Modeling, Analysis and Improvement for BitTorrent-Like File Sharing Networks

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
Email: {ytian, dwu, kwng}@cse.cuhk.edu.hk

***Abstract*— In this paper, a simple mathematical model is presented for studying the performance of the BitTorrent [1] file sharing system. We are especially interested in the distribution of the peers with different states of the download job completedness. With the model we find that in the stable state the distribution of the download peers follows a U-shaped curve, and the parameters such as the departure rate of the seeds and the abort rate of the download peers will influence the peer distribution in different ways notably. We also analyze the file availability and the dying process of the BitTorrent file sharing system. We find that the system's stability deteriorates with the clustering of the 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 when the size of the community is small. An innovative peer selection strategy which enables more peers to finish the download job and prolongs the system's lifetime is proposed, in which the peers cooperate to improve the stability of the system by making a tradeoff between the current download rate and the future service availability. Finally, experimental results are presented to validate our analysis and findings.**

I. 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 the p2p file sharing systems, the upload bandwidths of the peers are exploited to speed up the download process, especially when the original server is overloaded.

From the appearance of Napster in 1999, many p2p file sharing applications have evolved, such as Gnutella, KaZaA and BitTorrent. 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 [16], BitTorrent accounts for 35% of all the traffic on the Internet, which is more than all the other p2p systems combined.

The main feature of BitTorrent, as well as other similar protocols, such as eDonkey, is that a file is divided into many blocks, so a peer could start to serve the others even if it does not have a complete file. According to the analysis in [9], 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 experience of the actual deployment and the results of academic research. Recent measurements [4][5] have verified the characteristics of scalability, pollution-resistance and efficiency of the protocol. Theoretical analysis [9][10] has also provided insights for people to understand the system. However, as shown in [6], the design is not perfect, further investigations of the issues not well studied previously are required, for improvement of the system. In this paper, we will focus on the following two topics.

The download efficiency and peer distribution: In traditional file download, there is a server and a client. For other p2p file sharing protocols such as Gnutella, although a peer could be a server as well as a client, for a particular file transmission, definitely there is a sender and a receiver. However, in BitTorrent, upload and download must be taking place simultaneously for a single file. BitTorrent adopts a "tit-for-tat" (TFT) peer selection strategy to prevent free-riding [11]. 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 and Gnutella-like systems, in which the download efficiency depends mainly on the server's capacity and the underlying network, in BitTorrent, the probability of successfully finding a proper upload/download partner also determines a peer's download rate. Our first goal in this paper is to analyze the download efficiency for a peer with different portions of the file, which is influenced by this probability, and find out the peer distribution determined by the peers' download efficiencies.

File availability and incentives to improve the system's stability: The lifetime of a BitTorrent network is the period in which it could provide a complete file from all the online peers. If the shares 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 its dying process, the network stability is influenced by the shares on each peer, as well as the peers' behavior in the system in a complex way. Another related issue is the relationship between BitTorrent's stability and its incentive mechanism. BitTorrent adopts the TFT strategy which enables a peer to select the neighbors which could help it to finish the download as fast as possible. However, when the system is in the dying process, another objective besides the download rate for a peer to pursue is the possibility that it could finish the download before the system dies. 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 completedness. By modeling a peer's behavior based on the state it is in, we have a better insight for the performance of the BitTorrent network in its stable state and the dying process. The other contributions in this paper are listed below:

- We have derived a peer distribution under the stable state from our model, and found that the parameters such as the seeds' depart rate and the download peers' abort rate will influence the distribution notably in different ways.
- From the results of the simulation and the real-world application measurement, we have found that our model is accurate in capturing the overall peer distribution and a peer's download efficiency regarding its job completedness.
- We have analyzed the file availability and the dying process of the BitTorrent system, and proved that the system is more unstable when the peers arrive in clusters.
- We have found that the original TFT peer selection strategy cannot help to improve the file availability and lower the probability of the system's death caused by a sudden departure of a finished peer. An innovative TFT strategy is proposed to increase the system's stability, in which the future service availability as well as the current upload rate is taken into consideration when a peer is selecting the partners.
- From the simulation experiments, we have found our innovative TFT strategy could improve the stability of the system extensively at a negligible sacrifice of the download efficiency. We have also observed the unfairness caused by the innovative strategy among the peers which have finished the download job.

The rest of this paper is organized as follow. For section II, we provide a brief introduction for the BitTorrent protocol. In section III, related work on p2p file sharing networks and BitTorrent are surveyed. We present a mathematical model and our modeling results under the stable state in section IV. Then we analyze the file availability and incentive mechanism for BitTorrent in section V, and propose our innovative peer selection strategy. Finally, in section VI, experimental results are presented to support our modeling results and demonstrate the effect of the innovative peer selection strategy.

II. BitTorrent Overview

BitTorrent is a protocol which is popular for its efficiency in distributing and downloading large files. In BitTorrent, a file is divided into many equal sized shares (typically 256KB) for download. A peer could download shares from other peers concurrently while uploading shares it holds to those requesting peers at the same time. In this way, the load is distributed among all the peers in the system [2].

In the BitTorrent architecture, there are three kinds of 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 shares for all the peers in the system. When a new peer joins in, it will connect to the tracker for a random list of the existing peers in the network to get started. All the peers in the network will periodically report to the tracker of their progresses. Those peers which have downloaded the whole file but not left the network are called seeds. The seeds improve the performance of the system from two aspects: first, they will contribute their bandwidths and upload the shares to other peers; second, they could insure the integrity of the file since they hold all the shares of it. We refer to those peers which have not finished the download job as the download peers, which compose the third component in the system. 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 select which share to request. Simply speaking, it will try to download a share 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 and does not have any shares, it will simply download any shares available to bootstrap itself. Another exception is the ”endgame model”: when a peer has almost finished its download, it enters the ”endgame model” and requests the last few shares from everyone.

A peer in BitTorrent usually keeps connections with many other peers. For any connection between two peers, there are two states: choked or not. A peer could choose to “choke” a connection by refusing to upload shares any more. Download peers in BitTorrent play a “tit-for-tat” (TFT) strategy to choose which connections to unchoke: it periodically monitors its current connections, and choose a fixed number (typically four) of the connections with the highest download rates to upload. However, the seeds do not play this strategy, for a seed, it will simply choose five download peers at the highest rate to upload, in order to distribute the file as quickly as possible.

Besides the TFT strategy, a download peer will also play a strategy called optimistic unchoking periodically to probe new partners. In optimistic unchoking, a peer randomly chooses a requesting peer to upload. Optimistic unchoking has two important meanings for the BitTorrent system: first, it allows a peer to discover better peers for share exchange; second, the newcomers without any shares to upload could get bootstrapped by accepting shares from the peers playing optimistic unchoking.

III. Related Work

P2p file sharing systems operate by allowing the peers to form an application level overlay and contribute their content in a cooperative way. To understand the performance of p2p networks, many models [7][8][9][10] have been proposed. In [7], an analytic framework is presented for studying the data traffic of p2p applications with various underlying networks. [8] proposes a model with the generality and flexibility of capturing the different architectures of the p2p file sharing system. For BitTorrent, a branching process is discussed for studying the transient regime of the BitTorrent system when the content is first introduced, and a Markov chain model is proposed for understanding the service capacity of the system under the stable state in [9]. A fluid model is presented in [10], where the expressions of the numbers of the download peers and the seeds could be obtained from the parameters as the peer arrival, departure and the upload/download rates. The analysis also proves that BitTorrent achieves very good scalability.

There are many measurements [4][5][6] based on real world applications and simulations for BitTorrent. Very good properties such as the efficiency in distributing a file, robustness against pollution and scalability to support a large number of the downloaders are observed in these measurements. However, in [6], it is found that BitTorrent has the last piece problem and cannot prevent a systematical unfairness.

Enhancements to BitTorrent are proposed in Slurpie [14] and Avalanche [15]. 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 block propagation of the file more efficient than transmitting the uncoded blocks; the system is also more robust under extreme situations with the sudden departure of nodes. And in [17], a genetic algorithm based neighbor selection strategy is proposed to improve the file distribution efficiency.

The availability problem for p2p file sharing networks is studied from two aspects. In [18], the availability of the host is measured and analyzed based on the Overnet file sharing system, in which a file is usually served by a single peer. For the file availability, [13] proposes an algorithm to improve the availability of the files given nodes with limited capacity to replicate the pieces of the files published by other nodes. The author assumes that the peer will be available with some probability but not related to the time. However, for BitTorrent, the availability problem is a little different compared with the other systems, since in BitTorrent it is not a single peer but the whole BitTorrent community involved to serve a particular file. Currently, most of the works on BitTorrent's availability are experimental, such as the measurements discussed in [4] and [5].

Our model for BitTorrent is influenced by the models in [9] and [10]. However, in the models of [9] and [10], 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 state they are in, thus enabling a deeper understanding of the system.

IV. Modeling and Analysis for Peer Distribution

When a file is first introduced into the Internet by a .torrent file, there will be a burst of download requests, but only a few (usually one) seeds are available for upload. After the initial burst phase, the BitTorrent system would reach its stable state. At the stable state, the arrival of the peers is not in a burst, but could be viewed as a Poisson process [9]. In this section, we present a mathematical model for BitTorrent and analyze the performance of the system in its stable state. From the modeling results, we find that the peer distribution follows a U-shaped curve, and parameters such as the departure rate of the seeds and abort rate of the download peers could influence the distribution differently.

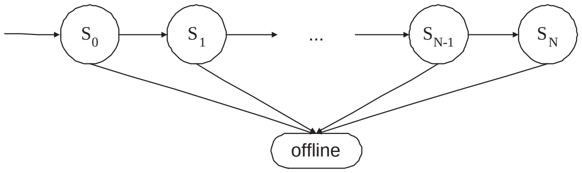

Fig. 1. The model for the BitTorrent file sharing system

A. The Model

Our model is illustrated in Fig. 1. In the 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 seeds. We choose the value of N as a integer big enough to embody the detailed difference for the peers with different amount of the shares, and the upper bound of N is M, which is the number of the shares.

When a new peer first enters 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 some shares to the peer, it will be a peer with $\frac{1}{N}$ portions of the file at the state S_1, and meanwhile it could upload the shares it holds to other download peers. The uploading and downloading processes continue until the peer has obtained all the shares and becomes a seed at the state S_N; however, it could also abort the system and go to the $offline$ state during the process before it finishes 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 Fig. 1.

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

- N: the total number of states in the model;
- $x_0(t), x_1(t), ..., x_{N-1}(t)$: the number of the download peers in the states of S_0, S_1, ..., S_{N-1} respectively;
- $y(t)$: the number of seeds;
- μ: the average upload rate for a peer, including the seeds;
- λ: the arrival rate of the new download peers;
- γ: the departure rate of the seeds;
- θ: the abort rate of the download peers;

- $\eta_{i,j}$: the efficiency of exchanging shares between a peer at the state of S_i and a peer at the state of S_j.

For the peers in the states of S_0, S_1, ..., S_{N-1}, S_N and $offline$, we model the arrival rate of the new download peers as a Poisson process with rate λ, and the seeds depart the system according to an exponential distribution with the rate of γ. For the download peers in the states of S_0, S_1, ..., S_{N-1}, they will abort the system according to an exponential distribution with the rate of θ. By applying a continuous time Markov chain model [12], we derive the transfer rates between two neighboring states as follows:

$$\begin{array}{lcl}
r(S_N, offline) & = & \gamma y(t) \\
r(S_{N-1}, S_N) & = & \begin{array}{l} \mu y(t) \times \frac{x_{N-1}(t)}{\sum_{j=0}^{N-1} x_j(t)} + \frac{x_{N-1}(t)}{\sum_{j=0}^{N-1} x_j(t)} \times \\ \mu \sum_{k=0}^{N-1} x_k(t) \times \eta_{N-1,k} \end{array} \\
\ldots\ldots & & \\
r(S_i, S_{i+1}) & = & \begin{array}{l} \mu y(t) \times \frac{x_i(t)}{\sum_{j=0}^{N-1} x_j(t)} + \frac{x_i(t)}{\sum_{j=0}^{N-1} x_j(t)} \times \\ \mu \sum_{k=0}^{N-1} x_k(t) \times \eta_{i,k} \end{array} \\
\ldots\ldots & & \\
r(S_0, S_1) & = & \begin{array}{l} \mu y(t) \times \frac{x_0(t)}{\sum_{j=0}^{N-1} x_j(t)} + \frac{x_0(t)}{\sum_{j=0}^{N-1} x_j(t)} \times \\ \mu \sum_{k=0}^{N-1} x_k(t) \times \eta_{0,k} \end{array} \\
r(S_0) & = & \lambda \\
r(S_i, offline) & = & \theta x_i(t), i = 0, 1, ..., N-1
\end{array} \tag{1}$$

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. We focus on the parameter $\eta_{i,j}$ now, which is the share exchange efficiency between the peers in the states of S_i and S_j. For a peer p_i in the state of S_i, on average it will have $\frac{i+0.5}{N}$ portions of the file, and are looking for the $\frac{N-i-0.5}{N}$ portions left from the other peers. When there is a connection between p_i and p_j, the probability of p_i choosing to upload to p_j is $\frac{i+0.5}{N} \times \frac{N-j-0.5}{N} = \frac{(N-j-0.5)(i+0.5)}{N^2}$; similarly, the probability of p_j choose to upload to p_i is $\frac{(N-i-0.5)(j+0.5)}{N^2}$. Since the peers play a TFT strategy, which means a peer will only upload to another peer when this peer is uploading to itself, the share exchange efficiency $\eta_{i,j}$ should be the product of the two probabilities, then we have $\eta_{i,j} = \frac{(N-i-0.5)(j+0.5)}{N^2} \times \frac{(N-j-0.5)(i+0.5)}{N^2}$.

B. Modeling results and Discussion

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 case, we have $\gamma \rightarrow \infty$ and $\theta = 0$. At the stable state of the model in Fig. 1, applying the principle that "*the rate of the flow out of a state = the rate of the flow into this state*"[12], we have a set of equations.

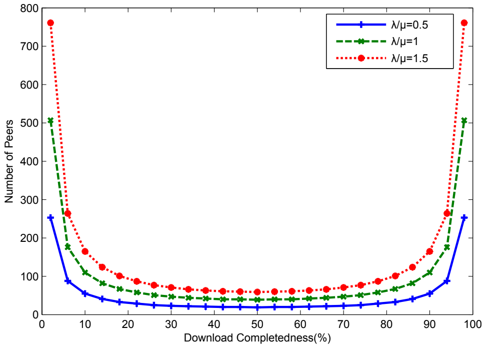

Fig. 2. Peer distribution in the stable state, without the seeds departure and the download peers aborting

$$\begin{array}{lcl}
\lambda & = & x_0 \times (2N-1) \times \mu \frac{\sum_{k=0}^{N-1} \frac{(2N-2k-1)(2k+1)}{16N^4} x_k}{\sum_{j=0}^{N-1} x_j} \\
\frac{x_0}{x_1} & = & \frac{(2N-3)\times 3}{(2N-1)\times 1} \\
\ldots\ldots & & \\
\frac{x_i}{x_{i+1}} & = & \frac{(2N-2i-3)\times(2i+3)}{(2N-2i-1)\times(2i+1)} \\
\ldots\ldots & & \\
\frac{x_{N-2}}{x_{N-1}} & = & \frac{1\times(2N-1)}{3\times(2N-3)}
\end{array} \tag{2}$$

Solving these equations, we have:

$$x_i = \frac{x_0(2N-1)}{(2N-2i-1)(2i+1)} \tag{3}$$

and

$$x_0 = \frac{\lambda}{\mu} \times \frac{16N^4}{(2N-1)} \times \sum_{j=0}^{N-1} \frac{1}{(2N-2j-1)(2j+1)} \tag{4}$$

The combination of (3) and (4) determines the number of the peers in each state, given the number of the states N, and the offered load $\frac{\lambda}{\mu}$ of the system.

Fig. 2 plots the number and the distribution of the peers with different degrees of download completedness at the offered load $\frac{\lambda}{\mu}$ of 0.5, 1 and 1.5 respectively, given the number of the states N as 25. From the modeling results, we find that the peer distribution follows a U-shaped curve, in which the peers are more concentrated in the states with very small or large portions of the file, and this could be explained with the relatively lower share exchange efficiencies of these states than the other states. We also 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 relationship among the numbers of the peers with different download completedness, will not get changed. Finally, 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, and this observation also conforms to the modeling results in [9].

We have previously assumed that $\gamma \rightarrow \infty$ and $\theta = 0$. However, in the practical applications, it is observed that many

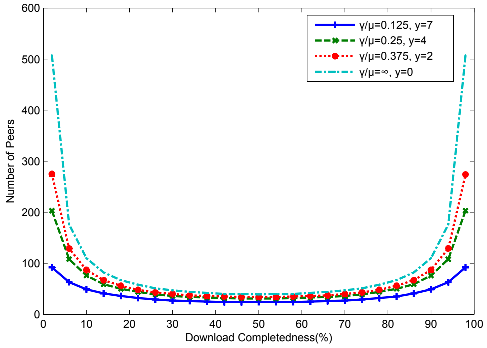

Fig. 3. Peer distribution in the stable state, influenced by the departure rate of the seeds

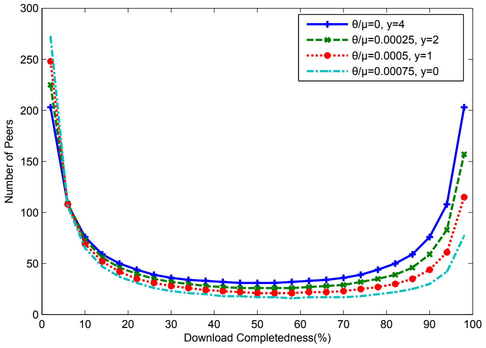

Fig. 4. Peer distribution in the stable state, influenced by the abort rate of the download peers

peers may stay in the system after they have completed their download and act as the seeds for a certain period of time [4]. And a peer may also abort the network before it has finished the download for some reasons, such as loss of the interest in the content. Based on these considerations, following we will discuss what will happen to the numbers and the distribution of the downloader peers, if more practical settings of the parameters γ and θ are introduced.

We set the offered load $\frac{\lambda}{\mu} = 1$ and the abort rate $\theta = 0$, and change the exit rate $\frac{\gamma}{\mu}$ from 0.125, 0.25 to 0.375 in our model. By applying the principle that "*the rate of the flow out of a state = the rate of the flow into this state*", we solve the equations drawn from formula (1) numerically, and obtain three curves shown in Fig. 3. We also give the curve of $\gamma \to \infty (\frac{\gamma}{\mu} \to \infty)$ for comparison. From the modeling results, we have two findings: (a) With the decrease of the exit rate, the number of seeds increases, however, the number of download peers decreases. This is because the smaller exit rate allows more seeds to stay in the system, and the seeds inversely help the download peers to finish the job and leave the system more easily. (b) The involvement of the exit rate will change the distribution of the download peers. When the exit rate decreases, the curves are getting more and more "flat". Since the peers could download from the seeds more efficiently with more seeds available, and they will transfer from state to state more quickly in the model of Fig. 1. In the extreme case when $\frac{\gamma}{\mu} \to 0$, which means the seeds will never depart the system, it could be forecasted that the distribution curve will be a horizontal straight line with the peers of the different percentages of the file being equal, since all the download peers will have a equal chance to get the shares from the seeds, and the number of the seeds are more than the download peers in the system.

We also study the case when $\theta \neq 0$, which means the download peers will abort the system. We set the offered load $\frac{\lambda}{\mu} = 1$ and the exit rate $\frac{\gamma}{\mu} = 0.25$, and change the abort rate $\frac{\theta}{\mu}$ from 0.00025 to 0.00075. The results numerically derived from the formula (1) are presented in Fig. 4. In this 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 of the smaller percentages of the file.

In summary, we obtain the number and the distribution of the peers at their different degrees of download completedness with the model. In the ideal case of $\gamma \to \infty$ and $\theta = 0$, expressions of the peer numbers are derived; and with the consideration of $\gamma \nrightarrow \infty$ and $\theta \neq 0$, similar results are obtained by numerically solving the model. It is shown that although the parameters of γ and θ will influence the peer distribution in different ways notably, the distribution still follows the U-shaped curve. In this meaning, the expressions in (3) and (4) are accurate in capturing the essence of the BitTorrent's peer distribution.

V. File Availability and Incentive Mechanism

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

A. *The File Availability*

The availability problem arises because of the decentralized nature of the p2p file sharing networks, especially when the peers are organized in an unstructured overlay. Usually, people correlate the host uptime with the file availability, such as the measurement in [13] and [5]. However, in BitTorrent, since all the peers download a single file cooperatively, and there are not the roles of senders and receivers, the file availability is more associated with the number and the distribution of the peers in the system. As mentioned in section II, the existence of the seeds could ensure the availability of a complete file since they hold all the shares; however, with the absence of the seeds, it is not necessary that the file is complete since each peer may only keep a part of it.

To estimate the file availability of a BitTorrent system, we assume that the shares are evenly distributed among all the online peers. This is a reasonable assumption because all the download peers play the LRF strategy. 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 shares, then the probability that a particular share is not held by p_i is $1 - \frac{m_i}{M}$, where M is the total number of the shares for the file. Furthermore, the probability that this share is not held by all the online peers is $\prod_{i=1}^{S}(1 - \frac{m_i}{M})$. Note that for BitTorrent, the missing of any single share systematically will cause the incompleteness of the file, thus, the estimated file availability at time t could be computed as:

$$Av(t) = 1 - \prod_{i=1}^{S}(1 - \frac{m_i}{M}) \approx 1 - (1-1)^{y(t)} \prod_{i=0}^{N-1}(1 - \frac{i+0.5}{N})^{x_i(t)} \tag{5}$$

The last expression in (5) is derived from the model represented in Fig. 1. We approximate the probability of not holding a particular share by a peer in the state S_i as $(1 - \frac{i+0.5}{N})$, the probability that the share is not held by all the peers in the state S_i is the $x_i(t)$ power of $(1 - \frac{i+0.5}{N})$, here $x_i(t)$ is the number of peers in the state of S_i. For the seeds in the state S_N, the probability of not keeping the share is $(1 - 1)^{y(t)}$, which is 0 if there is at least one seed, and 1 otherwise.

Note that in (5) , if there are seeds, we would have $Av(t) = 1$. Another thing should be noticed is that the peers with large percentages of the file contribute greatly to the file availability: consider a system composed of 20 peers with each keeping 10% of the shares, the availability calculated with (5) is 0.88; however, a system with only 5 peers with 10% of the shares and 2 peers with 90% of the shares has an availability of 0.99.

B. Analysis of the Dying Process

Each BitTorrent system has a lifetime. When the content is first published on the Internet by the .torrent file, usually a seed is available for uploading during a certain period of time. However, with the departure of the original seed and the decrease of the new peers arriving, at a certain moment, the file will become incomplete among all the online peers, and the peers staying in and arriving later cannot obtain a complete file, unless a seed rejoins the network. We call a system dead if the shares 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 the lifetime of a BitTorrent system; and the phase during which the system is vulnerable to die is called its dying process. Actually, the lifetime of a BitTorrent network varies from a few days to several months, depending on the popularity of the content.

We apply the model presented in Fig. 1 to study the dying process of the system. However, unlike the analysis of the stable state, which models the efficiency parameter $\eta_{i,j}$ differently for different state pairs (S_i, S_j), in the dying process, the size of the community is very small, factors such as the individual's upload/download bandwidths and the congestions of the underlying network will influence the average efficiency more significantly than the upload probability, so, we assume $\eta_{i,j} = \eta$ as a constant for each peer in the system for simplicity. We also assume $\gamma \to \infty$ and $\theta = 0$ in the our analysis.

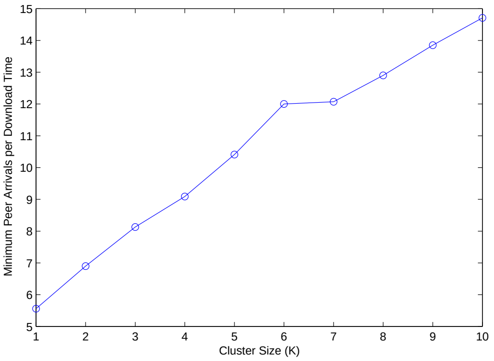

Fig. 5. The relationship between the required minimum peer arrivals per download time and the cluster size in the $k - cluster$ arrival pattern

First we model a smooth arrival of the new peers in the system. 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 [10], for a single peer, the time required to finish the download is $\frac{1}{\eta\mu}$, so, for the period of completing a download job, there are $\frac{\lambda}{\eta\mu}$ new peers arriving; the $\frac{\lambda}{\eta\mu}$ could also be interpreted as the number of the peers in the system when a peer is leaving. It is obvious that if λ decreases, the file availability calculated with (5) will decrease since the peers staying in the system are rarer, and for a certain availability, there exists a minimum λ, which also determines a minimum peer arrivals per download time $\frac{\lambda}{\eta\mu}$. We use the "minimum peer arrivals per download time" as an important metrics in our analysis. If we set the availability threshold as 0.98, which means we declare a system dead if the availability is lower than this threshold. Applying the last expression of (5), we find that in order to keep the file availability above this threshold, the minimum peer arrivals per download time should be 5.56, if the λ decreases such that $\frac{\lambda}{\eta\mu}$ is smaller than 5.56, the system will die under the smooth arrival pattern. In other words, if there are more than 5.56 peers in the system at any time, under the smooth arrival, the file availability will be kept above the threshold of 0.98. Obviously, this is a very good result, only 6 peers online will guarantee the integrity of the file for the system.

However, smooth arrival is not a good approximation for the peer's behavior in the real world, especially in the dying process of a system with fewer peers, in which the new peer's arrival is more likely to be in jitters. Thus, 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 $k = 2$, a arrival rate of λ means two peers arrive in the interval of $\frac{2}{\lambda}$, instead of one peer arriving at the interval of $\frac{1}{\lambda}$. Applying the $k - cluster$ arrival pattern to the last expression of (5), we find that the required minimum peer arrivals per download time increases significantly with k, given the availability threshold as 0.98. The results are plotted in Fig. 5.

From the modeling results in this figure, we find the system is more prone to die in the $k - cluster$ arrival pattern, if the value of k is big. For example, if the peers arrive in a cluster of 6, on average, the system should manage to keep nearly 12 peers in the system all the time in order to be alive.

C. Incentive Mechanism for the Dying Process

In BitTorrent, peers are playing the TFT strategy during the selection of the upload/download 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 [10] points out that under the TFT strategy, the peers at the stable state will reach a Nash equilibrium in which each peer will upload at its highest rate to maximize its utility, which is the peer's current overall download rate.

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 download 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 download rate in the dying process, since an incomplete file means nothing for most types of the contents.

From the studies on the file availability and the dying process of the system, we find that the death of a BitTorrent network is usually caused by the departure of a peer which has finished its job and leaving the system with only the peers keeping smaller percentages of the file. Unfortunately, the built-in TFT peer selection strategy cannot prevent those peers which have great importance for the stability of the system from completing their job and departing the system. Consider a scenario of a small BitTorrent network demonstrated in Fig. 6(a). In this system, assume the file is divided into 10 equal sized shares, indexed as 0, 1, 2,..., 9. There are 10 peers in the network, each keeping different shares, and they form a topology as shown in the figure. We assume that all peers have the same upload and download speed, 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 share 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 share from peer A in return, it will request a share from the set of $\{0, 1, 2, 3, 4, 5, 7\}$ with a equal chance, since they are the local rarest shares for it. Suppose peer A will depart the system immediately after it has finished exchanging the shares with peer J successfully. The probability that the file is complete among the peers B to J after peer A's departure is $1/7$, since the only chance that the file is complete is that peer J had requested share 0 from peer A, which happens at a probability of $1/7$. However, if peer J refuses to respond to A and exchanges the shares with the other peers first, it is likely that the system will survive with a higher probability. For example, when the peers B to J exchange shares so that each keeps 4, as demonstrated in Fig. 6(b) , when a peer of B to J with the share 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 shares with A, the probability now becomes $1/3$, since peer J will only request a share in the set of $\{0, 2, 4\}$.

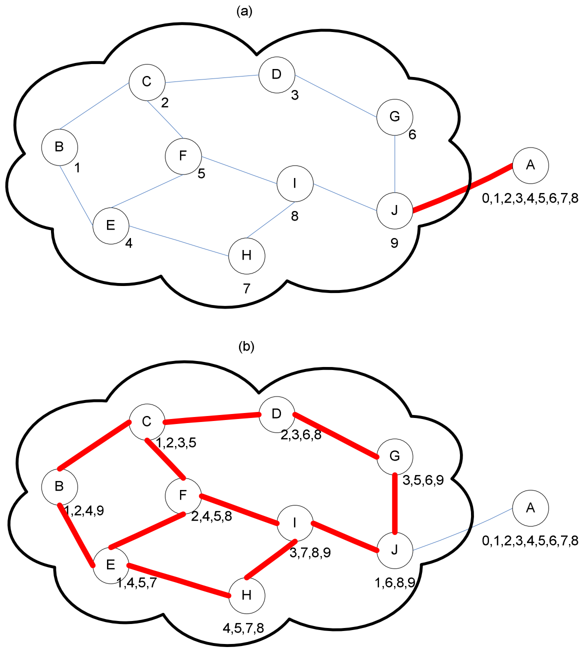

Fig. 6. The demonstration of the peer selection strategy and the system's stability

Motivated by this observation, we find it is important for the system to keep the seeds or the peers with large portions of the shares online, to preserve the integrity of the file. 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 shares is not a good idea, since the peers are selfish, it will pursue to maximize its utility, which is the overall download rate under the BitTorrent's current TFT strategy. Thus, the new mechanism must be incentive-compatible so that each peer will play the strategy which could benefit the individual's utility as well as the social welfare, which is the download efficiency combined with the stability of the network. 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, the system's efficiency will decrease; however, we will show that this sacrifice could be negligible in the simulation experiment in section VI.

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 peer selection strategy with the consideration of the future service availability. Consider a peer p_i with m_i shares and a upload rate of μ_i, and p_j is its neighbor which has m_j shares and is at a upload rate of μ_j and a download 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 upload 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 period of time that p_j could finish uploading all the shares p_i needs at the current upload rate, note that $m_j \frac{M-m_i}{M}$ is the estimation of the number of shares 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 finish 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:

$$g_j = \mu_j \sum_{i=0}^{\lceil \frac{T}{t_{int}} \rceil} \alpha^i \tag{6}$$

where $T = \min(T_1, T_2)$ is the estimated service time from p_j, t_{int} is the execution interval of the peer selection strategy, and α could be thought as a tradeoff factor between the file availability and the download efficiency, with a value in the range of $[0, 1)$. In our mechanism, 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 upload rate μ_j, then it is BitTorrent's original peer selection strategy; and for $\alpha \to 1$, it means p_i will consider p_j's future upload service with the equal importance as the upload rate currently obtained.

If we have 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 completedness 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 shares as $m_j = \frac{M^2}{6M-5m_i}$, which is a value very close to m_i. In other words, when applying the innovative peer selection strategy, peers will be more likely to choose to upload to a peer which has a similar job completedness, and leave those peers with higher percentages of the file for later use. Given that all the peers are playing the innovative peer selection strategy, the global effect is that the peers will form clusters according to their job completedness and cooperation is more likely to happen between the peers in the same cluster. If we revisit the scenario demonstrated in Fig. 6(a), under the innovative peer selection strategy, it is definitely that the peers B to J will keep peer A waiting while they cooperate mutually among themselves first.

BitTorrent's incentive mechanism works on the condition that the majority of the peers are playing TFT. However, our mechanism imposes a stricter requirement on the peers, we require that all the peers are playing the innovative TFT strategy; otherwise the mechanism will become meaningless. For example, in the network demonstrated in Fig. 6(a), if peer J deviates from the strategy by exchanging with peer A first, it is useless for the other peers to play the strategy. However, we believe that this kind of situation is not likely to happen since for a rational peer, violating the strategy cannot increase it utility unilaterally, but will lower its possibility of finishing the download job.

Finally, for the deployment of the innovative TFT strategy, since it is proposed aiming to improve the file 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 timing of the strategy switch: when the tracker detects that the file availability is lower than some threshold, it could notify the online peers to switch the peer selection strategy from the original TFT to our innovative one.

VI. Experimental Results

In this section, three sets of experiments are presented, our purpose is to validate our analysis results and support our innovative TFT strategy, as discussed in section IV and section V.

A. Experiment 1

In experiment 1, we study the results from a discrete-event simulation of a BitTorrent-like network. In the simulated network, we set the file size as $100M$, and let the share size to be the default, which is $256K$. For the download rate, since we are studying the distribution of the peers essentially influenced by the share exchange efficiency, $\eta_{i,j}$, each peer is allowed to obtain one share from an unchoked connection per minute, and the number of the maximum unchoked connections is set as four. 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. To bootstrap the system, 10 seeds are inserted into the system at the beginning. We observed that the system reaches its stable state after 200 minutes. However, due to the randomness of the peers' behavior, 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, the result is shown in Fig.7. We could see that on average, the peers follow a U-shaped distribution, which verifies that our model is a good approximation for the network in the stable state. We also find a linear decrease of the peer numbers from 10% to 80% of the job completedness, which we believe is caused by the abort rate, as demonstrated in our modeling results in Fig. 4. The abnormal huge number of peers near 0% of the job completedness is caused by the strict TFT strategy adopted in our simulation, in which a new peer could only get its first few shares from the seeds.

In Fig.8, we traced a single peer from its join to its leave as a seed peer, and recorded the download completedness 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 rate of the peer is almost a constant except for two periods

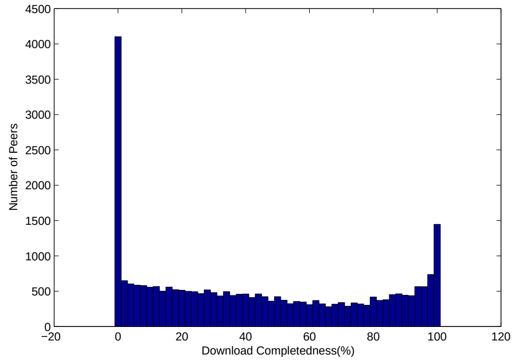

Fig. 7. Aggregated peer distribution from the simulation

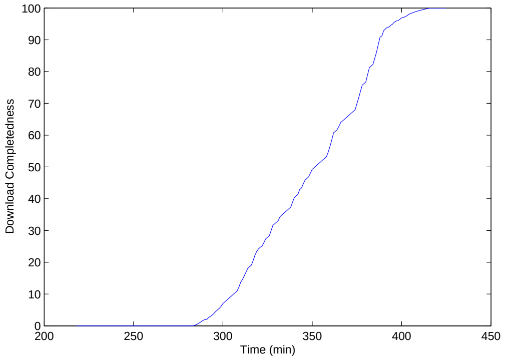

Fig. 8. The trace of the download progress for a single peer

before the 300^{th} minute and after the 380^{th} minute, which are at the beginning and at the end of the download respectively, and in which the download rate reduces greatly. Recall that in our previous analysis, a peer's download rate at its different states of the job completedness is determined by its efficiency in exchanging shares with the other peers, the download rate shown in the figure presents the peer's evolution in its download efficiency and supports our model.

B. Experiment 2

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

Generally, there are two possible methods to gather the real-world data: one is by setting up a tracker and introducing a file into the Internet; and the other method is to connect to the tracker of a file existing on the Internet. Since we are studying the behavior of a BiTorrent system with many peers at 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 experiment method.

Our measurement software is a script following the BitTorrent's client protocol [1]. 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 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. 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 in its stable state, the measurement is started at least two days after the .torrent file was published.

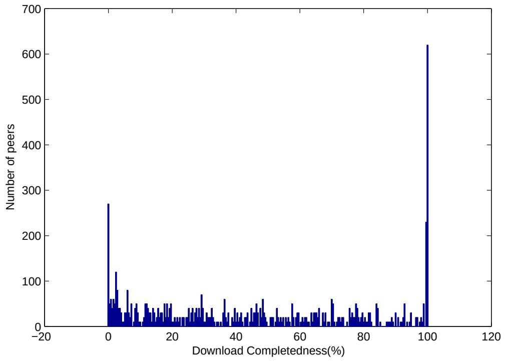

Fig. 9. Histogram of the download peers with the degrees of job completedness, observed from the real world BitTorrent application

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. Fig. 9 shows our measurement results from 50 "snapshots" taken by the script. We can see that the peer distribution also follows the U-shaped curve, which again confirms our modeling results. By numerically analyzing the measurement results, we also have the following findings: First, there are 13.5% of the download peers with shares less than 5%. Although BitTorrent use the optimistic unchoking strategy to help the new peers to bootstrap, the peers with few shares to upload still have difficulty to get unchoked by the other peers, and have to spend more time in downloading the shares when most of the peers play the TFT unchoking strategy. Second, there are 15.6% of the download peers with shares more than 95%, and more astonishing, there are 6% of peers holding 99.9% of the shares, which means that many of them are pending in seeking for the last few shares of the file, since the shares they request are relatively rare in the system. Although BitTorrent is regarded as free of the last piece problem [2], we observe that peers do have difficulty in downloading the last few shares. Our final finding is a slight linear decrease of the numbers of peers with shares between 20% and 80% observed. We believe this is because a certain number of peers abort the system before they complete their download, as shown in our modeling results in Fig. 4.

C. Experiment 3

We use the discrete-event BitTorrent simulator in the first experiment to study the dying process and the innovative

TABLE I
SIMULATION SETTINGS

Setting	Peer #	# of Clusters	Completedness 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	$8.3\%, 16.6\%, 25\%,$ $33.3\%, 41.6\%, 50\%,$ $58.3\%, 66.6\%, 75\%,$ $83.3\%, 91.6\%, 99\%$

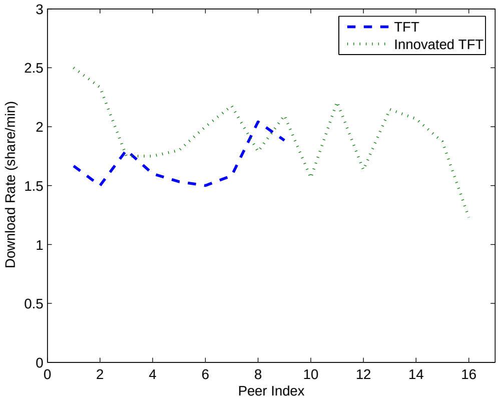

Fig. 10. The download rate for the finished peers in the setting 4, under the original TFT strategy and the innovative strategy

incentive mechanism of the BitTorrent network in this experiment. Our purpose 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 network with different settings, as shown in Table I. The basic idea is to divide the peers equally into different clusters according to the job completedness, 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 each keep 15% of the file shares randomly; the peers in the second group keep 40%; ... and so on. For all the settings, the parameter α of the innovative 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 the download, so the $\Delta T = 0$. When the shares 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 peers left in the dead system which could not finish the download; 2) the lifetime of the system, and 3) the average download rate for the peers which have finished the download job.

We show our simulation results in Fig.11 . Fig.11(a) shows the number of peers left in the dead system; Fig.11(b) presents the lifetime of the system under the different settings, and Fig.11(c) gives the average download rate of those peers which have finished downloading 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 completedness as a distance, our innovative TFT strategy enables more peers to finish the job and prolongs the 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 rates of the finished peers have little differences, which means on average, our system will not cause a significant delay for the download peers.

However, by analyzing the log file of the simulator, we find unfairness in the download time among the finished peers. In Fig. 10, the download rates 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 rates did not differ much; however, for the 16 peers which have finished the job under the innovative TFT strategy, we observed that the download rate 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 shares and leave the system under our innovative TFT strategy.

VII. CONCLUSIONS AND FUTURE WORK

In this paper, we first develop a mathematical model to study the behaviors of the peers in BitTorrent and the performance of the network. We find that under the stable state, the distribution of the peers regarding their download job completedness follows a U-shaped curve. We have obtained the expressions for the numbers of the download peers with different percentages of the file, under the assumptions of zero abort rate and infinite departure rate. By numerically solving the model, we find that the parameters of the departure rate of the seeds and the abort rate of the unfinished download peers influence the peer distribution differently and notably. The simulation results and the real world application measurements support our modeling results.

We also study the file availability of the BitTorrent system and its dying process with our model. We find that the system is unstable when the peers arrive in clusters. Moreover, we find that the death of the system is usually triggered by the departure of the last seed which is of great importance for

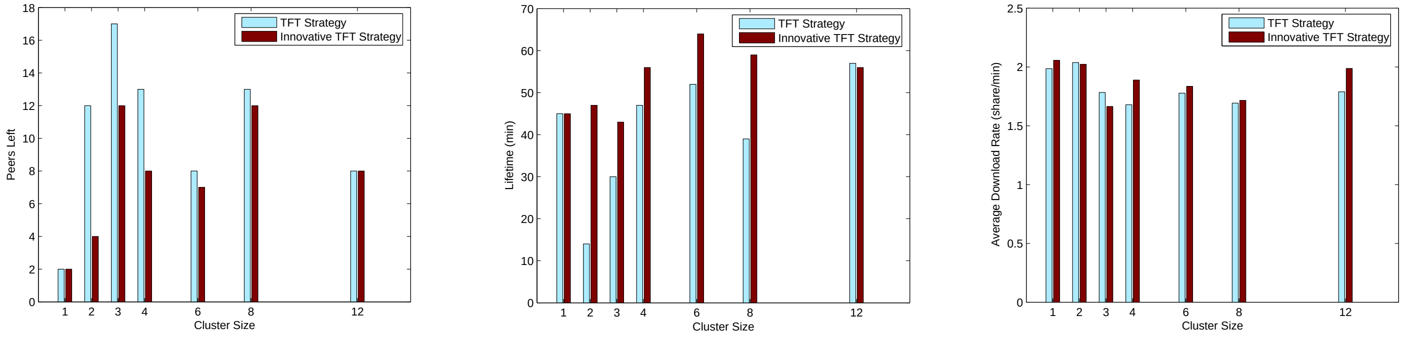

Fig. 11. (a) The number of the peers left in the dead system (b) The lifetime (c) The average download rate under each setting for peers playing the two strategies

the integrity of the file. An innovative TFT peer selection 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 file availability and prolongs the lifetime of the BitTorrent system extensively, only at a sacrifice of a longer download time for the last few finishing peers, but will not lower the download efficiency systematically.

Our proposed TFT strategy uses a parameter α as a tradeoff factor between the system's file availability and the peer's download efficiency. However, the optimal choice of α is a problem with the interactions of the number and the distribution of the peers, and their behaviors as the arrival, abort and departure, as well as each peer's upload/download bandwidths. Obviously this computationally complex optimization problem is not suitable to be solved for the dynamics of the BitTorrent network. The suboptimal choice of α based on the local information for an individual peer is our future research direction.

Another consideration in the deployment is how to protect the peers playing the innovative TFT strategy against the behaviors of the irrational peers. As we have discussed, the existence of even a single irrational peer will make the mechanism meaningless. One possible solution for this problem is to have the tracker detect the misbehavior of the irrational peers and isolate them by not informing them with the other peers existing in the network, or only inform them with the peers at the similar or lower job completedness.

References

[1] BitTorrent, "The official bittorrent home page," http://www.bittorrent.com.

[2] B. Cohen, "Incentives build robustness in bittorrent," in *Proceedings of Workshop on Economics of Peer-to-Peer Systems (P2PEcon'03)*, Berkeley, CA, USA, June, 2003.

[3] S. Saroiu, P. K. Gummadi and S. D. Gribble, "A measurement study of peer-to-peer file sharing systems," in *Proceedings of Multimedia Computing and Networking (MMCN'02)*, January, 2002.

[4] M. Izal, G. Urvoy-Keller, E.W. Biersack, P.A. Felber, A. Al Hamra and L. Garcés-Erice, "Dissecting bittorrent: five months in a torrent's lifetime," in *Proceedings of Passive & Active Measurement Workshop (PAM'04)*, France, April, 2004.

[5] J. Pouwelse, P. Garbacki, D. Epema and H. Sips, "The bittorrent p2p file-sharing system: Measurements and analysis," in *Proceedings of International Workshop on Peer-to-Peer Systems (IPTPS'05)*, Ithaca, New York, Cornell University, February, 2005.

[6] A. R. Bharambe, C. Herley, and V. N. Padmanabhan, "Analyzing and improving bittorrent performance," Technical Report, MSR-TR-2005-03, Microsoft Research, 2005.

[7] K. Ramachandran and B. Sikdar, "An analytic framework for modeling peer to peer networks," in *Proceedings of IEEE INFOCOM 2005*, Miami, FL, USA, March, 2005.

[8] Z. Ge, D. R. Figueiredo, S. Jaiswal, J. Kurose, and D. Towsley, "Modeling peer-peer file sharing systems," in *Proceedings of IEEE INFOCOM 2003*, San Francisco, USA, March, 2003.

[9] X. Yang and G. de Veciana, "Service capacity of peer to peer networks," in *Proceedings of IEEE INFOCOM 2004*, Hong Kong, China, March, 2004.

[10] D. Qiu and R. Srikant, "Modeling and performance analysis of bittorrent-like peer-to-peer networks," in *Proceedings of ACM SIGCOMM 2004*, Portland, OR, USA, August, 2004.

[11] E. Adar and B. A. Huberman, "Free riding on gnutella," Technical Report, CSL-00-3, Xerox PARC, 2000.

[12] K. S. Trivedi, *Probability and Statistics with Reliability, Queuing, and Computer Science Applications*, John Wiley and Sons, New York, 2002.

[13] F. M. Cuenca-Acuna, R. P. Martin, and T. D. Nguyen, "Autonomous replication for high availability in unstructured p2p systems," in *Proceedings of Symposium on Reliable Distributed Systems (SRDS'03)*, Florence, Italy, October, 2003.

[14] R. Sherwood, R. Braud, and B. Bhattacharjee, "Slurpie: A cooperative bulk data transfer protocol," in *Proceedings of IEEE INFOCOM 2004*, Hong Kong, China, March, 2004.

[15] C. Gkantsidis, P. Rodriguez, "Network coding for large scale content distribution," in *Proceedings of IEEE INFOCOM 2005*, Miami, FL, USA, March, 2005.

[16] A. Parker, "The true picture of peer-to-peer file-sharing," CacheLogic Presentation, July, 2004.

[17] S. G. M. Koo, C. S. G. Lee, and K. Kannan, "A genetic-algorithm-based neighbor-selection strategy for hybrid peer-to-peer Networks," in *Proceedings of IEEE International Conference on Computer Communications and Networks (ICCCN'04)*, Chicago, IL, USA, October, 2004.

[18] R. Bhagwan, S. Savagen and G. Voelker, "Understanding availability," in *Proceedings of 2nd International Workshop on Peer-to-Peer Systems (IPTPS'03)*, Berkeley, CA, February 2003.