Peer-to-Peer Networking and Applications manuscript No.
(will be inserted by the editor)

Mastering Selfishness and Heterogeneity in Mobile P2P Content Distribution Networks with Multiple Source Download in Cellular Networks

Daniel Schlosser · Tobias Hoßfeld

February 3, 2009

Abstract The performance of Peer-to-Peer (P2P) content distribution networks depends highly on the coordination of the peers. This is especially true for cellular networks with mobile and often selfish users, as the resource constraints on accessible bandwidth and battery power are even more limitating in this context. Thus, it is a major challenge to indentify mobile network specific problems and to develop sophisticated cooperation strategies to overcome these difficulties. Cooperation strategies, which are able to cope these problems, are the foundation for efficient mobile file exchange. The detailed performance of the strategies are determined by the peer capabilities and the peer behavior, such as the number of parallel upload connections, the selfishness, or the altruistic re-distribution of data.

The purpose of this work is to evaluate and investigate different cooperation strategies which are based on multiple source download and select the best one for mobile scenarios with even leeching peers, i.e. peers which depart as soon as they have finished their download. The question arises whether the cooperation strategy can smoothen the overall performance degradation caused by a selfish peer behavior. As performance indicators the efficiency, fairness, and robustness of the cooperation strategies are applied. The considered scenarios comprise best-case (altruistic peers) and worst-case scenarios (selfish peers). We further propose a new cooperation strategy to improve the file transfer even when mainly selfish peers are present, the CycPriM (cyclic priority masking) strategy. The strategy allows an efficient P2P based content distribution using ordered chunk delivery with only local information available at a peer.

1 Introduction

Peer-to-Peer (P2P) file sharing systems currently account for the majority of traffic volume transported in the Internet[1]. Whenever large files need to be distributed to many users, applications like BitTorrent make it possible to fulfill this task by merging the resources of many peers and overcome the problems experienced at high loaded servers. Hence efficient P2P file sharing applications create highly scalable and resilient content distribution networks (CDNs). The performance of such P2P CDNs in cellular networks depends highly on the coordination of heterogeneous and often selfish mobile users. Sophisticated cooperation strategies, such as the multiple

University of Würzburg, Institute of Computer Science, Department of Distributed Systems, Würzburg, Germany. {schlosser,hossfeld}@informatik.uni-wuerzburg.de

source download (MSD), are the foundation of the extreme efficiency of P2P content distribution networks. MSD means the simultaneous download of parts of a file, referred to as chunk, from several sources in parallel. The cooperation strategies applied in popular P2P CDN platforms such as eDonkey or BitTorrent, rely on the fundamental P2P assumption that all peers are equal in their functionalities as well as in their capabilities [2]. In cellular networks, however, the peers differ significantly in their characteristics, e.g. radio access system or available bandwidth which might change over time, or their on-line behavior, thus introducing heterogeneity and even selfishness in the peer community. Although all users are equal in their functionality, the P2P assumption of equal peers is not valid any more due to various peer capabilities.

The peer behavior itself is mainly described by two characteristics, (a) churn, i.e. the switching of a user between offline and online state, and (b) the willingness of a user to participate in the CDN. A user may behave selfish and try to minimize the upload of data or he may redistribute the data in an altruistic way. In the context of cellular mobile networks, churn and selfish behavior appear even more distinctive, e.g. to save battery resources or scarce and expensive uplink capacities. Another severe problem in resource restricted P2P CDNs is free riding. This term describes the behavior of a user that modifies the P2P application in order to avoid uploading any data. A P2P CDN with free riding peers loses all the advantages of MSD and degenerates into a client-server system. Thus free riding is never considered as an option in the design of any cooperation strategy. We will therefore focus on systems without free riders. However, a selfish user might shut down its application after he completed the download in order to minimize the resources he has to provide. As a result, the so-called "last chunk" problem, cf. [3], might arise which inhibits the data dissemination process and makes individual chunks starve in the network. Hence, a shortage of providers for the chunk appears and the remaining providing peers may be overloaded. As the result, the file exchange is delayed. Therefore the robustness, i.e. the capability of the CDN to be resistant against this user behavior, has to be considered.

In CDNs users are allowed to join or leave the system whenever they want. In mobile P2P CDNs it is likely to have a higher churn rate, because a peer might also leave the system due to loss of connectivity. As we consider a cellular network with dedicated channels, a mobile user is assumed to have a fixed access bandwidth. The movement of a user therefore only affects the system in terms of a higher churn rate. This means that the user not only leaves the P2P system when he wants to, but also when losing radio coverage. In the latter case, the user might get online again after a very short time, i.e. as soon as he reconnects to the cellular system. We subsume both, the user shutting down the application and the user losing connection, into the churn rate.

The various P2P platforms differ significantly in the actual implementation of the cooperation algorithms. The individual peer selection as well as the chunk selection mechanism lead to different system behaviors and performance results. The detailed performance of the strategies is further determined by the actual peer characteristics and the peer behavior. The peer characteristic includes, among others the available upload and download bandwidth, as well as the number of parallel upload and download connections. The mobility of a user may change these peer characteristics over time. Therefore, the question arises whether a cooperation strategy can leverage the effects of selfishness, mobility, or heterogeneity and establishes an efficient, fair and robust content distribution.

In this paper we propose a new cooperation strategy, which has an efficient chunk diffusion behavior and unlike the least-shared-first (LSF) strategy as used by BitTorrent it is based only on existing local information available at the peer. The key performance characteristic from the user's point of view are (a) efficiency, i.e. the time to download a file, and (b) fairness among the peers, in particular that none of the peer has to upload too much data and experiences much longer download times than the others. From the system's point of view, efficiency as well as robustness are the key measures. The latter one is the capability of the CDN to be resistant against user behav-

ior or heterogeneity. Within this work, an eDonkey-like random chunk strategy, a BitTorrent-like LSF strategy, and our proposed CycPriM cooperation strategy are evaluated with respect to their efficiency, fairness, and robustness.

The paper is organized as follows. In Section 2 we discuss other approaches to fight the ”last chunk” problem. Section 3 formulates the problem in detail. The three cooperation strategies considered in this work are described in Section 4. Section 5 outlines the proposed system model, its parameters, and the considered distribution scenarios. Section 6 compares the performance of the considered cooperation strategies. Finally, Section 7 concludes this work and gives a brief outlook on future work.

2 Related Work

Cooperation Strategies define how peers interact with each other. Penserini et al. [4] model peers within a special framework and research methods how to judge the cooperation strategies build up by the reasoning mechanism within the peers for a given task.

If we focus on content distribution the task is to quickly disseminate one or more files to a group of peers. Incentives help these groups to collaborate even if some of the peers behave selfish and thus may prevent the last chunk problem. In [5] the authors characterize the problem of selfish peers and shown that solutions based on the local knowledge on a peers behavior does not scale with an increasing peer group size and other options have to be considered. But it has also been shown in [6] that there are possibilities to reach near optimal sharing behavior even in large groups and with high churn using incentives. A comparison between different incentive strategies is presented in [7]. Many of these incentive mechanisms are based on the idea of trading upload volume against download volume by using some sort of virtual currency. However, if a new peer without any part of the file enters the system, it has to earn an amount of this currency in order to pay for its download. To encounter this problem Liao [8] propose to reward peers for staying in the system instead of endowing new peers the possibility to download parts of the file. In contrast to this, Anagnostakis and Greenwald [9] believe that incentives based on virtual currencies are either ineffective or much too complex. Therefore they propose a strong incentive, based on the idea of barter trade. In their proposed architecture peers prefer to trade parts of files with other peers, which provide them with parts they currently need and vice versa. Incentives might guarantee a good cooperation between peers. But that does not necessarily mean that the exchange of data is fair for all peers, as it is demonstrated by Veciana and Yang [10]. However, all these approaches define incentives in order to stabilize the cooperation of peers. In our work we propose an interaction scheme without incentives and compare it to some of the architectures proposed above. Another proposal for an incentive-less architecture is [11]. But in this work Hales assumes that peers are able to copy the neighborhood and the behavior of other peers, which is very hard to achieve in practice and is not necessary with our approach.

Le Fessant et al. [12] showed measurement results of several peer-to-peer content distribution systems and concluded that these systems provide the opportunity to gain efficiency by clustering peers with the same interests and regional togetherness. The idea of selecting proper peers in order to increase the efficiency was also discussed in [13]. This contribution proposes to build hierarchical structures in order to cope with problems locally and not to affect the whole network. In the same year Ng et al. [14] discuss how measurement-based optimization may influence bandwidth demanding peer-to-peer systems. The question which topologies are created by peers trying to minimize their connections and optimize the response times to their overlay neighbors is discussed in [15]. The peer selection may also have an influence and can optimize the dissemination of a file in the P2P system based on random selection of parts of the file which

is shown in [16]. In our contribution we do not restrict peers in their communication with other peers. A peer may interact with any other peer in the system. Thus, we focus on the timing when two peers interact and on the information they exchange, i.e. the scheduling within the resource access control and the chunk selection.

With the optimal selection of neighbors in the overlay it may be possible to structure and optimize the peer-to-peer network. However, the data exchanged between two interacting peers also influences the file dissemination. Felber and Biersack [17] discuss which peer and chunk selection strategies are able to cope with flash crowd effects. A more detailed look on the behavior of the content distribution systems which we compare with our solution are presented in [18] and [19]. Whereas [18] focuses on the eDonkey network, Legout et al. [19] regard the BitTorrent architecture.

Beyond these performance measures it is also crucial that the content distribution system does not decay in adverse circumstances. This feature is called robustness and is discussed in [20] and [21]. While [20] research the robustness of algorithms that distribute dictionaries over a group of peers, Triantafillou et al. apply the concept of robustness to content distribution peer-to-peer networks in [21]. The resulting content distribution system is complex and uses peer clustering. In contrast to this, our proposed architecture is flat and avoids the overhead needed to stabilize a hierarchical architecture.

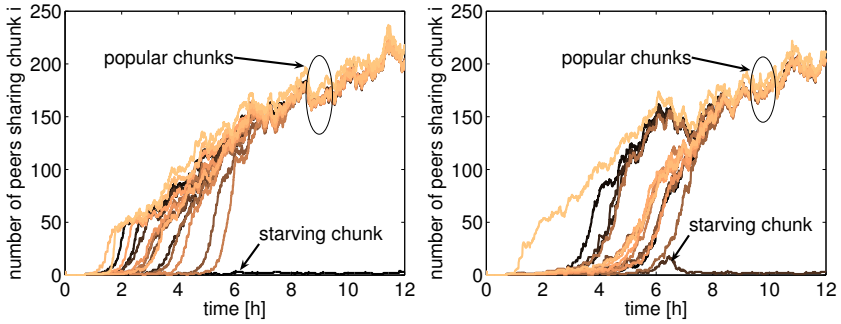
Despite the large literature on content distribution schemes, there exist only a few works on P2P CDNs in a mobile environment, especially in infrastructure-based wireless networks. Recently, mobile P2P research projects have received high attraction which is reflected by the popularity of the latest IEEE workshops MobiShare and MP2P. However, most the work addresses structured P2P networks based on distributed hash tables as lookup-service or considers mobile ad hoc networks. For example, [22] propose a cooperative P2P scheme that allows parallel download of the content based on swarming protocols in wireless ad hoc networks.

There are several possibilities to improve the performance of content distribution in mobile environments. In previous works, we studied mobile P2P CDNs based on an eDonkey and proposed an enhanced architecture concept [23] leading to a higher performance [24]. In [25], the size of the chunks and blocks was optimized for a mobile environment to minimize the required download times. In this work we investigate how cooperation strategies can further improve the performance of P2P CDNs in mobile environments and their special characteristics, cf. Section 5.2. For a mobile P2P CDN scenario it is of particular interest to investigate the robustness of a system, due to the increased churn behavior of peers and the poor connectivity of the peers which may increase the selfishness of peers.

3 Problem Formulation

P2P content distribution mechanisms which apply MSD split files into chunks and blocks which are subparts of chunks. For the eDonkey application for example, the chunk size is typically 9.5 MB and the block size is 180 kB. A downloading peer requests blocks from serving peers, i.e. sources of that file, and might download from these sources in parallel. As soon as a peer has downloaded a complete chunk, it becomes a source for the file, i.e., it can redistribute the already received chunks. As a result, MSD does not rely on a single source and can therefore avoid bottlenecks and overcome churn, which might be crucial for operating in cellular networks. From a system's point of view, we can characterize the CDN by the term efficiency, fairness and robustness.

The efficiency is determined by an optimal usage of provided resources. In the context of content distribution systems using MSD, this can be analyzed by investigating the download time



(a) eDonkey-like random selection strategy (one upload)

(b) BitTorrent-like LSF strategy (four parallel uploads)

Fig. 1: Illustration of the last chunk problem

and the availability of the chunks of the file. If the chunk dissemination is not in equilibrium, i.e. some chunks are less available, there is more upload capacity for the popular chunks which might be missing for the unpopular chunks. This lowers the effectiveness of the content distribution. Fairness means that peers contributing with the same resources should be treated in the same way. This means, that they should experience the same download time and the same uploaded volume. According to [26], robustness is a characteristic of a system, being stable under failure, misuse, and overload. For content distribution networks we see this demand fulfilled, if the system is resistant against changes in the user behavior, i.e. the file transfer times and upload volumes are stable even with selfish peers in the network. Hence, a content distribution network which is efficient, fair and robust can provide a reliable download experience with short download times and small upload volumes, which are the key characteristics from a user's point of view.

The coordination of the peers to enable the efficient, fair and robust MSD is realized by the *cooperation strategy*. Its task is to decide *a)* which peers requesting for blocks are served by an uploading peer using a priority function (like first-come-first-serve) and *b)* which is the next chunk to download by a downloading peer, the so-called chunk selection strategy.

The performance of the P2P CDNs is determined by the implementation of this cooperation strategy, the *peer capabilities*, i.e. the available capacity resources for exchanging files (upload and download bandwidth, maximum number of inbound and outbound connections), and the *user behavior*. The latter one includes the file request pattern, the churn behavior, and the willingness to participate in the network, i.e. selfish or altruistic peers. As an extreme case of selfish peers we consider *leechers* which immediately leave the system after finishing the download of a file. In such a case, the leaving users reduce the availability of chunks. As a result, in the worst-case a specific chunk may get rare in the system, i.e. there are only a few sources for this chunk in the CDN. Figure 1 depicts two examples for such a behavior. It shows the spreading of chunks, i.e. the number of sharing peers for each chunk i of the file, over time in a leeching scenario. Most of the chunks are spreading throughout the system, with label "popular chunks" in Figure 1. One of the chunks denoted as "starving chunk" will not spread because of the leeching behavior of the peers. As a result, unfinished peers, which seek the final, last chunk have to wait until they receive the chunk from one of the initial sources. This leads to large download times. This problem is called the *last chunk problem* in this paper.

The question arises whether a cooperation strategy can leverage the selfish behavior of the peers and establish an efficient, fair and robust content distribution.

4 Cooperation Strategies

The cooperation strategy defines the selection of the next peer being served as well as the choice which chunk should be transferred, i.e. the priority function and the chunk selection strategy. Both decisions have direct impact onto the last chunk problem especially in mobile networks. Three possible strategies are described and their influence on the chunk dissemination is discussed: the eDonkey-like random chunk strategy, the BitTorrent-like LSF strategy, and the proposed new *cyclic priority masking (CycPriM)* strategy.

Figure 2 (on the next page) depicts the first two rounds of the distribution process of a file for the three different cooperation strategies. In the example the file consists of two chunks. Every round is depicted by a box. In the boxes, the peers are drawn together with the chunks they own. There are three initial seeding sources $(S1, S2, S3)$, which share all chunks of a file. Furthermore there are ten leeching peers $(P0, \cdots, P9)$ who want to download the file. During each round, every peer that has at least one chunk uploads a specific chunk to another peer. The chunk dissemination at the end of the round is depicted by the next box. The arrows between two peers visualize the upload connection, that has been active during the last round. If a leecher finishes the complete download at the end of a round, it will leave the CDN immediately. In order to depict this situation, these peers are encircled and marked with an X.

The first round of the download process is equal for all strategies: peer P0 downloads chunk 1 from S1, peer P1 chunk 2 from S2, and peer P9 chunk 1 from S3. After the first round of transferring chunks, the dissemination behaves different for each strategy. The strategies will be explained the following.

4.1 Random Chunk Strategy

Applying the random chunk strategy, like the one used by eDonkey, a downloading peer issues a request to a sharing peer. The sharing peer queues this request in a first-come-first-serve (FCFS) manner. As soon as the downloading peer is served, it chooses a random chunk, which it has not downloaded yet. In the example of Figure 2 (case ”random”) peer P0 selects its missing part in the second Round and leaves the CDN after downloading from the network. In addition, peer P2 and peer P4 choose chunk 1 randomly and independently and download it from S1 and S3, respectively. The *random chunk strategy* relies on the random selection of required chunks. The randomization avoids that all downloading peers select the same chunk. Thus, the simultaneously downloading peers get different chunks and can therefore exchange these different chunks in the further distribution process. This fosters the cooperation among peers. As will be shown in Section 6.2.1 this strategy performs well as long as peers are altruistic, i.e. as long as peers are willing to share after they have completed their download. However, if most of the peers are leeching, the last chunk problem occurs frequently.

4.2 Least-Shared-First Strategy

The LSF strategy uses the same priority function like the random chunk strategy, i.e. requests are served in a first-come-first-serve manner. However, the chunk selection differs. Peers choose

as next chunk to be downloaded the one which is *least-shared* in the CDN. This means that this chunk has the smallest number of sharing peers, compared to the number of possible sources for other chunks. If there are several chunks fulfilling the least-shared criteria one of these is chosen randomly. In order to know the least-shared chunk, it is necessary that everybody is aware of the numbers of peers sharing a specific chunk. Thus it is necessary to make this information global available, e.g. using a tracker. Figure 2 (case "least-shared-first" - Round 2) shows the selection for the given example. Peers choose the least-shared chunk not yet being downloaded at the moment of the download.

A peer using this strategy selects the required chunk, which has the lowest number of providing peers. This mechanism results typically in a evenly spread number of sharing peers for all chunks of the file. However, there are cases in which this is not true. As it will be shown in Section 6 this strategy is very efficient as long as the chosen chunk is the least-shared one at the end of the download of this chunk. However, the decision which chunk is the currently least-shared one is done at the beginning of the download. Thus, another chunk can get the least-shared one, which undermines the homogeneous chunk dissemination.

4.3 CycPriM Strategy

A sharing peer should take care of the homogeneous chunk dissemination in the network to avoid the last chunk problem. If there is only local information available, the sharing peer should deliver chunks in an ordered way.

The basic idea is to distribute the file in upload rounds. In each upload round the mechanisms tries to distribute a sequence of all chunks to requesting peers, as long as requests for these chunks are available. If no request is available for one of the chunks in the sequence, this chunk is skipped

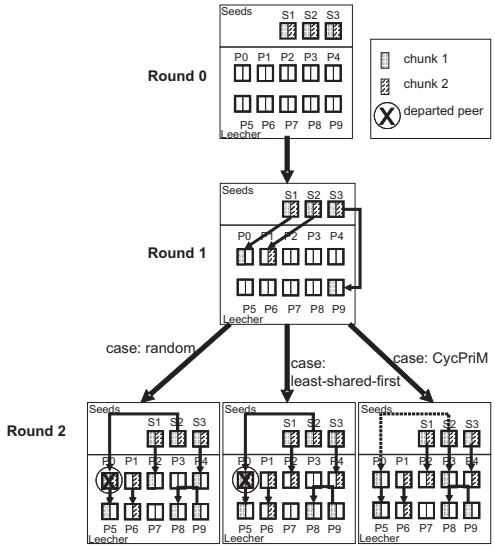


Fig. 2: Example of chunk exchange for the cooperation strategies in the first two rounds

and the next chunk of the sequence is chosen to be distributed. After the complete sequence is processed, a new upload round starts. In order to prevent the downloading peer from selecting any other chunk, we propose the following chunk selection strategy: The uploading peer offers only this one chunk. If a peer accepts this offer, or no peer wants the chunk, then the next chunk from the cycle is chosen. We call this strategy *CycPriM* (case "CycPriM" in Figure 2) which stands for *cyclic priority masking*. In Figure 2 the example shows, that all seeds share the opposite chunk in the second transfer phase, as they did in the first. For example seed S2 has transferred chunk 2 in round 1, thus S2 will distribute chunk 1 in round 2. Peer P0 would be served by seed S2 in the second round. However, peer P0 has been masked because it already has chunk 1. The next peer wanting to download chunk 1 (in the example: peer P3) is served instead. As we will see in Section 6 the upload of the file in rounds leads to a robust CDN with respect to the peer capabilities without requiring additional overhead.

5 Simulation Description

In this section we first describe the unified system model, which is used to compare the cooperation strategies on a equal basis. Thereafter, we provide a description, which kind of mobile scenarios we consider in this work.

5.1 System and Queuing Model

We assume a *"hybrid"* P2P CDN architecture. That means, the information where resources are located is offered by a central entity, which we call the index server. The index server only keeps track of the peers being connected to the CDN and the files the offer. It is not responsible for providing information about the dissemination of chunks. We focus on the sharing behavior of CDNs and therefore assume that global information about chunks shared in the network is available. The evaluation neglects the overhead caused by frequent status update messages which are necessary to maintain this information. Hence, the performance of the LSF strategy, which is the only strategy relying on this information, is actual worse than discussed next, since the transmission of the overhead consumes additional resources. As a result the download time is actually longer than discussed in this work.

In our model we assume that a peer, which is interested in a file, requests all available sources at the index server. Therefore each peer knows all sources, which are connected to the network at the moment of the request. New sources will be discovered by a periodical source request messages of a downloading peer, which are sent every ten minutes. However, this signaling traffic is neglected in the system model, because all strategies are effected in the same way. This means, it has no impact on the available resources and requires no transmission time. Table 1 summarizes the required overhead of the different strategies. However, we omit all overhead for joining or leaving the network and the additional overhead of the LSF strategy for updating the chunk dissemination in our system model.

In order to share a file it is divided into chunks. In [25] it was shown, that further transmission fragmentation of chunks into blocks is not necessary and can even worsen the download time in mobile scenarios. Thus the model assumes that peers try to transfer complete chunks. Nevertheless chunk fragmentation is possible due to a connection loss between two peers, which might be caused by the churn behavior, i.e. a peer leaves the CDN although it has active data transfers. This might happen due to connection loss or the end of battery power. In this case the received data is stored and the chunk will be finalized without any retransmission of already received data.

The model assumes that every time a peer receives a new source, it sends a download request containing an identifier for all required chunks. If the peer addressed by the request has none of the required chunks, the request is neglected. Otherwise the addressed peer queues the request in a waiting queue. The queue storing all peer requests is ordered to some priority function, as described in Section 4. We assume all peers to use the same priority function. A peer can serve at most up to Q_A peers in parallel. Requesting peers being served simultaneously share the uploading bandwidth of the providing peer. The limitation to maximal Q_A parallel uploads guarantees a minimum amount of bandwidth per P2P connection, which is at least $\frac{1}{Q_A}$ of the providing peers upload bandwidth. Thus, restricting the peer to only serve one other peer will be expressed by one parallel upload, i.e. $Q_A = 1$ However, if a downloading peer can not handle the offered bandwidth due to restrictions of his own download bandwidth, the surplus is equally divided among the other peer connections. After a download is completed, the providing peer verifies if there are any other chunks requested, which he actually shares. In this case the request is re-queued. Otherwise the request is discarded.

A major influence factor in the system model is the number of parallel uploads. Reducing the number of parallel uploads to one, which means no parallel uploads at all, could possibly enhance the diffusion. This is reasonable by considering a scenario in which only one source exists at time t_0 which provides a file consisting of one chunk. All peers are assumed to have the same upload bandwidth, which allows them to upload a single chunk within the time Δt using the complete bandwidth. If we assume a single connection per peer the original source will have replicated it's chunk after the time Δt. After another Δt both peers copied their chunk to other peers and so on. Thus we can calculate the number of chunks available (C) after $k\Delta t$ by the compound interest formula $C(k) = 2^k = (1 + PU)^k$, where the number of parallel uploads $PU = 1$. For the case of $Q_A > 1$ the source is able to reproduce its chunk PU times, but the duration for this transaction will be $PU\Delta t$, as it has to split the available upload bandwidth for PU connections. Hence for the general case of PU parallel uploads and a single source starting to upload at time $t_0 = 0$the number of available chunks is $C(t) = (1 + PU)^{\lfloor \frac{t}{PU\Delta t} \rfloor}$. It holds $(1 + 1)^{\lfloor \frac{t}{1\Delta t} \rfloor} \leq (1 + PU)^{\lfloor \frac{t}{PU\Delta t} \rfloor}$, for $1 < PU \in \mathbb{N}$ and $t > 0$. This means that an outbound degree of one performs best in theory. However, selfish behavior and churn might lead to other results in practice, which will be discussed in Section 6.

Table 1: Overhead required by the cooperation strategies

cooperation strategy	overhead for joining	overhead for leaving	overhead for updating chunk dissemination
random strategy	X		
LSF strategy	X	X	X
CycPriM strategy	X		

5.2 Mobile Scenario Description

The performance of the cooperation strategies is evaluated for different scenarios in which (a) the user behavior, i.e. the selfishness or altruism of users as well as churn, and (b) the peer capabilities, i.e. the number of parallel uploads per peer, are varied. Although user mobility has

an impact on the capacity of the cellular system, the effect of mobility for a P2P user with fixed access bandwidth in a cellular network can be described by a simple ON/OFF process. If the user is granted access to the wireless system, he may start its P2P application. Due to the loss of radio coverage, the peer appears to be offline in the P2P system. This process is modeled with random variables for the online and offline times. Thus, we may subsume (i) the online and offline behavior of a peer due to switching the P2P application on or off and (ii) the effect of mobility of a peer with a fixed bandwidth in a cellular network. In the following, we use the term "churn" which is described by a random variable taking (i) and (ii) into account.

In all considered scenarios in this paper, we assume churn.

We define the churn ratio γ as the ratio between the online time T_{on} and the offline time T_{off} of an arbitrary peer, formally

$$\gamma = \frac{T_{on}}{T_{off}} \ . \tag{1}$$

In our simulations results, T_{on} and T_{off} follow an exponential distribution with an average online respective offline time of $\lambda^{-1} = 1\,\mathrm{h}$. As a result, γ is a random variable whose cumulative distribution function can be derived as follows

$$\mathrm{P}\left(\frac{T_{on}}{T_{off}} \leq t\right) = \int_{a=0}^{\infty} \mathrm{P}\left(\frac{T_{on}}{a} \leq t\right) \mathrm{P}\left(T_{off} = a\right)\, da \tag{2}$$

$$= \int_{a=0}^{\infty} \left(1 - e^{-\lambda a t}\right) \lambda e^{-\lambda a}\, da = \frac{t}{1+t} \ . \tag{3}$$

The probability density function of the churn ratio γ is accordingly

$$\mathrm{P}\left(\gamma = t\right) = \frac{d}{dt}\mathrm{P}\left(\gamma \leq t\right) = \frac{1}{(1+t)^2} \ . \tag{4}$$

Figure 3(a) shows the cumulative distribution function of the churn ratio γ for the parameters as used in the simulation scenarios, while the probability density function is given in Figure 3(b). As can be seen from these results, the dynamics of the peers in the system due to churn might be quite high. This assumption is reasonable because of the considered mobility of the peers.

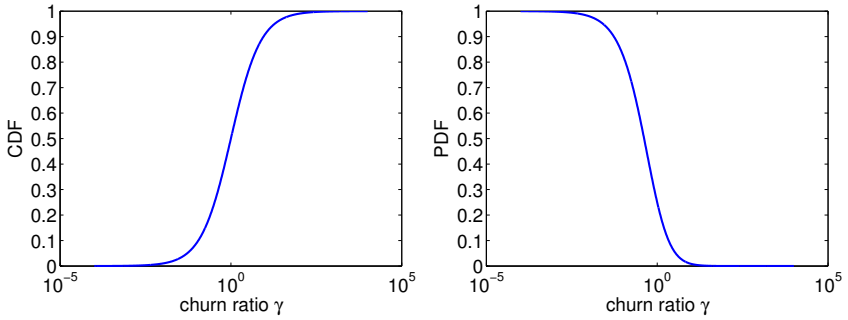
The dynamics of the system can also be identified when deriving the probability that a peer is online for an outside observer in two successive on and off phases, i.e. $\mathrm{P}\left(\frac{T_{on}}{T_{on}+T_{off}}\right)$. In an analogous way, we obtain

$$\mathrm{P}\left(\frac{T_{on}}{T_{on}+T_{off}} \leq t\right) = \int_{a=0}^{\infty} \mathrm{P}\left(\frac{a}{a+T_{off}} \leq t\right) \mathrm{P}\left(T_{on} = a\right)\, da \tag{5}$$

$$= \int_{a=0}^{\infty} e^{-\lambda a/t - \lambda a} \lambda e^{-\lambda a}\, da = t \ , \tag{6}$$

and $\mathrm{P}\left(\frac{T_{on}}{T_{on}+T_{off}} = t\right) = 1$ as probability density function.

The selfishness of peers is investigated in a worst-case scenario, the *leeching scenario*, and a best-case scenario, the *diffusion scenario* in which the peers are almost altruistic. In the diffusion scenario all peers finishing the file transfer will serve as uploading peers during the rest of the simulation. However, the churn behavior is not affected by becoming a source and the peers will go on to leave and reenter the network. From the diffusion scenario it can be concluded, whether a strategy uses the available resources efficiently or not. Against this, a peer finishing the download will depart from the net shortly after in the second scenario, which is called the *leeching scenario*.The selfishness in the leeching scenario will demonstrate if a strategy can deal with uncooperative peer behavior.



(a) Cumulative distribution function (b) Probability density function

Fig. 3: Churn γ as ratio of the online time T_{on} and the offline time T_{off} of a user, i.e. $\gamma = \frac{T_{on}}{T_{off}}$

Another influence factor on the system performance are the peer capabilities. In all scenarios peers are assumed to have the same bandwidth capabilities. We consider a mobile GPRS P2P CDN scenario as it will clearly reveal possible performance problems of each cooperation strategy due to the low available bandwidth at the sharing peers. Thus we assume that the peers have GPRS access with an upload bandwidth of 12 kbps and a download bandwidth of 48 kbps. The maximum number of outbound connections, i.e. parallel uploads of a peers, might strongly impact the system and is varied between one and four connections.

For the numerical evaluation we simulated ten runs for each scenario and each cooperation strategy. The considered network consists of 1000 peers. These peers are interested in one file which is provided by three initial seeds. The file has a size of 8 MB and consists of 17 chunks. Each chunk has a size of 480 kB. At the beginning one peer is chosen randomly to download the file. The interarrival time used to schedule the file requests of the other peers follows an exponential distribution with a mean of 80 seconds. Thus the file is requested much more often than it can be provided in the start up phase of the download.

6 Evaluation of the Strategies

In this section we characterize the system behavior in the diffusion scenario and in the leeching scenario. Additionally, the efficiency, fairness and robustness are compared. In order to illustrate the behavior we show one single simulation. These figures can not provide statistical firm data, but depict differences of the chunk diffusion behavior of the cooperation strategies. If the last chunk problem is inherent to a cooperation strategy, it is possible to spot it in every single run. However, which chunk this is, is completely random. Thus, for a large number of simulation runs the mean value of the chunk dissemination would be the same for all chunks although in every single run there is one chunk, which is hardly available in contrast to the others. Confidence intervals of the results of the download efficiency are omitted for reasons of clarity in Section 6.1 and Section 6.2. We have indicated in [27] that the confidence intervals are small enough to separate the performance results of the cooperation strategies from each other. Thus, the qualitative statements and results regarding the application of cooperation strategies in different scenarios

are identical, when taking significance levels of the simulation results into account. In order to show the statistical credibility, we take a closer look on some exemplary scenarios in Section 6.3.

6.1 System Behavior in the Diffusion Scenario

The diffusion scenario represents an ideal system. All peers behave altruistic and stay in the system after having downloaded the file. The scenario is used as a reference scenario for the discussion of the leeching scenario where robust and fairness is of major interest.

Download Efficiency

The left part of Figure 9 depicts the average download time in the diffusion scenario with the associated 95% and 99% quantiles. The average download times are in the same order of magnitude. They show that all cooperation strategies are very efficient and permit a short download time. However, the download time depends highly on the number of available sources and their upload bandwidth. Due to the altruistic behavior a peer that starts the download late sees many sources for the file where it can choose from. Hence, a peer arriving late experiences a short download time and has to upload less data.

The download time is related to the availability of the chunks. The more peers provide a chunk the faster a download can be completed. The availability, cf. Eq. (8), for the diffusion scenario is depicted in the left part of Figure 6. It nearly reaches the optimal value of 100% for all strategies.

Fairness

Although an altruistic behavior of peers subsumes fairness among these entities, we outline this system characteristic of the diffusion scenario for later comparison with a leeching behavior of peers. The notion of fairness is provided in Section 6.2.5, Eq. (9).

The left part of Figure 7 shows the fairness index for all three strategies. The indices are all in the same order, independently of the observed performance measure, i.e. the upload volume or the download time, and the number of parallel uploads.

6.2 System Behavior in the Leeching Scenario

The last chunk problem and the associated decrease of efficiency is assumed to be mainly caused by the selfish behavior of peers. We therefore continue to investigate the last chunk problem in the *leeching scenario*, i.e. a peer leaves the system immediately after completing the download. Thus, we examine the evolution of the number of peers sharing a chunk for the different cooperation strategies for the leeching scenario in detail.

6.2.1 Random Chunk Strategy

The starting point for the investigation is the ”random chunk strategy” which is the most simple one. This strategy is used in the eDonkey system. Figure 1(a) depicts the number of peers sharing a chunk throughout time using at most one upload connection. Each of the 17 chunk of the file is represented by one line in Figure 1(a). It is evident that the number of peers sharing a chunk does

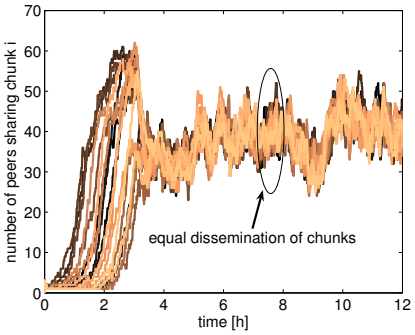


Fig. 4: Chunk dissemination in the leeching scenario using the least-shared-first strategy with one upload connection

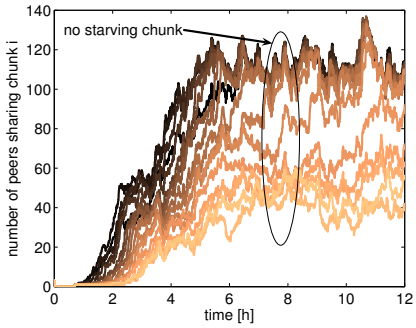


Fig. 5: Number of peers sharing a chunk for the leeching scenario using the CycPriM strategy variant with one upload connection

not rise equally. At the beginning some chunks are reproduced while others are not. A chunk being shared by some more peers than only the seeders becomes independent to churn, which allows a faster reproduction. Additionally, in the beginning there are several peers that share only one chunk. Therefore, they will distribute this chunk only, as long as they do not download any other chunks. Thus, these chunks will be more often downloaded than the other ones.

After the first chunk was distributed among the requesting peers, a peer downloads another chunk and this chunk spreads in the network like the first one. Later this leads to a situation in which nearly all chunks are often shared. This is the point where the leeching behavior harms the system. If a peer downloads this ”starving chunk” there are two possible situations. In the first case it has the other chunks already. Thus this peer has finished the download and leaves the CDN. In the second case some or all of the other chunks are still needed. Then the peer will be able to download the remaining chunks in a short time, because of the high number of peers sharing the other parts. Afterwards it leaves the network. In any case the time this peer provides this chunk as an additional source is very short. As a result, one chunk is shared only by a few peers and required by many peers which forces them to wait for this final chunk to be transferred. In Figure 1(a) this least shared chunk is referred to as ”starving chunk”.

6.2.2 Least-Shared-First Strategy

The least-shared-first strategy tries to overcome the last chunk problem by favoring rare chunks. In order to choose the least-shared chunk, it is imperative to know the dissemination of chunks at the moment of the chunk request. Thus, this information has to be up-to-date and global accessible, e.g. it is provided by a tracker.

In Figure 4 the number of sharing peers is visualized for each chunk using at most one upload connection of a serving peer. In the beginning the different chunks spread equally within the network. The number of sharing peers stays much closer together than for the same scenario using random strategy, cf. Figure 1(a). This is caused by the fact, that after the first peer has downloaded a chunk from the providing peers. Other downloading peers will not request this chunk until the other chunks achieve a higher dissemination. The least-shared-first strategy with one upload connection can stabilize the number of sharing peers for all the chunks and therefore

improves the download speed. As we have illustrated in Figure 1(b) for a higher number of parallel uploads, there are situations where the least-shared-first strategy is no longer able to prevent a starving chunk. The reason is that the least-shared chunk is determined at the beginning of the download. However, this is not necessary the least-shared one when the download ends. With a rising number of parallel uploads it gets more difficult to decide the least-shared chunk at the end of the download before it starts.

6.2.3 CycPriM Strategy

Figure 5 shows the number of peers sharing one chunk for the CycPriM strategy using one upload connection for a single simulation run in the leeching scenario. At the beginning the different chunks stay relatively close together. After this they spread in the CDN and the number of peers sharing one chunk is very dynamic. The most popular chunk is three times more often shared than the most unpopular one. As we will show in Section 6.2.5 this increases the download time. However, it prevents starving chunks in the network.

6.2.4 Chunk Availability

The discussion of the three strategies shows that the existence of a starving chunk may decrease the efficiency of a cooperation strategy dramatically. In order to measure the comparative difference between the average chunk dissemination and the number of peers sharing a rare chunk, we define the *rare chunk availability* A. Let N be the number of chunks and

$$\forall_{0 \leq i < N} A_i = \frac{\int_{t_0}^{t_1} C_i(t) dt}{t_1 - t_0}, \tag{7}$$

where $C_i(t)$ is the number of peers sharing chunk i at time t. We call A_i the availability of a chunk in the interval from t_0 to t_1. This value reflects the average number of peers sharing chunk i in the corresponding interval. The *rare chunk availability* A is the minimal availability of a chunk normalized by the average availability of all chunks, i.e.

$$A = \min_{0 \leq i < N} \left\{ \frac{A_i}{\frac{1}{N} \sum_{0 \leq i < N} A_i} \right\}. \tag{8}$$

A low value of A indicates starving chunks. The left part of Figure 6 shows the rare chunk availability for the regarded scenarios. As expected, the rare chunk availability in the diffusion scenario is nearly 100% for all strategies. For the leeching scenario, the last chunk problem occurs at the random strategy and the least-shared-first strategy with a high outbound degree. This is reflected by the low values presented in the right part of Figure 6. The CycPriM strategy has a rare chunk availability of about 50%, which is a result of the wide spectrum of numbers of sharing peers already discussed in the last section. The random strategy and least-shared-first with multiple uploads reveal the last chunk problem. Therefore, the efficiency of the algorithms is disrupted by the user behavior in the leeching scenario.

6.2.5 Fairness and Robustness

In the case of selfish peers it is important, that the strategies accomplish fairness among the downloading peers. In particular all peers should experience the same download time and upload

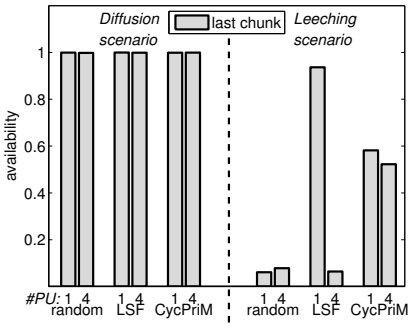


Fig. 6: Comparison of rare chunk availabilities for different number of parallel uploads (PU)

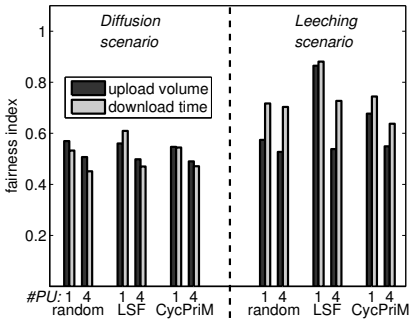


Fig. 7: Jain's fairness index achieved by the cooperation strategies for different number of parallel uploads (PU)

the same amount of data. We choose Jain's fairness index presented in [28] to quantify fairness. Jain's fairness index is defined by

$$J = \frac{(\sum_{i \in M} x_i)^2}{|M| \sum_{i \in M} x_i^2}, \tag{9}$$

where x_i are the values of the considered performance measure, M is the set of all measurement values, and $|M|$ is the number of measurement values. It holds $J = \frac{1}{1+c_x^2}$, where c_x^2 is the corresponding coefficient of variance. The fairness index returns values between zero and one, i.e. $0 \leq J \leq 1$. Low values of the fairness index indicate an unfair system, while a fairness index of one describes a completely fair system, where all users experience exactly the same performance w.r.t. the considered measure.

Figure 7 visualizes J for the upload volume and download times experienced by the peers using the different cooperation strategies. The fairness index of all cooperation strategies in the leeching scenario is mostly on the same level, see right part of Figure 7. Only the least-shared-first strategy with one parallel upload has a fairness index that is significantly higher. The fairness indexes of the other strategies is more or less the same.

A cooperation strategy is considered to be *robust*, if the amount of data uploaded and the time needed to finish the download are close to the values obtained in the diffusion scenario. Figure 8 shows the data volume the peers have uploaded. In all scenarios the mean value of uploaded data volume is the same. The 95% quantile and the 99% quantile show how much individual peers have contributed in uploading. From this figure the high fairness index of the least-shared-first strategy with a single upload becomes evident. This variant is the only strategy that assures single peers not to upload much more data than two times of the download. All other strategy variants have much higher values for the 95% quantile and 99% quantile of the uploaded volume.

The average download time of the peers in the leeching scenario is visualized by the right part of Figure 9. The LSF strategy with one parallel upload has download times, which are in the same order as in the diffusion scenario (left part of Figure 9). This feature and the low upload volume for each peer demonstrate the very good robustness of the LSF strategy with one parallel upload against leeching behavior. The CycPriM strategy permits fast download times in the leeching scenario as well. It can not provide always the short download times of the LSF strategy. However,

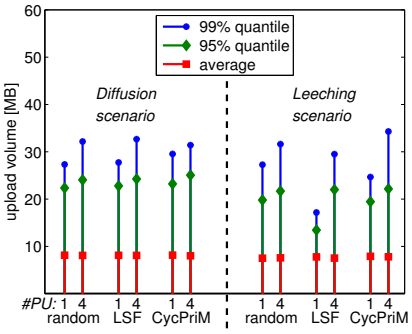


Fig. 8: Data volume uploaded by peers for different number of parallel uploads (PU)

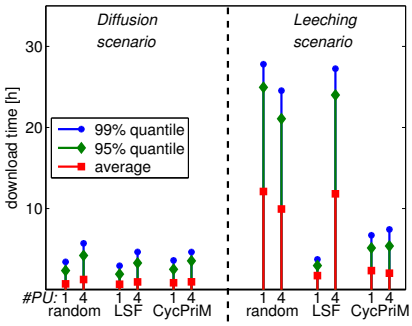


Fig. 9: Download times experienced by peers for one and four parallel uploads (PU)

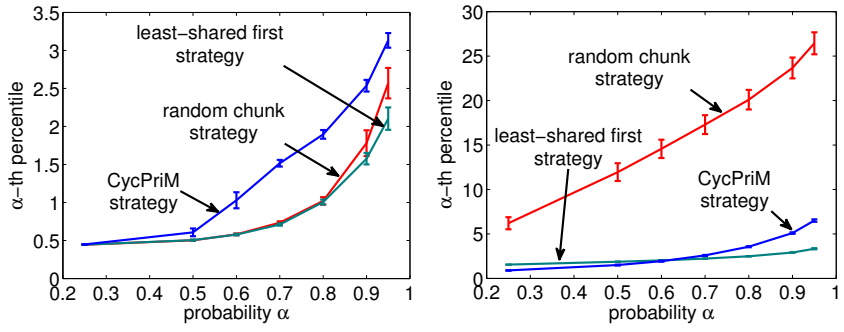
the robustness of the CycPrim strategy does not depend on the peer capabilities, i.e. the number of parallel uploads as in the LSF algorithm. Figure 9 (right part) reveals, that the 99% quantiles of the download times for both variants of the CycPriM strategy are below ten hours. In contrast, the corresponding values of the random strategy and the LSF strategy with four parallel uploads are three times higher. The LSF strategy with four parallel uploads nor any of the variants of the random strategy are robust against selfish behavior of peers. This feature is only achieved by the CycPriM strategy.

As a result of the comparison, the least-shared-first strategy with one parallel upload outperforms the other strategies. The price to pay for this advantage is the higher amount of overhead needed by the least-shared-first strategy. Table 1 denotes all situations in which the cooperation strategy need to update global information, which has to be accessible to all the peers. Against all other strategies the LSF strategy needs to refresh the numbers of peers sharing one chunk. As we have stated in [27] the timeliness of this updates is crucial for the performance of least-shared-first strategy. Thus, if it is possible to keep the numbers of peers sharing one chunk up-to-date, then the LSF strategy should be used. If it is not possible to refresh this information the CycPriM strategy is a good alternative, which is a little bit slower but therefore only depends on local available information and is robust independent of the peer capability, i.e. the number of parallel uploads.

6.3 Statistical Credibility of Simulation Results

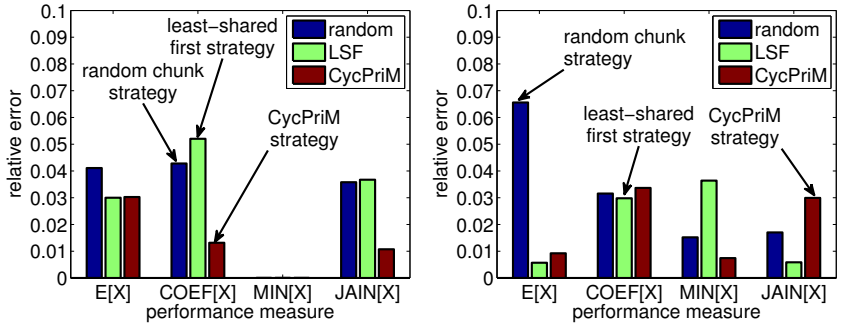
The statistical credibility of the simulation results presented so far is investigated in this section. We focus here on the diffusion and leeching scenario with one parallel upload. For the scenarios with four parallel uploads, we obtain similar results. The investigated performance measure is the download time experienced by a peer in the system. Different performance metrics like rare chunk availability also show similar statistical results.

Figure 10 shows the confidence intervals at a significance level of 95 % for the α-th percentile x_α of the download time X from ten conducted simulation runs, i.e. $P(X \leq \alpha) = x_\alpha$. Although the number of simulation runs is quite small, we see that the confidence intervals are also quite small. In particular, the drawn conclusions at the end of the previous section are not affected. For



(a) Diffusion scenario (b) Leeching scenario

Fig. 10: Confidence intervals at significance level of 95 % for α-th percentile



(a) Diffusion scenario (b) Leeching scenario

Fig. 11: Relative error of key performance measures from several simulation runs

the random chunk strategy, the confidence intervals are larger which is simply caused by the pure random dissemination of chunks in the network and the resulting potential risk of starving chunks. Especially in the leeching scenario, cf. Figure 10(b), this can be observed clearly. However, as this strategy leads too much worse results than the LSF and the CycPriM strategy using one parallel upload, this has again no impact on the derived conclusions.

Next, we consider the relative error of key performance measures of the download time X. They include the average value $E[X]$, the coefficient of variation $COEF[X]$, the minimum download time $MIN[X]$, as well as Jain's fairness index $JAIN[X]$ and are plotted in Figure 11. The relative error is computed as the length of the confidence interval at a significance level of 95 %, normalized by the average value of the performance measure from the ten individual simulation runs. It has to be noted that the restriction to ten runs has been done because of the excessive computational effort. Nevertheless, we see that the relative error is below 10 % for the considered performance measures in the different scenarios and we conclude that the results are statistically credible for validating our reasoning.

7 Conclusions and Outlook

This article analyzes the eDonkey-like random chunk strategy, the BitTorrent-like least-shared-first strategy and the CycPriM strategy for chunk dissemination in mobile P2P content distribution networks (CDNs) which are based on the multiple source download. As performance indicator the efficiency, fairness, and robustness of the cooperation strategies are chosen.

It is shown, that in cases where most of the peers are selfish, i.e. show a leeching behavior, the performance of the CDN can be significantly improved with an appropriate cooperation strategy. This strategy has to be robust. This mean that it has to enable fast downloads and fair upload volumes even with a selfish peer behavior. The detailed chunk distribution analysis demonstrates, that the random chunk strategy suffers from the last chunk problem. Thus, this strategy is not robust against leeching behavior and can not guarantee a fast file distribution in a scenario with selfish peers. The least-shared-first (LSF) strategy with one parallel upload overcomes these problems. It is highly efficient, fair, very robust and supports fast file distribution. However, it is necessary to update the global information about the number of sharing peers for every chunk. In practice it is not possible to have this global information. The study of LSF with multiple parallel uploads revealed that a CDN using LSF without perfect knowledge will easily diverge and therefore produce the last chunk problem. The CycPriM strategy is robust against leeching as well as against changes of peer capabilities, as it is neither dependent on global knowledge nor on restricting the number of parallel uploads. It is also efficient and should be used to save overhead or whenever it is not possible to keep track of dissemination of chunks in the network.

Acknowledgment
This work has been performed in the framework of the DFG Project MobileP2P (TR 257/22-1).

References

1. M. Perényi, T.D. Dang, A. Gefferth, and S. Molnár. Identification and Analysis of Peer-to-Peer Traffic. *JOURNAL OF COMMUNICATIONS*, 1(7), 2006.
2. S. Androutsellis-Theotokis and D. Spinellis. A survey of peer-to-peer content distribution technologies. *ACM Computing Surveys*, 36(4), December 2004.
3. A. Al Hamra and P. A. Felber. Design choices for content distribution in p2p networks. *ACM SIGCOMM Computer Communication Review, volume 35, issue 5*, October 2005.
4. L. Penserini, L. Liu, J. Mylopoulos, M. Panti, and L. Spalazzi. Cooperation strategies for agent-based p2p systems. *Web Intelligence and Agent Systems*, 1(1), 2003.
5. K. Lai, M. Feldman, I. Stoica, and J. Chuang. Incentives for cooperation in peer-to-peer networks. In *Workshop on Economics of Peer-toPeer Systems*, Berkeley, CA, USA, 2003.
6. M. Feldman, K. Lai, I. Stoica, and J. Chuang. Robust incentive techniques for peer-to-peer networks. In *5th ACM conference on Electronic commerce*, New York, NY, USA, 2004. ACM.
7. M. Feldman and J. Chuang. Overcoming free-riding behavior in peer-to-peer systems. *ACM SIGecom Exchanges*, 5(4), 2005.
8. W.-C. Liao, F. Papadopoulos, and K. Psounis. A peer-to-peer cooperation enhancement scheme and its performance analysis. *Journal of Communications (JCM)*, 1(7), 2006.
9. K. G. Anagnostakis and M. B. Greenwald. Exchange-based incentive mechanisms for peer-to-peer file sharing. In *24th International Conference on Distributed Computing Systems*, Tokyo, Japan, 2004.
10. G. de Veciana and X. Yang. Fairness, incentives and performance in peer-to-peer networks. In *Forty-first Annual Allerton Conference on Communication, Control and Computing*, Monticello, IL, USA, 2003.
11. D. Hales. Emergent group level selection in a peer-to-peer network. *ComPlexUs*, 3, 2006.
12. F. Le Fessant, S. Handurukande, A. M. Kermarrec, and L. Massoulie. Clustering in peer-to-peer file sharing workloads. In *3rd International Workshop on Peer-to-Peer Systems (IPTPS)*, San Diego, CA, USA, 2004.
13. L. Garces-Erice, E. W. Biersack, P. A. Felber, K. W. Ross, and G. Urvoy-Keller. Hierarchical peer-to-peer systems. *Parallel Processing Letters*, 13(4), 2003.

14. T. Ng, Y. Chu, S. Rao, K. Sripanidkulchai, and H. Zhang. Measurement-based optimization techniques for bandwidth-demanding peer-to-peer systems. In *22nd Annual Joint Conference of the IEEE Computer and Communications Societies INFOCOM*, San Francisco, CA, USA, 2003.
15. T. Moscibroda, S. Schmid, and R. Wattenhofer. On the topologies formed by selfish peers. In *Twenty-fifth annual ACM symposium on Principles of distributed computing*, Denver, CO, USA, 2006. ACM.
16. I. Norros, B. Prabhu, and H. Reittu. Flash crowd in a file sharing system based on random encounters. In *Workshop on Interdisciplinary systems approach in performance evaluation and design of computer & communications sytems*, Pisa, Italy, 2006.
17. P. Felber and E. W. Biersack. Self-scaling networks for content distribution. In *International Workshop on Self-* Properties in Complex Information Systems*, Berinoro, Italy, 2004. Springer.
18. K. Tutschku. A measurement-based traffic profile of the edonkey filesharing service. In *5th Passive and Active Measurement Workshop (PAM2004)*, Antibes Juan-les-Pins, France, 2004.
19. A. Legout, G. Urvoy-Keller, and P. Michiardi. Rarest first and choke algorithms are enough. In *6th ACM SIGCOMM on Internet measurement*, Rio de Janeiro, Brazil, 2006.
20. J. Risson and T. Moors. Survey of research towards robust peer-to-peer networks: Search methods. *Computer Networks*, 50(17), 2006.
21. P. Triantafillou, C. Xiruhaki, M. Koubarakis, and N. Ntarmos. Towards high performance peer-to-peer content and resource sharing systems. In *Conference on Innovative Data Systems Research (CIDR)*, Asilomar, CA, USA, 2003.
22. P. Michiardi and G. Urovoy-Keller. Performance analysis of cooperative content distribution in wireless ad hoc networks. In *fourth annual conference on Wireless on Demand Network Systems and Services*, Obergurgl, Austria, 2007.
23. J. Oberender, F.-U. Andersen, H. de Meer, I. Dedinski, T. Hoßfeld, C. Kappler, A. Mäder, and K. Tutschku. Enabling mobile peer-to-peer networking. In *Mobile and Wireless Systems, LNCS 3427*, Dagstuhl, Germany, January 2005.
24. T. Hoßfeld, K. Tutschku, F.-U. Andersen, H. de Meer, and J. Oberender. Simulative performance evaluation of a mobile peer-to-peer file-sharing system. In *NGI2005*, Rome, Italy, April 2005.
25. T. Hoßfeld, K. Tutschku, and D. Schlosser. Influence of the size of swapping entities in mobile p2p file-sharing networks. In *Peer-to-Peer-Systeme und -Anwendungen*, GI/ITG-Workshop in conjunction mit KiVS 2005, Kaiserslautern, Germany, March 2005.
26. A. Birolini. *Qualität und Zuverlässigkeit technischer Systeme, Theorie, Praxis, Management*. Springer, 1991.
27. D. Schlosser, T. Hossfeld, and K. Tutschku. Comparison of robust cooperation strategies for p2p content distribution networks with multiple source download. In *Sixth IEEE International Conference on Peer-to-Peer Computing*, Cambridge, UK, September 2006.
28. R. Jain, D.M. Chiu, and W. Hawe. A Quantitative Measure of Fairness and Discrimination for Resource Allocation in Shared Systems. Technical Report DEC TR- 301, Digital Equipment Corporation, 1984.