peers ($SR \geq 1$) seed less than when using the original SRE. The maximum welfare degree controls the maximum welfare a peer can get, to prevent the over-seeding seeders from achieving their desired sharing ratios too quickly.

Community administrators can choose different values for θ and φ.In our simulation we choose $\theta = \varphi = 2$.

8.3 Remuneration according to effort

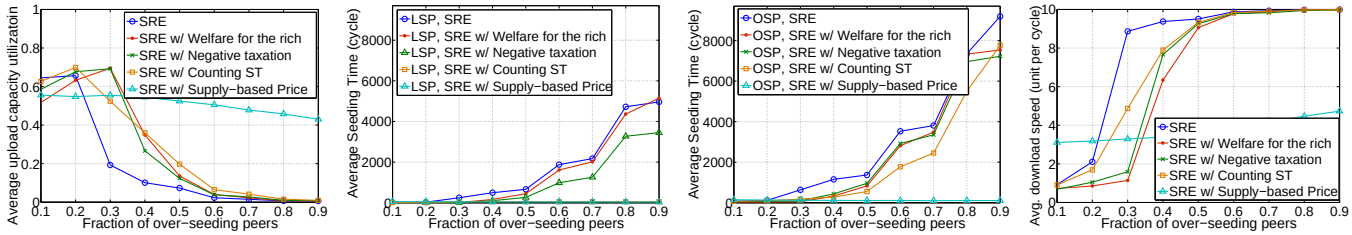
In participatory economics, the *maxim of remuneration according to effort* has been introduced [4]. Under this scheme, people are paid according to the effort they put in rather than the amount of contribution. Taking inspiration from this concept, we propose the third strategy which takes into account the effort of users in terms of their seeding times. Previous studies have shown that the effort-based incentive policy applied in the leeching process improves the system-wide performance [21]. We expect the same improvement when this effort-based methodology is applied in a private community.

More specifically, by applying SRE with *counting seeding time*, a peer can start a new download when either it has achieved the SRE threshold or it has seeded for a sufficiently long time. In this way, peers that are stuck in long seeding process in oversupplied swarms can leave and perform further downloads. The new demand generated by these peers helps to balance the bandwidth demand and supply in the system.

Clearly, the definition of "a sufficiently long period" is quite vague. Community administrators may choose various values, like 4 hours, 10 hours, or one day. In our simulations, we simply assume that it equals the size of the shared file divided by the upload capacity of a peer. Note that since over-seeding peers are deposit-oriented, they still start new downloads only when they have achieved their desired sharing ratios.

8.4 Supply-based price

According to the law of supply and demand, if the demand remains constant and the supply increases, the price of an item decreases and vice versa. For an insightful discussion of the relationship between supply,



(a) Upload capacity utilization (b) Seeding time of LSP (c) Seeding time of OSP (d) Downloading speed

Fig. 9. Strategy performance in alleviating the eternal seeding problem under different fractions of lazy-seeding peers (LSP) and over-seeding peers (OSP).

demand and price, we refer the reader to [6]. We take inspiration from this insight to devise our fourth strategy, i.e., SRE with *supply-based price*. The basic idea is that the price a downloader needs to pay for downloading one unit of data should be inversely correlated with the supply in the swarm, i.e., the higher the seeder-to-leecher ratio, the less a downloader should pay and vice versa. In this way, in an oversupplied swarm, a leecher pays less and potentially achieves a higher sharing ratio by the end of its leeching process. Hence it is less likely for it to have an insufficient sharing ratio and thus stay as a seeder, which indirectly solves the oversupply problem in this swarm. On the other hand, in an undersupplied swarm, a leecher pays more and potentially achieves a smaller sharing ratio, which makes it stay as a seeder with a higher possibility than using the original SRE. In this way, the undersupply problem is also alleviated indirectly.

Here, we use the *seeder-to-leecher ratio* (SLR) as a metric to decide whether a swarm is oversupplied or undersupplied. Community administrators can set different SLR values as the threshold, but we simply assume that when $SLR \geq 1$ the swarm is oversupplied and when $SLR < 1$ the swarm is undersupplied. The download amount of a peer is calculated as its actual download amount multiplied by coefficient $\mathcal{P}$ defined as:

$$\mathcal{P} = \max\{1/SLR,\ \phi\}, \tag{10}$$

where ϕ represents the *lowest price* for downloading one unit of data, which is used to alleviate the threat of free-riders. Community administrators can choose different values for the lowest prices. In our simulation we choose $\phi = 0.1$.

Note that the free-leech strategy we proposed in Section 6 to solve credit crunch is an extreme case of SRE with supply-based price, with price equal to 0.

9 Strategy Evaluation

In this section we evaluate the performance of the new strategies proposed in Section 8. The experimental setup is the same as in Section 7 and results are shown in Figs. 9 and 10.

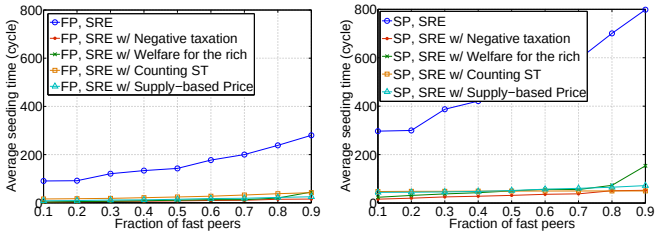
9.1 Higher upload capacity utilization and shorter seeding time

From Fig. 9 we see that by using any of the new strategies, peers achieve higher upload capacity utilizations, as well as smaller seeding times. As shown in Fig. 9(a), when there are 40% over-seeding peers the upload capacity utilization is increased 2-3 times compared to using the original SRE. While all other strategies have decreasing upload capacity utilizations with an increasing fraction of over-seeding peers, SRE with supply-based price performs stably. Given any fraction of over-seeding peers, on average peers can utilize at least 40% of their total upload capacities while for the original SRE it drops to less than 1% when there are 90% over-seeding peers.

With the improved upload capacity utilization, the average seeding time is reduced significantly. As shown in Figs. 9(b) and 9(c), when there are 60% over-seeding peers, SRE with welfare for the rich reduces at least 10% of the original seeding time for both lazy-seeding and over-seeding peers. SRE with negative taxation deals with lazy-seeding peers directly, hence it achieves an even better performance in reducing the seeding time of lazy-seeding peers, which is a 50% improvement compared to that achieved by SRE with welfare for the rich.

SRE with counting seeding time further relieves lazy-seeding peers from the long seeding process in a more effective manner. As shown in Fig. 9(b), they only need to seed for a negligible time compared to when using the original SRE, or either of the above two new strategies. Interestingly, by applying SRE with counting seeding time, the seeding time of over-seeding peers is also decreased (Fig. 9(c)), even though they still desire the high sharing ratios as when using the original SRE. We believe this is due to the fact that with lazy-seeding peers finishing their seedings sooner, the upload competition is reduced and over-seeding peers can achieve their desired threshold more quickly. Meanwhile, when the lazy-seeding peers are released from the seeding process, they join other swarms as new leechers, which indirectly alleviates the oversupply in those swarms.

Finally, the best performance in reducing the seeding time for all peers is achieved by SRE with supply-based price. The seeding time of both lazy-seeding and over-



(a) Seeding time of fast peers (FP) (b) Seeding time of slow peers (SP)

Fig. 10. Strategy performance in alleviating discrimination with 30% over-seeding peers (OSP).

seeding peers is reduced by three orders of magnitude. In our view the main reason for the success of SRE with supply-based price is that it adaptively adjusts the supply and demand in a swarm. When the swarm is oversupplied, the price for downloading one unit of data is lower and peers can finish downloads at less expense, which directly reduces their consequent seeding amount and hence avoids adding more seeders in this oversupplied swarm. In this way, the imbalance of bandwidth supply and demand is mitigated, and the strategy gives a way to escape out of the seeder's dilemma as described in Section 7.3. A similar argument can also be applied to an undersupplied swarm.

9.2 Tradeoff: slightly decreased downloading speed

By adopting any of the new strategies, while the seeding time is dramatically reduced, as a trade-off, the average downloading speed is decreased (Fig. 9(d)), hence the downloading time is increased. However, given that in our simulations we consider files with size equal to 10 units, the increase of the downloading time (tens of cycles) is negligible compared to the decrease of the seeding time (hundreds or even thousands of cycles).

9.3 Reduced discrimination

To examine the effectiveness of the proposed strategies in alleviating SRE's discrimination against peers with limited capacities, we repeat our experiments by further considering a bandwidth-heterogeneous system with two classes of peers, fast and slow. From Fig. 10 we see that *all* the proposed strategies effectively alleviate SRE's discrimination against low-capacity peers. With 30% over-seeding peers, originally slow peers need to seed 200-500 cycles more than fast peers do. By applying any of the new strategies, this difference is reduced to within tens of cycles.

10 Dynamic File Popularity

So far, we have only considered scenarios in which all files have the same constant popularity. However, many measurement studies [14], [5] show that the popularity of a file decreases quickly after it is first published. In this section, we analyze the effects of SRE and evaluate our proposed strategies under dynamic file popularity.

10.1 Experimental setup

We use the same simulator and consider the same initial settings as in Section 3, except that to better abstract the effects of dynamic file popularity, we only consider one swarm with decreasing popularity. The simulation starts with one injector, who stays in the swarm as a permanent seeder. In successive cycles, new peers arrive according to an exponentially decreasing arrival rate ($\lambda(t) = \lambda_0 e^{-\frac{t}{\tau}}$), a peer arrival pattern that has been observed in many BitTorrent swarms [19]. Each peer joins the swarm with zero upload and download amounts. After a peer finishes its download, it seeds, if necessary, until it achieves its target threshold.

By default, we set 30% peers to be over-seeding. As shown in Section 7, this percentage is typical for showing the effects of SRE. We choose $\lambda_0 = 10$ and $\tau = 300$ and we run the simulation for 2000 cycles. All together 3000 peers are included. We have also tested different values for λ_0 and τ, which give very similar results.

10.2 The effects of SRE under dynamic file popularity: arrive sooner to avoid long seeding time

We first demonstrate the effects of SRE under dynamic file popularity. Fig. 11 shows each peer's average download speed, where a smaller peer ID means an earlier arrival time. We see that the first 200 peers experience download speeds similar to their upload capacities (1 unit per cycle). From peer 200, the download speed increases quickly: with file size equal to 10 units, it soon reaches the maximum, i.e., 10 units per cycle. This means that new peers can finish their downloads within the same cycle that they join the swarm.

The high downloading speed is due to the oversupply. As shown in Fig. 12, the number of seeders increases quickly and after the first 30 cycles, the swarm is occupied with hundreds of seeders but only with very few leechers. The presence of existing seeders increases the difficulty for a new seeder to achieve its target threshold and leave the swarm, and vice versa. We term this as *cumulative seeder effect*. As a consequence, the upload capacity utilization decreases severely. As shown in Fig. 13, within the first 500 cycles, both seeder's and leecher's upload capacity utilization decrease to less than 5%, with seeders performing a little bit better than leechers as they do not face the piece availability problem.

With the above differences in instantaneous system performance, peers arriving at different times achieve markedly different performance. As shown in Fig.14, arriving earlier means higher upload speeds and hence larger upload amount during the leeching process (Figs. 14(a) and 14(b)), as well as better upload capacity utilization during the seeding process (Fig. 14(c)). Thus, peers that arrive earlier experience much smaller seeding times. As shown in Fig. 15, to achieve the target thresholds, the first 500 peers only need to seed tens of cycles. After this, the seeding time increases quickly to hundreds or even thousands of cycles.

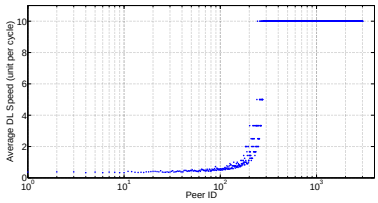


Fig. 11. Individual average downloading speed.

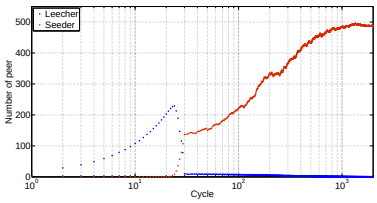


Fig. 12. The number of peers in the system.

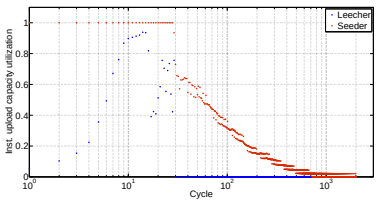
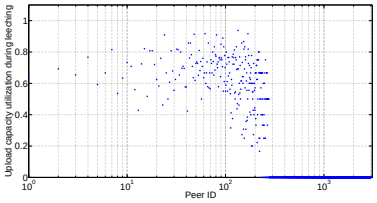
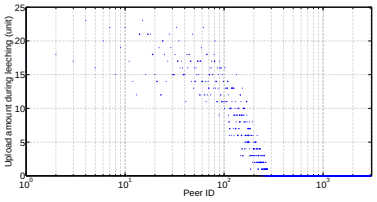


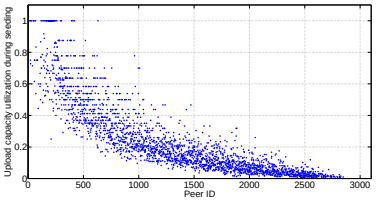
Fig. 13. The average upload capacity utilization.



(a) Individual upload capacity utilization during leeching.



(b) Individual upload amount during leeching.



(c) Individual upload capacity utilization during seeding.

Fig. 14. Individual upload activity.

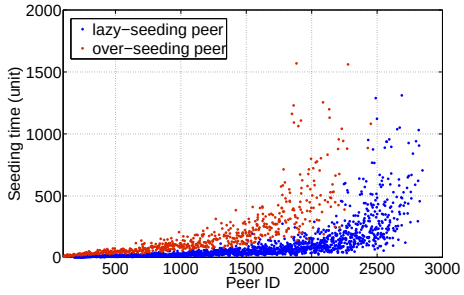


Fig. 15. Individual seeding time.

10.3 Proposed strategies under dynamic file popularity

We evaluate the performance of our proposed strategies under dynamic file popularity. The results are shown in Figs. 16, 17, 18, and 19.

1) SRE with negative taxation and *2) SRE with welfare for the rich: limited effect for very low file popularity:* As in swarms with constant file popularity, applying SRE with negative taxation or SRE with welfare for the rich alleviates the oversupply. Comparing Figs. 12, 16(a), and 17(a), we see that these two new strategies reduce the instantaneous number of seeders to around 80% of the case when the original SRE is adopted. But their effects are limited with very low file popularity. As shown in Figs. 16(b) and 17(b), peers that arrive late still experience relatively long seeding times.

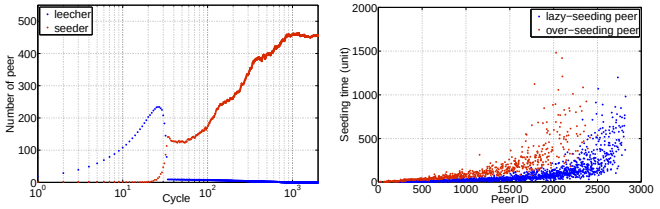
3) SRE with counting seeding time: strong effect in reducing the seeding time: With the hard SRE requirement replaced by seeding for a particular period, SRE with counting seeding time dramatically alleviates the oversupply in the swarm. As shown in Fig. 18(a), except for the large number of seeders during the first 200 cycles, the instantaneous number of seeders is almost always under 100. Among these, we conjecture that most are over-seeding peers, since lazy-seeding peers can leave the swarm once they've seeded for a relatively short time, i.e., 10 cycles in our experiment. While these peers are released from the endless seeding process, the seeding time of over-seeding peers is also dramatically decreased to less than 100 cycles.

4) SRE with supply-based price: effectively stabilize the supply: Among all the four new strategies, SRE with supply-based price stabilizes the supply most effectively. As shown in Fig. 19(a), the instantaneous number of seeders stays stable after 200 cycles. With this constant supply (and not oversupply), peers experience relatively small seeding times (Fig. 19(b)). When the number of new peers decreases, as a matter of course, peers experience longer seeding times, but still much smaller compared to adopting the original SRE. We believe this constant supply and small seeding time are due to the fact that SRE with supply-based price is self-organized, and will adjust the demand and supply automatically.

It should also be noted that when adopting SRE with supply-based price, the first 200 peers have relatively longer seeding times than peers arriving later. We believe this is due to the fact that those peers arrive during the phase that the swarm is occupied with a large number of leechers and a small number of seeders. Hence, the price for downloading is higher and peers need to pay more to achieve their target thresholds.

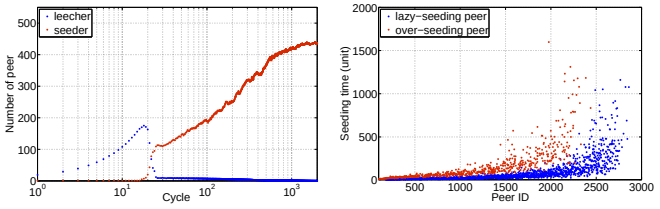
11 Related Work

This paper is based on two previous papers [20], [12] with extensions including demonstrating detailed measurement results, unifying the analysis of SRE-based and credit-based private communities, analyzing the



(a) Instantaneous number of peers (b) Individual seeding time

Fig. 16. SRE with negative taxation under dynamic file popularity.



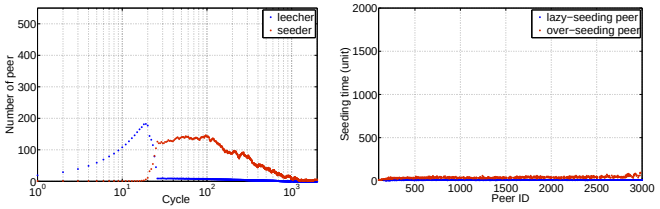
(a) Instantaneous number of peers (b) Individual seeding time

Fig. 17. SRE with welfare for the rich under dynamic file popularity.

systemic risk of SRE-based private communities, as well as the influence of swarm popularity.

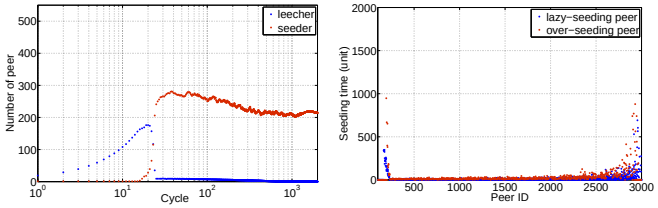
Many P2P incentive schemes based on credits have been proposed in the literature. Vishnumurthy *et al.* [23] present a system involving a virtual currency called *Karma*, which is defined as the value capturing the amount of resources a peer has contributed and consumed. The level of Karma (or credit) in the system is maintained and measures are taken to avoid inflation and deflation that can occur when peers leave the system. However, in avoiding inflation and deflation, the only aim of the paper is to maintain the per-capita Karma, i.e., the total Karma divided by the number of active users.

Kash *et al.* [13] show that in a scrip system, where agents can consume and produce services, both an overabundance of money supply and its shortage lead to inefficiency. They also consider hoarders and how to optimize the credit supply. Our work is different in that we focus not on a generic service exchange scenario but on a file sharing scenario inspired by BitTorrent private communities. Also we apply multiple user behaviors



(a) Instantaneous number of peers (b) Individual seeding time

Fig. 18. SRE with counting seeding time under dynamic file popularity.



(a) Instantaneous number of peers (b) Individual seeding time

Fig. 19. SRE with supply-based price under dynamic file popularity.

rather than focusing only on one that optimizes the utility. In addition we focus on detecting and avoiding extreme crashes and crunches, where the entire system seizes. Last but not the least, we also study the effects of such credit-based schemes from the perspective of user level performance.

As stated, for grounding our work we chose the realm of private P2P file sharing communities. To date, only few works have analyzed private communities. Zhang *et al.* [24] investigate hundreds of private trackers and depict a broad and clear picture of the private community landscape. Chen *et al.* [7] compare system behaviors among 13 private trackers and 2 public trackers, and they show their differences regarding user viscosity, single torrent evolution, user behaviors, and content distribution. Liu *et al.* [17] also perform measurement studies and further develop a model to show that SRE indeed provides effective incentives, but is vulnerable to collusion.

While these studies all focus on demonstrating the high seeding level achieved by private communities, there have been a few preliminary works that show the adverse effects. Andrade *et al.* [5] focus on the dynamics of resource demand and supply, and they show that users typically try to increase their contribution levels by seeding for longer and not by providing more bandwidth to the system. However, our paper shows that providing limited bandwidth is not the will of users, but it is a consequence of the oversupply in private communities. Chen *et al.* [8] also notice the oversupply problem and provide a model to identify the optimal stable SLR range. However, they didn't propose strategies to solve the problem of oversupply. Kash *et al.* [14] demonstrate that there are significant disparities in the cost of new and old files in a private community named DIME, and users compensate for the high cost of older files by downloading more copies of newer files or by preferentially consuming older files during freeleech periods. Particularly, they have shown that after a period of freeleech, there are more download activities in the community. This is consistent with our theoretical result that during a pre-crunch state, injecting credit will increase the system throughput. While these papers mainly perform measurement-based studies to analyze the positive and adverse effects of SRE-like schemes on user-level performance, our paper is based

on measurement, theoretical model, as well as extensive simulations. Further we propose new strategies to alleviate SRE's punishment, which are evaluated to be very effective through simulations.

12 Conclusion

In this paper we have studied the effects of credit-based and SRE-based incentive policies employed in private P2P communities, from both the system-level and the user-level performance perspective.

Based on two user behaviors abstracted from real world observations, i.e., lazy-seeding and over-seeding, we examine the system-level credit dynamics and show that crunches and crashes can easily happen in private communities. Crunches and crashes are due to credit shortage and credit abundance, respectively, and we apply a theoretical analysis to characterize the conditions that lead to these extreme outcomes. We apply the derived conditions to implement a novel *adaptive credit intervention mechanism* that proactively stops the system from seizing by temporarily changing the credit policies. A system that is predicted to crunch allows freeleech, and conversely, a system that is predicted to crash imposes freeseed. Simulation results show that our mechanism is very effective in avoiding crunches and crashes.

Given a private community that is sustainable, we further analyze its user-level performance by analyzing the positive and negative effects of SRE. Our simulation results show that with the existence of over-seeding peers, by adopting SRE, swarms tend to be extremely oversupplied. Although achieving an increase in the average downloading speed, the oversupply induces undesired effects, including low upload capacity utilizations, extremely long seeding times, and an unfair playing field for late entrants into swarms. To alleviate these problems, we propose four strategies and the simulation results show that they are all very effective. Particularly, *SRE with supply-based price*, while maintaining a system-wide high downloading speed, achieves very stable high upload capacity utilization and reduces seeding durations by three orders of magnitude as compared to the original SRE. When then the adaptive intervention mechanism is run in the background to check the extreme conditions for crunches and crashed, the system is ensured to have a high and sustainable performance.

References

[1] http://chdbits.org/.
[2] http://www.bitsoup.org/.
[3] http://hdchina.org/.
[4] M. Albert. Parecon: Life after capitalism. *London*, 2003.
[5] N. Andrade, E. Santos-Neto, F. Brasileiro, and M. Ripeanu. Resource demand and supply in bittorrent content-sharing communities. *Computer Networks*, 53, 2008.
[6] D. Besanko, R. Braeutigam, R.R. Braeutigam, and J. Michael. *Microeconomics*. Wiley, 2010.
[7] X. Chen and X. Chu. Measurements, analysis and modeling of private trackers. In *Proceeding of IEEE P2P*, 2010.
[8] X. Chen, X.W. Chu, and Z. Li. Improving sustainability of private p2p communities. In *Proceeding of IEEE ICCCN 2011*, 2011.
[9] M. Friedman. Capitalism and freedom: Fortieth anniversary edition. *University of Chicago Press*, 2002.
[10] D. Hales, R. Rahman, B. Zhang, M. Meulpolder, and J.A. Pouwelse. BitTorrent or BitCrunch: Evidence of a credit squeeze in BitTorrent? In *Proceeding of Wetice*, 2009.
[11] A.L. Jia, L. D'Acunto, M. Meulpolder, and J.A. Pouwelse. Modeling and analysis of sharing ratio enforcement in private bittorrent networks. In *Proceeding of IEEE ICC*, 2011.
[12] A.L. Jia, R. Rahman, T. Vinko, J.A. Pouwelse, and D.H.J. Epema. Fast download but eternal seeding: the reward and punishment of sharing ratio enforcement. In *Proceeding of IEEE International Conference on Peer-to-Peer Computing (P2P'11)*, 2011.
[13] I.A. Kash, E.J. Friedman, and J.Y. Halpern. Optimizing scrip systems: Efficiency, crashes, hoarders, and altruists. In *Proceeding of ACM Conference on Electronic Commerce*, 2007.
[14] I.A. Kash, J.K. Lai, H. Zhang, and A. Zohar. Economics of bittorrent communities. In *Proceeding of NetEcon 2011*, 2011.
[15] G. Kaufman. Banking and currency crises and systemic risk: A taxonomy and review. *Financial Markets, Institutions and Instruments*, 9, 2000.
[16] D. Lewis. Louder voices: The corporate welfare bums. 1972.
[17] Z. Liu, P. Dhungel, D. Wu, C. Zhang, and K.W. Ross. Understanding and improving incentives in private p2p communities. In *Proceeding of ICDCS*, 2010.
[18] M. Meulpolder, L. D'Acunto, M. Capota, M. Wojciechowski, J.A. Pouwelse, D.H.J. Epema, and H.J. Sips. Public and private bittorrent communities: A measurement study. In *Proceeding of IPTPS*, 2010.
[19] D. Qiu and R. Srikant. Modeling and performance analysis of bittorrent-like peer-to-peer networks. In *Proceeding of SIGCOMM*, 2004.
[20] R. Rahman, D. Hales, T. Vinko, J.A. Pouwelse, and H.J. Sips. No more crash or crunch: Sustainable credit dynamics in a P2P community. In *Proceeding of International Conference on High Performance Computing and Simulation (HPCS'10)*, 2010.
[21] R. Rahman, M. Meulpolder, D. Hales, J.A. Pouwelse, D.H.J. Epema, and H.J. Sips. Improving efficiency and fairness in p2p systems with effort-based incentives. In *Proceeding of IEEE ICC*, 2010.
[22] G. Soros. The worst market crisis in 60 years. *Financial Times*, January 2008.
[23] V. Vishnumurthy, S. Chandrakumar, and E.G. Sirer. Karma: A secure economic framework for peer-to-peer resource sharing. In *Workshop on Economics of Peer-to-Peer Systems*, 2003.
[24] C. Zhang, P. Dhungel, Z. Liu Di Wu, and K.W. Ross. Bittorrent darknets. In *Proceeding of IEEE INFOCOM*, 2010.