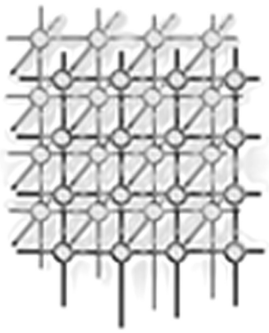
CONCURRENCY AND COMPUTATION: PRACTICE AND EXPERIENCE
Concurrency Computat.: Pract. Exper. 2007; **00**:1–7 *Prepared using* cpeauth.cls *[Version: 2002/09/19 v2.02]*



Performance Analysis and Improvement for BitTorrent-like File Sharing Systems

Ye Tian*,†, Di Wu, and Kam-Wing Ng

Department of Computer Science and Engineering
The Chinese University of Hong Kong, Shatin, N.T., Hong Kong

SUMMARY

In this paper, we present a simple mathematical model for studying the performance of the BitTorrent [5] file sharing system. We are especially interested in the distribution of the peers in different states of the download job progress. With the model we find that the distribution of the download peers follows an asymmetric U-shaped curve under the stable state, due to BitTorrent's unchoking strategies. In addition, we find that the seeds' departure rate and the download peers' abort rate will influence the peer distribution in different ways notably. We also analyze the content availability under the dying process of the BitTorrent file sharing system. We find that the system's stability deteriorates with the decreasing and unevenly distributed online peers, and BitTorrent's built-in "tit-for-tat" unchoking strategy could not help to preserve the integrity of the file among the download peers. We propose an innovative "tit-for-tat" unchoking strategy which enables more peers to finish the download job and prolongs the system's lifetime. By playing our innovative strategy, download peers could cooperate to improve the stability of the system by making a tradeoff between the current downloading rate and the future service availability. Finally, experimental results are presented to validate our analytical results and support our proposals.

KEY WORDS: *Peer-to-peer; BitTorrent; continuous time Markov chain; content availability; incentive mechanism; tit-for-tat*

*Correspondence to: Ye Tian, Department of Computer Science and Engineering, The Chinese University of Hong Kong, Shatin, N.T., Hong Kong
†E-mail: ytian@cse.cuhk.edu.hk

Received
Revised

1. Introduction

Peer-to-peer (P2P) file sharing systems, which allow users to distribute and obtain files in a cooperative manner, have changed the Internet greatly in recent years. Unlike the traditional schemes such as HTTP and FTP, in P2P file sharing systems, the uploading bandwidths of the peers are exploited to speed up the download process, especially when the original server is overloaded.

From the appearance of Napster [1] in 1999, many P2P file sharing applications have evolved, such as Gnutella [2], KaZaA [4] and BitTorrent [5]. Among them, BitTorrent has attracted the largest number of users and has become the main application scheme for P2P file sharing. According to a recent study [27], BitTorrent accounts for 35% of all the traffics on the Internet, which is more than all the other P2P applications combined. Moreover, the file swarming techniques, with BitTorrent as the representative, have been widely adopted in many other applications. For example, Limewire [3], one of the most popular file sharing applications following the Gnutella protocol, began supporting the BitTorrent-like downloading in 2006.

The main feature of BitTorrent, as well as other similar protocols, is that a file is divided into many pieces, so a peer could start to serve the others even if it does not have a complete file. As X. Yang et al. [17] pointed out, the service capacity increases greatly compared with schemes that share the file as a whole.

BitTorrent has been proved to be successful from both the real world deployments and the academic researches. Recent measurements [10, 11, 13] have verified the characteristics of scalability, pollution-resistance and efficiency of the protocol. Theoretical analysis [17, 18, 19, 20] has also provided insights for people to understand the system. However, as shown in [12] and [14], the design is far from perfect, further investigations of the issues not well studied previously are required for understanding and improving the system. In this paper, we focus on the following two issues.

The download efficiency and peer distribution: Unlike the traditional file distribution, where there are roles of servers and clients, peers in BitTorrent are usually uploading and downloading simultaneously. BitTorrent adopts a "tit-for-tat" (TFT) unchoking strategy to prevent free-riding [21]. Under this strategy, a peer must upload as well as download at the same time when interacting with another peer. So, unlike the client-server model, where the download efficiency depends mainly on the server's capacity and the underlying network, in BitTorrent, the probability of successfully finding a proper uploading/downloading partner also determines a peer's downloading rate. Our first goal in this paper is to investigate the system-wide download performance of the peers with different progresses of the download job, and to derive the peers' distribution determined by their download performance.

File availability and incentives to improve the system's stability: The lifetime of a BitTorrent system is the period in which it could provide a complete file from all the online peers. If the content kept by the peers are incomplete, we call the system dead, and the phase that the system is vulnerable to die is called the dying process of the system. During the dying process, the BitTorrent system's stability is influenced by the content on each peer, as well as the peers' behaviors in a complex way. Another related issue is the relationship between the system's stability and its incentive mechanism. A peer in BitTorrent adopts the TFT strategy to select the neighbors which could help it to finish the download job as fast as possible.

However, when the system is in its dying process, another objective besides the downloading rate for a peer to pursue is the possibility of finishing the download job before the system dies. Thus, a good incentive mechanism should be able to encourage the peers to preserve the integrity of the file as well as to download at a reasonable rate.

In this paper, we have developed a simple mathematical model, in which a peer is in one of the N states according to its download job progress, to study BitTorrent. By modeling a peer's behavior based on the state it is in, we have a better insight for the performance of the BitTorrent system under its stable state and the dying process. The other contributions in this paper are listed below:

- We have derived the peer distribution under the stable state with our model. We find that the distribution of the download peers follows an asymmetric U-shaped curve, and could be influenced by various system parameters in different ways notably.
- From the results of the simulation and the real world measurement, we have found that our model is accurate in capturing the peers' download efficiencies and their overall distribution regarding the download job progress.
- We have analyzed the file availability and the dying process of the BitTorrent system. We show that the system is more unstable when the peers are few in number and are unevenly distributed regarding their download progresses.
- We have found that the original TFT unchoking strategy could not be able to preserve the integrity of the file by lowering the probability of the system's death caused by sudden departures of the finished peers. An innovative TFT strategy is proposed to improve the system's stability, in which the future service availability and the current uploading rate is combined when a peer is selecting its partners.
- From the simulation experiments, we have found that our innovative TFT strategy could improve the system's stability extensively at a negligible sacrifice of the download efficiency. We have also observed some unfairness caused by the innovative strategy among the peers which have finished their download jobs.

The remainder of this paper is organized as follows. For Section 2, we give a brief introduction to the BitTorrent protocol; in Section 3, related work on P2P file sharing systems and BitTorrent are discussed; we present a mathematical model and our analytical results under the stable state in Section 4; the content availability and the incentive mechanism for BitTorrent are discussed in Section 5, and we propose our innovative unchoking strategy; finally, in Section 6, experimental results are presented to support our analytical results, testify our motivations, and demonstrate the effectiveness of the innovative unchoking strategy in improving the system's stability.

2. BitTorrent Overview

BitTorrent is a protocol popular for its efficiency in distributing large files. In BitTorrent, a file is divided into many equal sized pieces (typically 256KB) for download, and each piece is further divided into blocks. A peer could download pieces from other peers concurrently while

uploading the pieces it holds to those requesting peers at the same time. In this way, the load is distributed among all the peers in the system [8].

In the BitTorrent architecture, there are three components: the tracker [†], the download peers and the seeds. Tracker is a central server which keeps the global information of the system such as the IP address, port and the number of finished pieces for each peer in the system. When a new peer joins in, it will contact the tracker for a random list of the existing peers in the system to bootstrap. All the online peers should periodically report to the tracker of their job progresses. Those peers which have downloaded the entire file but have not left the system are called seeds. The seeds improve the performance of the system from two aspects: first, they will contribute their bandwidths and upload the pieces to other peers; second, they could insure the integrity of the file since they hold all the pieces of it. We refer to those peers which have not finished the download job as the download peers. In BitTorrent, the download peers are contributing their bandwidths while consuming the bandwidths of others at the same time.

Download peers in BitTorrent use a "local rarest first" (LRF) technique to determine which piece to request. Simply speaking, it will try to download a piece that is least replicated among its neighboring peers. Actually, the LRF strategy is combined with a random choice in the practical deployment. However, for a new peer which has just joined in without any pieces, it will simply download any pieces available to bootstrap itself. Another exception is the "endgame model": when a peer has almost finished its download job, it enters the "endgame model" and requests the last few blocks from everyone.

A peer in BitTorrent usually keeps connections with many other peers. For a connection between two peers, there are two states: choked or not. A peer could choose to "choke" a connection by refusing to upload pieces any more. A download peer in BitTorrent plays a "tit-for-tat" (TFT) strategy to choose which connections to unchoke: it periodically monitors its current connections, and chooses a fixed number (typically four) of the connections with the highest downloading rates to upload. Such a connection is called as a regular unchoked connection. According to the BitTorrent protocol [6], a peer will evaluate its regular unchoked connections and execute the TFT strategy once every 10 seconds. However, the seeds do not play this strategy: for a seed, it will simply choose five download peers with the highest rate to upload, in order to distribute the file as quickly as possible.

Besides the TFT strategy, a download peer also plays a strategy called optimistic unchoking periodically. In optimistic unchoking, a peer randomly chooses a requesting peer to upload regardless whether or not this peer uploads to itself every 30 seconds [6]. Such a connection is referred as an optimistic unchoked connection. Optimistic unchoking has two important meanings for the BitTorrent system: first, it allows a peer to discover better partners for piece exchanging; second, the newcomers without any pieces to upload could get bootstrapped by accepting pieces from the peers playing optimistic unchoking.

[†]Trackerless BitTorrent has been developed recently using DHT techniques

3. Related Work

P2P file sharing systems operate by allowing the peers to form an application level overlay and contribute their contents in a cooperative way. To understand the performance of P2P file sharing systems, many models [15, 16, 17, 18, 19, 20] have been proposed. K. Ramachandran et al. [15] proposed an analytic framework for studying the data traffic of P2P applications with various underlying networks. Z. Ge et al. [16] presented a model with the generality and flexibility of capturing the different architectures of the P2P file sharing systems. For BitTorrent, X. Yang et al. [17] used a branching process for studying the transient regime of the BitTorrent system when the content is first introduced, and used a Markov chain model for understanding the service capacity of the system under the stable state. D. Qiu et al. [18] developed a fluid model for BitTorrent, where the closed-formed solutions for the numbers of the download peers and the seeds could be obtained from the parameters of the peer arrival, departure and the uploading/downloading rates. The analysis also proved that BitTorrent achieves very good scalability. L. Massoulie et al. [19] presented an exhaustive probabilistic model for evaluating the performance issues such as the sojourn time and the population size in file swarming systems (e.g., BitTorrent). Y. Tian et al. [20] studied the system-wide inefficiencies in some phases of the download job progress and discussed on how to prolong the system's lifetime.

There are many measurements [10, 11, 12, 13, 14] based on real world deployments and simulations for BitTorrent. Very good properties such as the efficiency in distributing the file, robustness against pollution and scalability to support a large number of downloaders are observed in these measurements. However, A. Bharambe et al. [14] showed that peers experience the last block problem under a flash crowd, and L. Guo et al. [12] found that there is a fairness issue regarding the peers with heterogeneous bandwidth conditions in BitTorrent.

Enhancements to BitTorrent are proposed in Slurpie [23] and Avalanche [24]. The Slurpie system integrates techniques such as group size estimation, back off and bandwidth estimation to improve the utilization of the link bandwidth and to decrease the burden of the topology server. For Avalanche, the pieces of the file are encoded both on the source and on the nodes, which makes the propagation of the file more efficient than transmitting the uncoded pieces. The system is also more robust under extreme situations with the sudden departures of nodes.

The availability problem for P2P file sharing systems is studied from two aspects: host availability and content availability. R. Bhagwan et al. [25] measured and analyzed the availability of the hosts on the Overnet [7][‡] file sharing system, in which a file is usually served by a single peer. For the content availability, F. M. Cuenca-Acuna et al. [26] proposed an algorithm to improve the availability of the files on an unstructured P2P replication platform. Such an platform is featured with the nodes' static availabilities. However, for BitTorrent, the availability problem is a little different compared with the other systems; as in BitTorrent, it is not just a single peer but the entire BitTorrent community is involved in serving a particular file, and all the peers will eventually leave the system. Currently, most of the works

[‡]Overnet/eDonkey was taken down by RIAA in Sep. 2006

on BitTorrent's availability are experimental, such as the measurements discussed in [10] and [11].

Our model for BitTorrent is different from the ones in previous works. Concretely, unlike the models in [17] and [18], in which the download peers are considered as a whole, and are assumed to behave uniformly; in our analysis, we model the behaviors of the peers differently according to the states they are in, thus enabling a deeper understanding and more accurate evaluation of the system.

4. Modeling and Analysis for Peer Distribution

Contents are published via BitTorrent with a metadata file (with *.torrent* as the extension name). When a file is first introduced, there will be a burst of download requests, but only a few (usually one) seeds are available for uploading the pieces. After the initial burst phase, the BitTorrent system would reach its stable state, where the arrival of the peers is not in a burst, but could be viewed as a Poisson process [17]. In this section, we present a mathematical model for BitTorrent and analyze the performance of the system under its stable state. From the analytical results, we find that the peer distribution follows an asymmetric U-shaped curve, and the parameters such as the departure rate of the seeds and abort rate of the download peers could influence the distribution in different ways notably.

4.1. The Model

Before presenting the analytical model, first we introduce some necessary definitions. As described in Section 2, in BitTorrent, unchoked connections could be categorized as regular unchoked connections and optimistic unchoked connections. However, this categorization can not reflect the reciprocation status of the connections. For example, a peer may or may not be able to download with high rate through an optimistic unchoked connection. In addition to the regular/optimistic unchoked connections, we introduce the concepts of the reciprocal/altruistic connections: a reciprocal connection is defined as an unchoked connection through which a peer could upload and download with high rates; and an altruistic connection is defined as an unchoked connection through which a peer could only upload with high rate, but downloads poorly. Note that with these definitions, a regular unchoked connection is always a reciprocal connection, and an optimistic unchoked connection may be either a reciprocal connection or an altruistic connection, depending on the performance of the peer on the other end of this connection.

Our model for analyzing the BitTorrent system is illustrated in Figure 1. In this model, the states of S_0, S_1, ..., S_{N-1} represent the peers with $[\frac{0}{N}, \frac{1}{N})$,$[\frac{1}{N}, \frac{2}{N})$,...,$[\frac{N-1}{N}, 1)$ portions of the file respectively, and S_N is the state for the seeds. We choose the value of N as a constant integer big enough to embody the detailed difference between the peers with different piece amounts. The upper bound of N is M, which is the total number of pieces for the file.

When a new peer first enters into the BitTorrent system, it will start as a download peer with $\frac{0}{N}$ portions of the file at the state S_0. After some peers in the system have uploaded pieces to this peer, it will be a peer with $\frac{1}{N}$ portions of the file at state S_1, and meanwhile it could

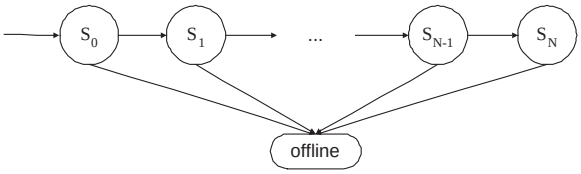


Figure 1. The model for the BitTorrent system

Table I. Parameters in the BitTorrent model

Parameter	Meaning
N	total number of the states in the model
M	total number of the pieces for the file
$x_i(t)$	number of the download peers in the state of S_i, $i = 0, .., N-1$
$y(t)$	number of the seeds
μ	average uploading rate for a peer, including the seeds
λ	arrival rate of the new download peers
γ	departure rate of the seeds
θ	abort rate of the download peers
$\eta_{i,j}$	relative efficiency of obtaining pieces between peers in the state of S_i and peers in the state of S_j through the reciprocal connections, $i, j = 0, ..., N-1$
$\eta'_{i,j}$	relative efficiency of obtaining pieces by peers in the state of S_i through the altruistic connections from peers in the state of S_j , $i, j = 0, ..., N-1$
m	number of pieces in the local rarest first set
ρ	portion of the uploading bandwidth allocated in the reciprocal connections of a download peer compared with its total uploading bandwidth

upload the pieces it holds to other download peers. The uploading and downloading processes continue until the peer has obtained all the pieces and becomes a seed at state S_N; however, it may also abort the system and goes to the $offline$ state during the download process before finishing the download job. Finally, a seed will depart the system after it has served in the system for a certain period of time. The process is demonstrated by our model in Figure 1.

We use $x_0(t)$, $x_1(t)$, $x_2(t)$, ... , $x_{N-1}(t)$ to denote the number of the download peers in states S_0, S_1, ..., S_{N-1} in the BitTorrent system at time t respectively; for the seeds, we use $y(t)$ for the population at time t. We are interested in the number of peers as $x_0(t)$, $x_1(t)$, $x_2(t)$, ... , $x_{N-1}(t)$ and $y(t)$ under the stable state of the system. However, before the discussion, we must introduce the notations and parameters listed in Table I which will be used in our model.

For the peers in the states of S_0, S_1, ..., S_{N-1}, S_N and $offline$, we model the arrival of the new download peers as a Poisson process with a rate of λ, and the seeds depart the system according to an exponential distribution with a rate of γ. The download peers in the states of S_0, S_1, ..., S_{N-1} are assumed to abort the system according to an exponential distribution with

a rate of θ. For a download peer, it can obtain pieces of the file from three sources: first, it could exchange the pieces with other download peers through the reciprocal connections; second, it could download the pieces from other download peers through their altruistic connections; finally, it could download from the seeds as the third source. However, the efficiencies of obtaining the pieces from the three sources are different. Suppose that the download peers could download from the seeds with an efficiency of one, we use $\eta_{i,j}$ $(0 < \eta_{i,j} < 1)$ for the relative piece obtaining efficiency of the peers in state S_i through the reciprocal connections with the peers in state S_j; and we use $\eta'_{i,j}$ $(0 < \eta'_{i,j} < 1)$ for the relative piece obtaining efficiency of the peers in state S_i though the altruistic connections from the peers in state S_j. By applying the continuous time Markov chain [22], we derive the transfer rates between two neighboring states in Figure 1 as follows:

$$\begin{cases} r(S_N, offline) &= \gamma y(t) \\ r(S_i, S_{i+1}) &= \mu N y(t) \cdot \frac{x_i(t)}{\sum_{j=0}^{N-1} x_j(t)} + \rho \cdot \frac{x_i(t)}{\sum_{j=0}^{N-1} x_j(t)} \cdot \mu N \sum_{k=0}^{N-1} x_k(t) \cdot \eta_{i,k} \\ & \quad +(1-\rho) \cdot \frac{x_i(t)}{\sum_{j=0}^{N-1} x_j(t)} \cdot \mu N \sum_{k=0}^{N-1} x_k(t) \cdot \eta'_{i,k} \\ r(S_0) &= \lambda \\ r(S_i, offline) &= \theta x_i(t) \end{cases} \quad (1)$$

for $i = 0, 1, ..., N-1$. Here we use $r(state1, state2)$ to denote the transfer rate of the peers from $state1$ to $state2$. $r(S_0)$ denotes the entry rate of the new peers into the system. Note that for a download peer transferring from two neighboring states, we sum up the contributions from the three sources. We use ρ for the portion of the uploading bandwidth allocated in the reciprocal connections of a download peer divided by its total uploading bandwidth, and the parameter m is the number of the pieces a download peer will choose to request simultaneously according to the LRF strategy.

We first focus on the parameters of $\eta_{i,j}$ and $\eta'_{i,j}$, which are defined as the relative efficiencies of peers in state S_i to obtain pieces from peers in state S_j through reciprocal connections and altruistic connections respectively, given that the efficiency of obtaining pieces from the seeds is one. As once a reciprocal connection is setup, like the seeds, the two peers of the connection will use all the available bandwidth for uploading, then the relative piece obtaining efficiency for this reciprocal connection is one. For $\eta_{i,j}$, taking all the reciprocal connections which could be potentially setup between the peers in state S_i and state S_j into consideration, we could further interpret $\eta_{i,j}$ as the probability that a random peer in state S_i could successfully setup a reciprocal connection with a random peer in state S_j. Similarly, $\eta'_{i,j}$ could be interpreted as the probability that a random peer in state S_i could successfully have an altruistic connection from a random peer in state S_j, as once an altruistic connection is setup, its relative piece obtaining efficiency is one.

To obtain the expressions of the two probabilities, we assume that under the stable state, all the pieces are evenly distributed among the peers, as is indicated in [13]. With this assumption, we believe that the piece obtaining efficiencies are only related to the unchoking strategies of the peers statistically. Consider a peer p_i in state S_i, on average p_i has $\frac{i+0.5}{N}$ portions of the file, and is looking for the $\frac{N-i-0.5}{N}$ missing portions from its neighbors. For another peer p_j in state S_j, the probability that p_i can download an arbitrary piece from p_j

is $\frac{j+0.5}{N} \cdot \frac{N-i-0.5}{N} = \frac{(N-i-0.5)(j+0.5)}{N^2}$, and since p_i will request m pieces simultaneously, the probability that p_j could upload at least one piece to p_i is $1-(1-\frac{(N-i-0.5)(j+0.5)}{N^2})^m$; similarly, the probability that p_i could upload at least one piece to p_j is $1-(1-\frac{(N-j-0.5)(i+0.5)}{N^2})^m$. As the two peers in a reciprocal connection must upload and download simultaneously, we can see that $\eta_{i,j}$, now interpreted as the probability of successfully setting up a reciprocal connection between S_i and S_j, could be expressed as

$$\eta_{i,j} = (1-(1-\frac{(N-i-0.5)(j+0.5)}{N^2})^m) \cdot (1-(1-\frac{(N-j-0.5)(i+0.5)}{N^2})^m) \quad (2)$$

Note that $\eta_{i,j} = \eta_{j,i}$, as it is the relative piece obtaining efficiency though reciprocal connections between two states. For $\eta'_{i,j}$, since one peer is uploading altruistically, we have

$$\eta'_{i,j} = (1-(1-\frac{(N-i-0.5)(j+0.5)}{N^2})^m) \quad (3)$$

which now is interpreted as the probability that p_j could successfully setup an altruistic connection to p_i. For the value of m, as a download peer usually requests approximately 4 or 5 pieces simultaneously in BitTorrent, we set $m = 4$ in our study.

We now estimate the value of ρ, which is defined as the ratio of the uploading bandwidth allocated in the reciprocal connections of a download peer compared with its total uploading bandwidth. In BitTorrent, a peer will maintain four regular unchoked connections and try one optimistic unchoked connection every 30 seconds. Note that for an optimistic unchoked connection, it could be setup as long as it is altruistic, then it could either stay altruistic or become reciprocal, depending on whether or not the peer on the other end of the connection uploads actively. Suppose in a particular 30 second period, the peer's optimistic unchoked connection stays altruistic, then it will compete the peer's uploading bandwidth with the other four regular unchoked connections, and will take $\frac{1}{5}$ of the bandwidth on average; but if it becomes reciprocal, all the peer's uploading bandwidth is allocated in reciprocal connections. Obviously, if we know the probability that an optimistic unchoked connection becomes reciprocal, then we could estimate the value of ρ. From the previous analysis, we know that the probability of successfully setting up an altruistic connection from peers in state S_j to S_i is $\eta'_{i,j}$, while the probability of successfully setting up a reciprocal connection is $\eta_{i,j}$, so the probability that an optimistic unchoked connection becomes reciprocal could be expressed as $\frac{\eta_{i,j}}{\eta'_{i,j}} = \eta'_{j,i}$, as an optimistic unchoked connection must be altruistic initially. If we let $m = 4$, then $\eta'_{j,i} \approx 0.684$ by choosing i, j as $N/2$. Finally, we could estimate the value of of ρ as $\rho \approx 1 \times 0.684 + \frac{4}{5} \times (1-0.684) \approx 0.94$. From the estimation, we can see that the download peers in BitTorrent allocate a majority of their uploading capacity in the reciprocal connections. Although this is very effective against free-riding, we will see later that some performance loss will be caused by this bandwidth allocation scheme.

4.2. Analytical Results and Discussions

We first study the situation when the download peers will never abort the system while all the seeds will depart the system immediately after they have finished their download job. In this

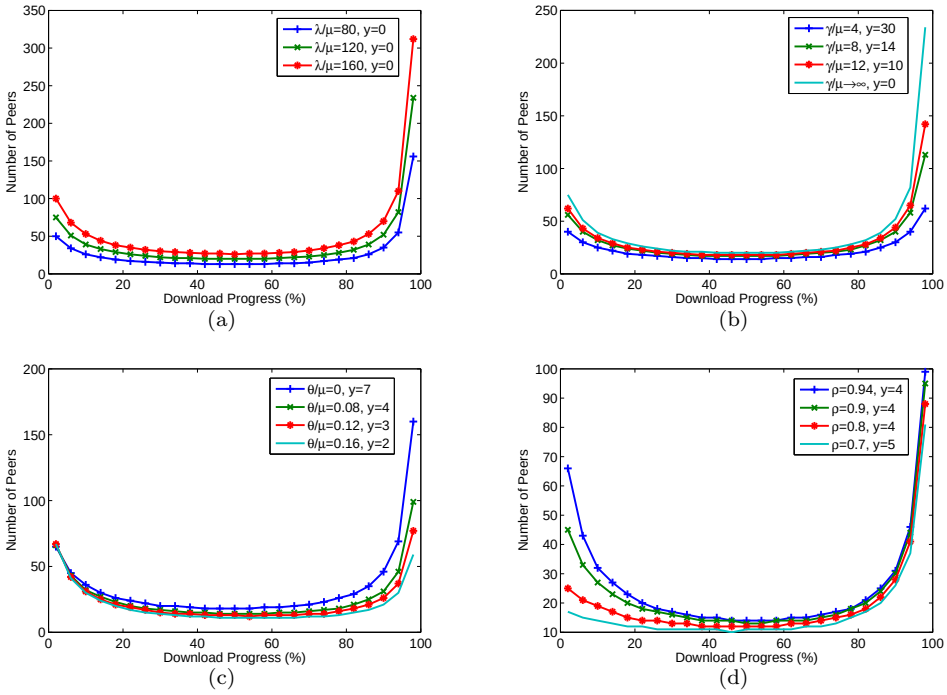


Figure 2. Peer distribution under the stable state, with (a) varying offered loads; (b) varying departure rates; (c) varying abort rates; and (d) varying reciprocal connection bandwidth portions.

case, we have $\gamma \to \infty$ and $\theta = 0$. Under the stable state of the model in Figure 1, applying the principle that "*the rate of the flow out of a state = the rate of the flow into this state*"[22], we can numerically solve the model in Equation (1) combined with Equation (2) and Equation (3).

Figure 2(a) plots the number and the distribution of the peers with different download progresses under the offered load $\frac{\lambda}{\mu}$ as 80, 120 and 160 respectively. We set the number of the states N as 25. From the analytical results, we find that the peer distribution follows an asymmetric U-shaped curve, in which the peers are more blocked in the states with very small or large portions of the file. This could be explained by the relatively lower reciprocal connection piece obtaining efficiencies of the peers in these states than in other states. We also find that there are more peers blocked in the states near the end of the download progress than in the states at the beginning of the job, making the U-shaped curve asymmetric. We explain this with the altruistic connections: although peers at both the beginning and the end stages of their download jobs have difficulties in obtaining the pieces from the reciprocal connections, the beginners could benefit from the altruistic connections much easier than the peers seeking

the last few pieces at the end of the download job. Note that our analytical results here conform to the fact that the "last block problem" is widely observed while the corresponding "first piece problem" is rarely mentioned. Finally, we find that by altering the parameters of λ and μ, we could only change the number of the peers at each state, however, the shape of the curve, which presents the relationships among the populations of the peers with different download job progresses, will not get changed. The three curves with the increasing offered load $\frac{\lambda}{\mu}$ also demonstrate the linear relationship between the total number of the peers and the offered load, as is observed in [17].

Previously we have assumed that $\gamma \to \infty$ and $\theta = 0$. However, in real world applications, it is observed that many peers may stay in the system after they have finished their download job and serve as the seeds for a certain period of time [10], and a peer may also abort the system before it has finished the job for some reasons (e.g. loss of interest in the contents). Based on these considerations, we will discuss what will happen to the numbers and the distribution of the download peers, if more practical settings of the parameters γ and θ are introduced.

First we study the influence of the departure rate $\frac{\gamma}{\mu}$. We let the offered load $\frac{\lambda}{\mu} = 120$ and the abort rate $\theta = 0$, and vary the departure rate $\frac{\gamma}{\mu}$ from 4, 8 to 12 in our model. By solving the model numerically, we obtain three curves of the peer distributions shown in Figure 2(b). We also plot the peer distribution when $\gamma \to \infty$ ($\frac{\gamma}{\mu} \to \infty$) for comparison. By observing the analytical results, we have two findings: First, with the decrease of the departure rate, the number of the seeds increases, however, the number of the download peers decreases. This is because the smaller departure rate allows more seeds to stay in the system, and these seeds inversely help the download peers to finish the job and leave the system more easily. Second, the introduction of the departure rate will change the distribution of the download peers. When the departure rate decreases, the curves are getting more and more "flat". This is because the peers could download from the seeds more efficiently with more seeds staying in the system, and they will transfer from state to state more quickly in the model of Figure 1.

We also study the case when $\theta \neq 0$, which means the download peers will abort the system. We set the offered load $\frac{\lambda}{\mu} = 120$ and the departure rate $\frac{\gamma}{\mu} = 16$, but vary the abort rate $\frac{\theta}{\mu}$ from 0 to 0.16. The results numerically derived from the model are presented in Figure 2(c). From the figure, it is observed that the increase of the abort rate $\frac{\theta}{\mu}$ will not only decrease the total number of the download peers and the seeds, but will also influence the peer distribution. We could see that with the increase of the abort rate $\frac{\theta}{\mu}$, the peers are getting more concentrated in the states with the smaller percentages of the file.

Previously we have estimated that the value of the parameter ρ, which is the portion of the download peers' bandwidth allocated in the reciprocal connections, is approximately 0.94. For our last numerical study, we wish to know what will happen if the download peers allocate more bandwidth in their altruistic connections by setting up more optimistic unchoked connections. For simplicity, here we just vary the value of ρ from 0.7 to 0.94, and fix the parameters $\frac{\lambda}{\mu} = 120$, $\frac{\gamma}{\mu} = 16$, and $\frac{\theta}{\mu} = 0.08$. The curves of the peer distributions are plotted in Figure 2(d). From the figure, we find that although the performance of the peers near the end of the download job does not get improved significantly, the inefficiencies at the beginning are well alleviated when more bandwidth is allocated in the altruistic connections. The observation indicates that if we allow the download peers to behave more altruistically by increasing the number of

the optimistic unchoked connections, the system's performance will get improved. However, fairness is threatened in this case, as peers may free-ride on the altruistic connections with a considerable bandwidth. In the current BitTorrent protocol, fairness is considered with higher priority than efficiency, so, the altruistic connections are allocated with a very small portion of the bandwidth, and only serve for discovering new partners instead of improving the system's download efficiency.

In summary, we have studied the population and the distribution of the peers at their different download job progresses by numerically solving the model. We find that under the ideal case that when $\gamma \to \infty$ and $\theta = 0$, the peer distribution follows an asymmetric U-shaped curve, while with the consideration of $\gamma \nrightarrow \infty$and $\theta \neq 0$, similar results could be obtained but the parameters of γ and θ influence the peer distribution in different ways notably. Moreover, we believe that the peer distribution curves in Figure 2(c) reflect the distribution in the real world, with the following features: 1) the distribution follows a U-shaped curve; 2) the distribution is asymmetric, with more peers blocking near the end of the download progress than at the beginning; 3) except for the states at the beginning and at the end of the download job, there is a slight decreasing of the peers' population from the states with smaller portions of the file to the states with larger portions. We then discuss the consequent peer distribution if the peers behave more altruistically with more optimistic unchoked connections, and show that there lays a tradeoff between the fairness and the efficiency in designing the BitTorrent's unchoking strategies. We will verify our model by simulations and real world measurements in Section 6.

5. Content Availability and Incentive Mechanism

In this section, we discuss the relationship between the content availability of the BitTorrent system and its incentive mechanism. We first study the content availability in BitTorrent, then we define and analyze the dying process of a BitTorrent system, finally the innovative TFT strategy aiming to preserve the integrity of the file, increase the stability and prolong the lifetime of the BitTorrent system is proposed.

5.1. The Availability

The availability problem arises because of the decentralized nature of the P2P file sharing systems, especially when the peers are organized in an unstructured overlay. Usually, people correlate the host uptime with the availability of the content, such as the measurement in [11] and [26]. However, in BitTorrent, since all the peers download a single file cooperatively, and there are no roles of senders and receivers, the content availability is more associated with the number and the distribution of the peers in the system. As mentioned in Section 2, the existence of the seeds could ensure the availability of a complete file since they hold all the pieces; however, with the absence of the seeds, it is not necessary that the file is complete since each peer may keep only a part of it.

We study the piece availability instead of the file availability for a BitTorrent system because it is independent of the file size. We assume that in a BitTorrent system, the pieces are

evenly distributed among all the online peers. This is a reasonable assumption because all the download peers play the LRF strategy, and its soundness is proved in [13]. Suppose there are S peers online (including the seeds) as $\{p_1, p_2, ..., p_S\}$, and at time t, peer p_i holds m_i pieces, then the probability that a particular piece is not held by p_i is $(1 - \frac{m_i}{M})$, where M is the total number of the pieces for the file. For this particular piece, we could calculate its availability, which is the probability that it is held by at least one online peer, as:

$$Av(t) = 1 - \prod_{i=1}^{S}(1 - \frac{m_i}{M}) \approx 1 - (1-1)^{y(t)} \prod_{i=0}^{N-1}(1 - \frac{i+0.5}{N})^{x_i(t)} \quad (4)$$

The last expression in Equation (4) is derived from the model in Figure 1. We approximate the probability of not holding the particular piece by a peer in state S_i as $(1 - \frac{i+0.5}{N})$, the probability that the piece is not held by all the peers in state S_i is the $x_i(t)$th power of $(1 - \frac{i+0.5}{N})$, here $x_i(t)$ is the number of the peers in state S_i. For the seeds in the state S_N, the probability of not keeping the piece is $(1-1)^{y(t)}$, which is 0 if there is at least one seed, and 1 otherwise. Note that with this expression, the piece availability is independent of M, the file's piece number. For a file with M pieces, its file availability could be expressed as $Av(t)^M$.

In Equation (4), if there are seeds, we would have $Av(t) = 1$. Furthermore, the peers with a large part of the file contribute greatly to the piece availability: consider a system composed of 20 peers with each keeping 10% of the pieces, the piece availability calculated with Equation (4) is 0.88; however, a system with only 5 peers keeping 10% of the pieces and 2 peers keeping 90% of the pieces has a piece availability of 0.99.

5.2. Analysis of the Dying Process

Each BitTorrent system has a lifetime. When the content is first published on the Internet, a seed is usually available for uploading for a certain period of time. However, with the departure of the original seed and a decrease in the arrival of new peers arriving, the file may become incomplete among all the online peers at a certain moment, and the peers staying in and arriving later may not be able to obtain a complete file, unless a seed rejoins the system. We call a system dead if the pieces of the file are not complete among all the online peers. The period between the file's initial introduction and the time when it becomes incomplete is referred to as the lifetime of a BitTorrent system; and we call the phase during which the system is vulnerable to die as its dying process. Actually, the lifetime of a BitTorrent system may vary from a few days to several months, depending on the type and the popularity of the content.

We apply the model presented in Figure 1 to study the dying process of the system. However, unlike the analysis under the stable state, where we model peers' download efficiencies differently for different states; in the dying process, as the size of the community is very small, factors such as the individual peer's uploading/downloading bandwidths and the congestions of the underlying network will influence the peer's download efficiency more significantly than the unchoking strategies. For simplicity, here we assume $\rho\eta_{i,j} + (1-\rho)\eta'_{i,j} = \eta$ as a constant for each peer in the system. We also assume $\gamma \to \infty$ and $\theta = 0$ in our analysis.

First we consider the situation that peers are evenly distributed on the entire download progress by modeling a smooth arrival of the new peers. If the arrival rate is λ, the smooth arrival means that the time interval between two consecutive arrivals is $\frac{1}{\lambda}$. From the analysis of [18], for a single peer, the time required to finish the download job is $\frac{1}{\eta\mu}$. So, for the period of completing a download job, there are $\frac{\lambda}{\eta\mu}$ newly arrived peers. $\frac{\lambda}{\eta\mu}$ could also be interpreted as the number of the online peers in the system when a peer is leaving. Obviously if λ decreases, the piece availability calculated with Equation (4) will also decrease as the number of peers staying in the system is rarer; and for a certain availability threshold, there exists a minimum λ, which also determines a minimum number of the online peers $\frac{\lambda}{\eta\mu}$ ensuring the system to have a piece availability higher than the threshold. We use the "minimum number of online peers" as an important metrics in our analysis. If we set the availability threshold as 0.99998 (five nines), by applying the last expression of Equation (4), we find that in order to keep a piece availability above this threshold, the minimum number of online peers should be 11.11, which means that if λ decreases such that $\frac{\lambda}{\eta\mu}$ is smaller than 11.11, the system would die. Obviously, this is a very good result: only 12 peers online will ensure the integrity of the file with high probability.

However, assuming an even peer distribution with smooth arrival is not a good approximation for the peers' behaviors in the real world, as peers arrive and download independently. Actually, some peers may be close to each other than to other peers regarding their download progresses, and some clusters may be formed. This phenomenon is more obvious in a BitTorrent system during its dying process, as there are fewer peers. For simplicity we discuss the extreme case that peers are grouped in clusters with exactly the same download progresses. As we have assumed that the groups of peers have the same download efficiency, to enable the clustering, we model the arrival of the new peers in a $k-cluster$ fashion, in which $k > 1$ peers arrive simultaneously each time. For example, if $k = 2$, an arrival rate of λ means two peers arrive in the interval of $\frac{2}{\lambda}$, instead of one peer arriving in the interval of $\frac{1}{\lambda}$. Applying the $k-cluster$ arrival pattern to the last expression of Equation (4), we find that the required minimum number of online peers increases with k. Given the availability threshold as 0.99998, the results are plotted in Figure 3.

From the results in Figure 3, we find that the system is more prone to die when peers are in clusters with larger cluster size k. For example, if the peers arrive in a cluster of 6, on average, the system should manage to keep nearly 21 peers in the system all the time in order to keep alive.

5.3. Incentive Mechanism for the Dying Process

In BitTorrent, peers are playing the TFT strategy during the selection of their uploading/downloading partners: by default, a peer will choose to upload to four peers from which it could download at the highest rates. A game theoretical analysis in [18] points out that under the TFT strategy, the peers under the stable state will reach a Nash equilibrium: each peer will upload at its highest rate to maximize its utility, which is defined as the peer's current overall downloading rate.

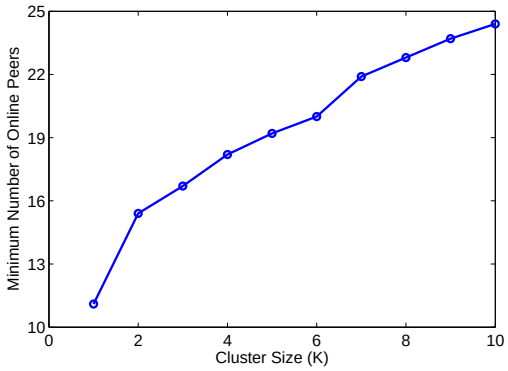


Figure 3. The relationship between the required minimum number of online peers and the cluster size in the $k-cluster$ arrival pattern

However, when the system enters into its dying process, both the social welfare and individual peer's benefit get changed. First, for the social welfare, the system should try to ensure the availability of the file by prolonging the system's lifetime, while providing the download peers a reasonable downloading rate as high as possible. Second, for the individual peer, it will try to download a complete file as quickly as possible. Obviously, preserving the integrity of the file should be considered with higher priority than the downloading rate in the dying process, since an incomplete file means nothing for most types of the contents.

From the studies on the piece availability and the dying process of the system, we find that the death of a BitTorrent system is usually caused by the departure of a peer which has finished its job and leaving the system with only the peers keeping smaller portions of the file. Unfortunately, the built-in TFT unchoking strategy cannot prevent those peers which have great importance to the stability of the system from finishing their job and departing the system. Consider a scenario of a small BitTorrent system demonstrated in Figure 4(a). In this system, assume the file is divided into 10 equal sized pieces, indexed as 0, 1, 2,..., 9. There are 10 peers in the network, each keeping different pieces, and they form an overlay as shown in the figure. We assume that all peers have the same uploading and downloading rates, and note that the TFT strategy degenerates to a random choice under this assumption. When the game starts, obviously peer A will request to peer J for the piece 9, since it keeps the only replica among all the other peers in the system. Assume that peer J responds to the request of peer A immediately and asks a piece from peer A in return, it will request a piece from the set of $\{0, 1, 2, 3, 4, 5, 7\}$ with equal chance, since they are the local rarest pieces for it. Suppose peer A will depart the system immediately after it has finished exchanging the pieces with peer J successfully. The probability that the file is complete among the peers B-J after peer A's departure is $1/7$, since the only chance that the file is complete is that peer J had requested piece 0 from peer A, which happens at a probability of $1/7$. However, if peer J refuses to

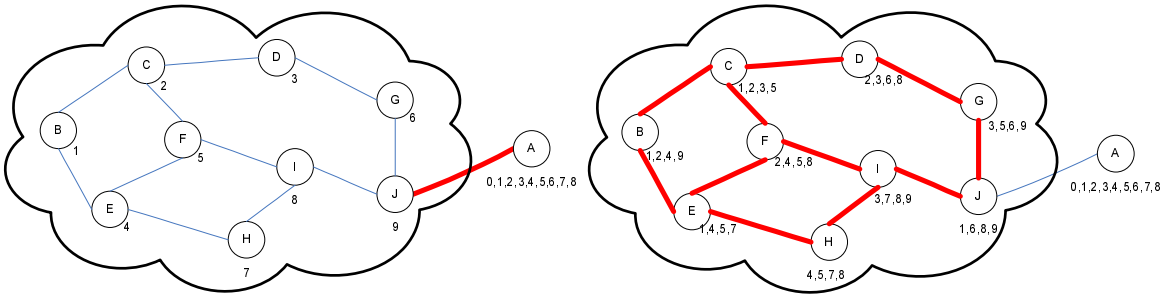


Figure 4. The demonstration of the peer unchoking strategy and the system's stability

respond to A and exchanges the pieces with other peers first, it is likely that the system will survive with a higher probability. For example, when the peers B-J exchange pieces so that each keeps 4 pieces, as demonstrated in Figure 4(b) , when a peer from B-J with the piece 9 starts to respond to A's request, the probability of a complete file after peer A's departure will surely increase. Suppose it is also J that exchanges the pieces with A, the probability now becomes $1/3$, since peer J will only request a piece in the set of $\{0, 2, 4\}$.

Motivated by this observation, we find that to preserve the integrity of the file, it is important for the system to keep the seeds or the peers with large portions of the file online. Since we cannot prevent the seeds from departing the system, the only choice is to keep the peers with large portions of the file online. However, simply blocking the peers from exchanging the pieces is not a good idea, since the peers are selfish, it will pursue to maximize its utility, which is the overall downloading rate under the BitTorrent's current TFT strategy. Thus, the new mechanism must be incentive-compatible, where the individual peer's utility as well as the system's social welfare are benefited. Furthermore, we believe that the algorithm should be fully distributed without burdening the trackers, as such a centralized component is easily overloaded. Another issue that should be noticed is the tradeoff between the file availability and the system's efficiency. Since some peers will get into difficulty when trying to finish the download job, the system's overall efficiency will decrease by this reason. However, we will show that this sacrifice could be negligible in the simulation experiment in Section 6.

Our innovative incentive mechanism is also a "tit-for-tat" strategy executed periodically, in which a peer will only upload to a partner it is downloading from. However, we modify the unchoking strategy in neighbor selection with the consideration of the future service availability. Consider a peer p_i with m_i pieces and a uploading rate of μ_i, and p_j is its neighbor which has m_j pieces and is at a uploading rate of μ_j and a downloading rate of d_j. In the BitTorrent's TFT strategy, p_i will choose p_j to upload if μ_j is one of the four highest uploading rates among all of p_i's neighbors. However, for our innovative TFT strategy, we calculate two times: $T_1 = \frac{m_j \frac{M-m_i}{M}}{\mu_j}$ and $T_2 = \frac{M-m_j}{d_j} + \Delta T$. For T_1, it is the estimated time that p_j could finish uploading all the pieces p_i needs at the current uploading rate, note that $m_j \frac{M-m_i}{M}$ is the estimation of the number of the pieces which p_i may request from p_j. And for T_2, it is the

estimated time that p_j will be online, where $\frac{M-m_j}{d_j}$ is the estimated job finishing time and ΔT is the average service time of the seeds. For the neighbor peer p_j, p_i will calculate its potential gain as:

$$g_j = \mu_j \sum_{i=0}^{\lceil \frac{T}{t_{int}} \rceil} \alpha^i$$

where $T = \min(T_1, T_2)$ is the estimated service time from p_j, t_{int} is the execution interval of the unchoking strategy, and α could be interpreted as a tradeoff factor between the file availability and the download efficiency, with a value in the range of $[0, 1)$. In our innovative TFT strategy, peer p_i will choose p_j to upload when the gain g_j is among the highest four gains estimated among its neighbors. Note that if $\alpha = 0$, we only compare the uploading rate μ_j, then it is BitTorrent's original unchoking strategy; and for $\alpha \to 1$, it means p_i will consider p_j's future uploading service with equal importance as the uploading rate currently obtained.

If we take a closer look at the estimated service time $T = \min(T_1, T_2)$, from the definition, we could find that T_1 will increase linearly with p_j's download progress as $\frac{m_j}{M}$; and T_2 will decrease linearly with it. If we assume $d_j = 5\mu_i$ for all the neighboring peers, and let $\Delta T = 0$, the optimal choice for p_i to upload is a peer with $m_j = \frac{M^2}{6M-5m_i}$ pieces, which is a value very close to m_i when m_i is near M. In other words, when applying the innovative TFT strategy, peers will be more likely to upload to a peer which has a similar download job progress, and leave those peers with larger portion of the file for later use. Given that all the peers are playing the innovative TFT strategy in selecting their neighbors, the global effect is that the peers will form clusters according to their job progresses and cooperation is more likely to happen between the peers in the same cluster. If we revisit the extreme scenario demonstrated in Figure 4(a), under the innovative TFT strategy, it is definite that the peers B-J will keep peer A waiting while they cooperate mutually among themselves first.

Finally, for the deployment of the innovative TFT strategy, since it is proposed aiming to improve the content availability for the BitTorrent system in the dying process, peers will not play it until the system is prone to die. The tracker could be used for the switching of the strategies: when the tracker detects that the piece availability is lower than some threshold, it could notify the online peers to switch the unchoking strategy from the original TFT to our innovative one.

6. Experimental Results

In this section, four sets of experiments are presented, our objectives are to validate the analytical results, and support the innovative TFT strategy, as discussed in Section 4 and Section 5.

6.1. Experiment 1: Stable State Simulation

In this experiment, we study the results from a discrete-event simulation of a BitTorrent-like system. In the simulated system, we set the file size as $100M$, and let the piece size to be the

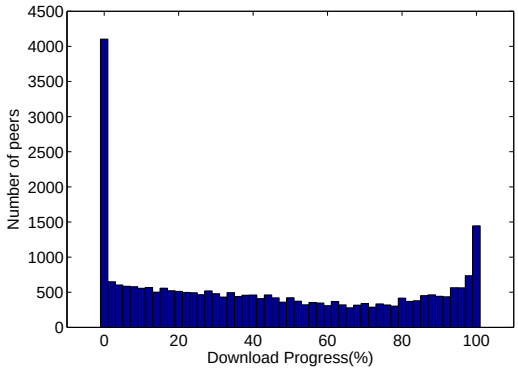


Figure 5. Aggregated peer distribution from the simulation

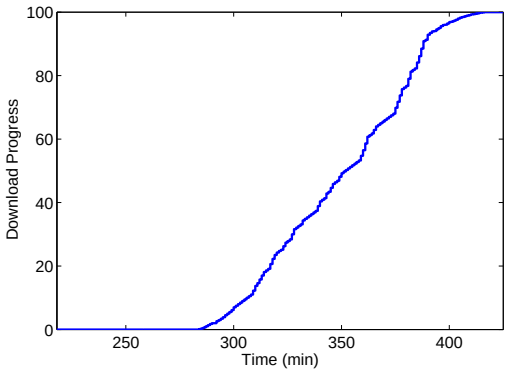


Figure 6. The trace of the download progress for a single peer

default, which is $256K$. For the downloading rate, since we are studying the distribution of the peers essentially influenced by the reciprocal connection piece obtaining efficiency $\eta_{i,j}$, we assume a homogenous case where each peer is allowed to obtain one piece from an unchoked connection per simulation minute. The number of the maximum unchoked connections is set as four. For simplicity, we do not implement the optimistic unchoking strategy in our simulator and enable each peer with a global view of the entire system. In our simulation, the new peer's arrival rate is $0.5/\min$ and the seeds depart the system at a rate of $0.007/\min$, the abort rate of the unfinished peers is set as $0.001/\min$. For this setting, each seed stays in the system

for $\frac{1}{0.007} \approx 143$ minutes on average, which is longer than a peer's average download time, approximated as $\frac{100M}{256K \times 4} = 100$ minutes, as observed in [9].

To bootstrap the system, ten seeds are inserted into the system at the beginning. We observed that the system reached its stable state after 200 minutes. However, due to the randomness of the peers' behaviors, the peer distribution appears more in disorder than following some distribution, thus we aggregated the peer distributions from the 200^{th} minute to the 473^{rd} minute, and obtained an integrated peer distribution, as shown in Figure 5. We could see that on average, the peer distribution is U-shaped, which indicates that our model is a good approximation for a BitTorrent system under the stable state. Note that since there is no optimistic unchoking in the simulation, unlike the curves in Figure 2, there are more peers at the beginning of the download progress than at the end of it. We also find a linear decrease of the peer population from 10% to 80% of the job progress, which we believe is caused by the aborts of the download peers, as demonstrated in our analytical results in Figure 2(c). The abnormally huge number of peers near 0% of the job progress is caused by the strict TFT strategy in our simulation, in which a new peer could only get its first few pieces from the seeds due to the absence of the optimistic unchoking.

In Figure 6, we traced a single peer from its joining to its departure as a seed, and recorded its download progress every minute from the 205^{th} minute to the 425^{th} minute. From the figure, we can see that the peer waits 66 minutes before it begins to download because of the strict TFT strategy, and it stays in the system as a seed for 20 minutes before the departure. More interestingly, it is observed that the download efficiency of the peer is almost constant except for two periods before the 300^{th} minute and after the 380^{th} minute, which are at the beginning and at the end of the download respectively. In these two phases, the peer's download efficiency reduces greatly. Recall that in our previous analysis, a peer's download efficiency at its different states of the job progress is largely determined by its efficiency in exchanging pieces with other peers, the download progress shown in the figure presents the peer's evolution in its download efficiency and supports our model.

6.2. Experiment 2: Stable State Measurement

In this experiment, we present the measurement of the peer distribution with different download job progresses based on the data obtained from the real world BitTorrent application.

Generally, there are two possible approaches to gather the real-world BitTorrent data: one is the server-based approach by setting up a tracker and introducing a file on the Internet; and the other client-based method is to connect to the tracker of a file shared on the Internet. Since we are studying the behavior of a BitTorrent system with many peers under the stable state, we require that the system should have a large number of the download peers and the system should have a long lifetime. However, due to copyright consideration, we cannot publish very popular content on the Internet, so, we have chosen the second experimental method.

Our measurement program is a script following the BitTorrent's client protocol [6]. We choose a large and popular file and run the script to connect to the trackers of this content. The script connects to the tracker every 30 minutes and gets a list of the online peers with their progresses in the system. Since each time the tracker returns with a set of randomly chosen peers, we can view them as a "snapshot" for all the peers in the BitTorrent system.

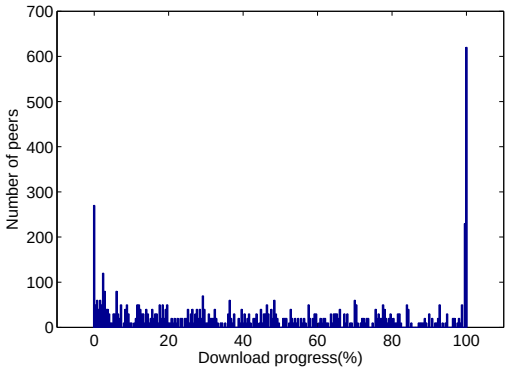


Figure 7. Histogram of the download peers with the degrees of download progress, observed from the real world BitTorrent application

For the independence of the data, we filter out those records of the same IP address which appear in more than one "snapshot". To make sure that the system is under its stable state, the measurement is started at least two days after the .torrent file was published.

We traced the BitTorrent download of the popular movie "Star Wars: Episode III - Revenge of the Sith", which was introduced on the Internet on May 29, 2005 and attracted a huge number of downloads. Figure 7 shows our measurement results from 50 "snapshots" taken by the script. We can see that the peer distribution follows the asymmetric U-shaped curve, which again confirms our analytical results. By numerically analyzing the measurement results, we have the following findings: First, there are 13.5% of the download peers with pieces less than 5%. This indicates that with limited optimistic unchoking, the peers with very few pieces to upload still have some difficulty to get unchoked by other download peers, and have to spend more time in downloading the pieces when most of the connections are ruled by TFT unchoking strategy. Second, there are 15.6% of the download peers with pieces more than 95%, and more astonishing, there are 6% of peers holding 99.9% of the pieces, meaning that many of them are pending in seeking for the last few pieces of the file, as the pieces they request are relatively rare in the system. Although BitTorrent is regarded as free of the last block problem [8], we observe that peers do have difficulty in downloading their last data. Moreover, we find that there are more peers blocked at the end of the download progress than at the beginning of it, which verifies our analytical result and could be explained by the altruistic uploading in the optimistic unchoking connections. Our last finding is that there is a slight linear decrease of the numbers of the peers with pieces between 20% and 80% observed. We believe this is because some peers abort the system before they could finish their download job, as shown in our analytical results in Figure 2(c).

Table II. Dying Process Measurement

Dying System	Num. of Peers	Maximum Num. of Peers per Bin	$\log_{10}(N \cdot W^2)$
1	18	5	−1.991
2	18	2	−2.655
3	11	4	−1.186
4	20	2	−2.704
5	12	3	−2.163
6	7	3	−1.172
7	23	3	−2.323
8	17	3	−1.655
9	11	4	−1.527
10	5	2	−1.648
11	10	2	−2.182
12	10	3	−1.785
13	9	3	−1.185
14	15	4	−2.256

6.3. Experiment 3: Dying Process Measurement

In the dying process analysis in Section 5, we state that peers tend to form clusters rather than evenly distributed on the download job process, when the system in under its dying process. In this experiment, we verify this statement by studying the distribution of the peers on the download progress in a number of dying BitTorrent systems. We also use the "Cramer-Smirnov-Von-Mises" test [28] to investigate whether or not these peers are following an uniform distribution on the download progress. The summary of the measurement study is shown in Table II.

We used the method described in Experiment 2 to study a number of dying BitTorrent systems, with each of them having less than 30 online peers. As there are so few peers, we could obtain a global view. For each dying system, we evenly divided the entire download progress into bins, and the number of the bins equals to the number of the peers observed. By studying how the peers are falling in these bins, we could have an overview of the peer distribution. We use the maximum number of the peers falling in a single bin as an indication for the unevenness of the peer distribution in each dying system, and report them in Table II.

We also use the "Cramer-Smirnov-Von-Mises" test to study how the distribution of the observed peers on the download progress deviates from an uniform distribution. In particular, we calculate W^2 for each dying system as

$$W^2 = \int [S_N(x) - F(x)]^2 f(x) dx$$

where $S_N(x)$ is the cumulative distribution of the N observed samples with a hypothesis function whose cumulative distribution is $F(x)$ and whose density function is $f(x)$. In our case, the hypothesis function is simply an uniform distribution and the samples are the observed

Table III. Simulation Settings

Setting	Peer Num.	Num. of Clusters	Download Progress for each Cluster
1	24	1	50%
2	24	2	30%, 80%
3	24	3	10%, 43%, 86%
4	24	4	15%, 40%, 65%, 90%
5	24	6	5%, 22%, 39%, 56%, 73%, 90%
6	24	8	4.5%, 17%, 29.5%, 42%, 54.5%, 66%, 78.5%, 90%
7	24	12	8.3%, 16.6%, 25%, 33.3%, 41.6%, 50%, 58.3%, 66.6%, 75%, 83.3%, 91.6%, 99%

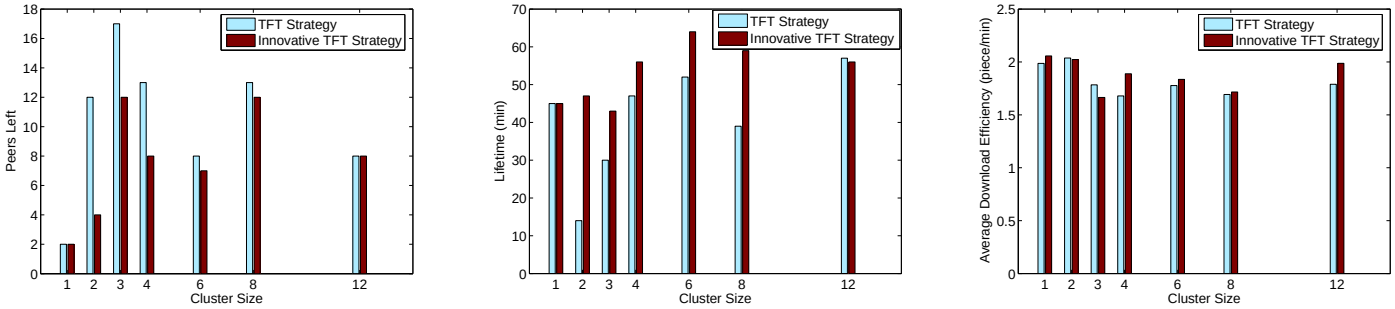


Figure 8. (a) The numbers of the peers left in the dead systems; (b) the lifetimes of the systems; (c) the average downloading efficiencies for the finished peers under each setting for peers playing the two strategies

download progresses of the peers. We use the value $\log_{10} W^2$ to show the degree of the observed distribution deviating from the uniform distribution. The more negative the value of $\log_{10} W^2$ is, the more the samples deviate from the hypothesis distribution function. From Table II, we find that the values of $\log_{10} W^2$ for each dying system is between -1 and -3, indicating that the peers' download progresses are statistically not uniformly distributed.

In summary, by studying the BitTorrent systems under the dying process in the real world, we validate our assumption that peers are not evenly distributed, but tend to form clusters with each other on the download job progress under the dying process of BitTorrent systems. As shown in Section 5, the clustering of the peers will deteriorate the system's stability greatly. Our observation also provides practical reasons for the simulation settings in Experiment 4 in the following section.

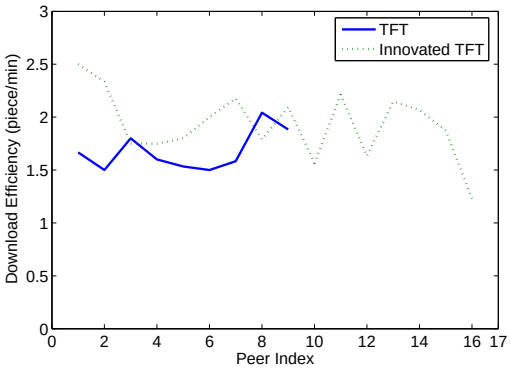


Figure 9. The downloading efficiencies for the finished peers in the setting 4, under the original TFT strategy and the innovative strategy

6.4. Experiment 4: Dying Process Simulation

We use the discrete-event BitTorrent simulator in the first experiment to study the dying process and the innovative incentive mechanism of the BitTorrent system in this experiment. Our objective is to investigate the performance of the system when the peers are playing the original TFT strategy and our innovative one.

In the experiment, we start the simulated BitTorrent system with different settings, as shown in Table III. The basic idea is to divide the peers equally into different clusters according to the download job progress, and simulate the dying process under each setting. For example, in setting 4, we divide 24 peers into 4 groups, with 6 peers in each one. The peers in the first group will each keep 15% of the pieces randomly; the peers in the second group each keep 40%; ... and so on. For all the settings, the parameter α of the innovative TFT strategy is set as 0.8.

During the execution, new peers will not arrive and the peers will depart the system immediately after they have finished their download jobs, so the $\Delta T = 0$. When the pieces of the file are not complete among the online peers, the system is declared dead and the execution is terminated. We are interested in three metrics of the system, which are: 1) the number of the peers left in the dead system which could not finish the download; 2) the lifetime of the system; and 3) the average download efficiency for the peers which have finished the download job.

We show our simulation results in Figure 8. Figure 8(a) gives the number of the peers left in the dead system, Figure 8(b) presents the lifetime of the system under the different settings, and Figure 8(c) shows the average download efficiencies of those peers which have finished the job in the system. We could see that when the peers are swarmed as one cluster, both the original TFT strategy and our innovative strategy performs well with very few peers left;

however, when the peers are gathered as two clusters with 50% of the download job progress as a distance, our innovative TFT strategy enables more peers to finish their jobs and prolongs the system's lifetime more than twice of the one under the original TFT strategy. When the number of the clusters increases, it is observed that the innovative strategy outperforms the original TFT less, and when there are 12 clusters, the two strategies again achieve the same result, but both leaves 8 peers unfinished. From the third figure, we can see that although the lifetime differs greatly under some settings, the average download efficiencies of the finished peers have little differences, which means statistically, our system will not cause a significant delay for the download peers.

However, by analyzing the log file of the simulator, we find some unfairness in the download time among the finished peers. In Figure 9, the download efficiencies of the finished peers under the two strategies are presented in Setting 4. For the original TFT strategy, only 9 peers finished the download, and their download efficiencies did not differ much; however, for the 16 peers which had finished the job under the innovative TFT strategy, we observe that the download efficiency for the last peer leaving the system is much smaller than the others. This obvious unfairness could be explained as: since we enable the peers with the global view in this experiment, usually the last finished peer will be kept in the system much longer than the peers that depart earlier, as it must wait for other peers to exchange among themselves to some extent before it could download the last few pieces and leave the system under our innovative TFT strategy.

7. Conclusion

In this paper, we first develop a mathematical model to study the behaviors of the peers in BitTorrent and the performance of the system. By numerically solving the model, we find that under the stable state, the distribution of the peers on the download job progress follows an asymmetric U-shaped curve, with relatively more peers blocked at the beginning and at the end of the download job progress due to the TFT unchoking strategy; and the optimistic unchoking strategy compensates the peers at the beginning of the job to some extent. We also find that the departure rate of the seeds and the abort rate of the unfinished download peers influence the peer distribution in different ways notably. The observations from simulations and the real world measurements support our analytical results.

We also study the content availability of the BitTorrent system and its dying process with our model. We find that the system is unstable when the peers are unevenly distributed on the download progress. Moreover, we find that the death of the system is usually triggered by the departure of the peer which is of great importance for the integrity of the file. An innovative TFT unchoking strategy is proposed aiming to improve the system's stability. By comparing the two TFT strategies with the simulation, we find the innovative strategy helps in preserving the content availability and prolongs the lifetime of the BitTorrent system extensively, only at a sacrifice of a longer download time for the last few finished peers, but will not lower the overall download efficiency.

REFERENCES

1. Napster. http://www.napster.com.
2. Gnutella. http://www.gnutella.com.
3. LimeWire. http://www.limewire.com.
4. KaZaA. http://www.kazaa.com.
5. BitTorrent. http://www.bittorrent.com.
6. BitTorrent Protocol Specification v1.0. http://wiki.theory.org/BitTorrentSpecification.
7. eDonkey. http://en.wikipedia.org/wiki/EDonkey2000.
8. Cohen B. Incentives build robustness in bittorrent. *Proceedings of Workshop on Economics of Peer-to-Peer Systems*, June 2003.
9. Saroiu S, Gummadi P K, Gribble S D. A measurement study of peer-to-peer file sharing systems. *Proceedings of Multimedia Computing and Networking*, January 2002. SPIE Press: Bellingham, WA, 2002; 156-170.
10. Izal M, Urvoy-Keller G, Biersack E W, Felber P A, Al Hamra A, Garcés-Erice L. Dissecting bittorrent: five months in a torrent's lifetime. *Proceedings of Passive and Active Measurement Workshop (Lecture Notes in Computer Science, vol. 3015)*, Barakat C, Pratt I (Eds.). Springer: Berlin, 2004; 1-11.
11. Pouwelse J, Garbacki P, Epema D, Sips H. The bittorrent p2p file-sharing system: Measurements and analysis. *Proceedings of International Workshop on Peer-to-Peer Systems (Lecture Notes in Computer Science, vol. 3640)*, Castro M, Van Renesse R (Eds.). Springer: Berlin, 2004; 205-216.
12. Guo L, Chen S, Xiao Z, Tan E, Ding X, Zhang X. Measurements, analysis, and modeling of bittorrent-like systems. *Proceedings of Internet Measurement Conference*, October 2005. USENIX Association: Berkeley, CA, 2005; 35–48.
13. Legout A, Urvoy-Keller G, Michiardi P. Rarest first and choke algorithms are enough. *Proceedings of Internet Measurement Conference*, February 2006. ACM Press: New York, NY, 2006; 203-216.
14. Bharambe A R, Herley C, Padmanabhan V N. Analyzing and improving bittorrent performance. *Proceedings of IEEE INFOCOM*, April 2006. IEEE: Piscataway, NJ, 2006.
15. Ramachandran K, Sikdar B. An analytic framework for modeling peer to peer networks. *Proceedings of IEEE INFOCOM*, March 2005. IEEE: Piscataway, NJ, 2005; 215- 269.
16. Ge Z, Figueiredo D R, Jaiswal S, Kurose J, Towsley D. Modeling peer-peer file sharing systems. *Proceedings of IEEE INFOCOM*, March 2003. IEEE: Piscataway, NJ, 2003; 2188- 2198.
17. Yang X, de Veciana G. Service capacity of peer to peer networks. *Proceedings of IEEE INFOCOM*, March 2004. IEEE: Piscataway, NJ, 2004; 2242- 2252.
18. Qiu D, Srikant R. Modeling and performance analysis of bittorrent-like peer-to-peer networks. *Proceedings of ACM SIGCOMM*, August 2004. ACM Press: New York, NY, 2004; 367-378.
19. Massoulie L, Vojnovic M. Coupon replication systems. *Proceedings of ACM SIGMETRICS*, June 2005. ACM Press: New York, NY, 2005; 2-13.
20. Tian Y, Wu D, Ng K-W. Modeling, Analysis and Improvement for BitTorrent-Like File Sharing Networks. *Proceedings of IEEE INFOCOM*, April 2006. IEEE: Piscataway, NJ, 2006.
21. Adar E, Huberman B A. Free riding on gnutella. Technical Report, CSL-00-3. Xerox PARC, 2000.
22. Trivedi K S. *Probability and Statistics with Reliability, Queuing, and Computer Science Applications*. John Wiley and Sons: New York, NY, 2002.
23. Sherwood R, Braud R, Bhattacharjee B. Slurpie: A cooperative bulk data transfer protocol. *Proceedings of IEEE INFOCOM*, March 2004. IEEE: Piscataway, NJ, 2004; 941- 951.
24. Gkantsidis C, Rodriguez P. Network coding for large scale content distribution. *Proceedings of IEEE INFOCOM*, March 2005. IEEE: Piscataway, NJ, 2005; 2235-2245.
25. Bhagwan R, Savagen S, Voelker G. Understanding availability. *Proceedings of International Workshop on Peer-to-Peer Systems (Lecture Notes in Computer Science, vol. 2735)*, Kaashoek F, Stoica I (Eds.). Springer: Berlin, 2003; 256-267.
26. Cuenca-Acuna F M, Martin R P, Nguyen T D. Autonomous replication for high availability in unstructured p2p systems. *Proceedings of Symposium on Reliable Distributed Systems*, October 2003. IEEE Computer Society: Los Alamitos, CA; 99-108.
27. Parker A. The true picture of peer-to-peer file-sharing. CacheLogic Presentation. CacheLogic, 2004.
28. Bock R K, Krischer W. *The Data Analysis Briefbook*. Springer: Berlin, Germany, 2006.