

A Multiphased Approach for Modeling and Analysis of the BitTorrent Protocol

Vivek Rai, Swaminathan Sivasubramanian, Sandjai Bhulai, Pawel Garbacki*, Maarten van Steen

Vrije Universiteit

Amsterdam

The Netherlands

Delft University*

Delft

The Netherlands

Email: {vivekr, swami, sbhulai, steen}@few.vu.nl, p.j.garbacki@tudelft.nl

Abstract

BitTorrent is one of the most popular protocols for content distribution and accounts for more than 15% of the total Internet traffic. In this paper, we present an analytical model of the protocol. Our work differs from previous works as it models the BitTorrent protocol specifically and not as a general file-swarming protocol. In our study, we observe that to accurately model the download process of a BitTorrent client, we need to split this process into three phases. We validate our model using simulations and real-world traces. Using this model, we study the efficiency of the protocol based on various protocol-specific parameters such as the maximum number of connections and the peer set size. Furthermore, we study the relationship between changes in the system parameters and the stability of the protocol. Our model suggests that the stability of BitTorrent protocol depends heavily on the number of pieces a file is divided into and the arrival rate of clients to the network.

1. Introduction

BitTorrent is an example of a self-scalable P2P protocol that is used for efficient content distribution [4]. The fundamental idea behind the protocol is to divide a file into pieces and to make the downloaders of that file forward (i.e., upload) pieces to each other. By sharing the cost of uploading among downloaders, the protocol allows hosting of a file to become affordable. In principle, when there are N downloaders, the bandwidth of the outgoing link at a host need only facilitate a single complete download, instead of N file downloads as would be normally the case.

BitTorrent achieves fairness among peers by using a “tit-for-tat” mechanism, by which a peer uploads a piece of a file to another downloading peer only if the latter has a new piece to offer. The BitTorrent protocol uses *decentralized* decision making for selecting which peer to trade pieces with (peer selection) and which pieces to trade for (piece selection).

Despite the popularity of BitTorrent for distributing con-

tent across a large number of users, relatively few attempts have been made to analytically model the BitTorrent protocol. Earlier research on modeling peer-to-peer content distribution systems assumed that each peer has global knowledge of other peers and the status of their downloads [12, 9]. However, in the BitTorrent protocol, each peer makes its own decision regarding the peer and pieces to select based on a limited view of the complete network. Hence, assuming global knowledge is unrealistic and does not really help in understanding the system dynamics and the performance of the protocol.

In this paper, we provide an analytical model with less restrictive assumptions that models the BitTorrent protocol better. To this end, we first analyze the real world traces of the download process of BitTorrent clients. Our measurement analysis reveals that the evolution of the download process of a BitTorrent client can be split into three phases: the *bootstrap*, the *efficient download*, and the *last download* phase. To accurately model this download process, we model each of these phases separately. We observe that our model can correctly capture the behavior of real-world BitTorrent clients. An important distinction with previous works is that we validate our model using real-world traces. To the best of our knowledge, we are the first to do so. Using our model, we also analyze the behavior of the protocol for different protocol related parameters such as the maximum number of connections and the peer set size.

Our model also allows us to analyze the *stability* of the protocol. A notable work on analyzing stability of a similar system is done in [8], where the authors analyze the stability of *coupon replication systems*, which approximate the behavior of BitTorrent. We analyze the stability of the BitTorrent protocol itself. In our study, we observe that stability of the BitTorrent protocol depends heavily on the number of pieces a file is divided into and the arrival rate of peers to the network. We show how the system can reach an unstable state in realistic scenarios.

The contribution of this paper is threefold. First, we identify that in order to accurately model the BitTorrent pro-

tol, we need to divide the evolution of the download process of a BitTorrent client into three phases. Second, we are among the first to provide an analytical model of the BitTorrent protocol that captures all phases and validate them with real-world traces and simulations. Third, we provide an accurate analysis of the efficiency and stability properties of the actual BitTorrent system in contrast to other research that has provided analysis of systems that are only roughly similar to BitTorrent.

The rest of the paper is structured as follows. Section 2 presents the protocol background and related work. Section 3 presents our multiphased download model and Section 4 presents the experimental validation of the model. Section 5 discusses the impact of the number of connections on system efficiency and Section 6 discusses the stability of the protocol. Section 7 discusses the open issues and the insights we have gained during our analysis. Section 8 concludes the paper.

2. Background

2.1. Algorithms and Mechanisms

The data distribution model of BitTorrent is based on the *file swarming* paradigm which assumes that the file is sliced into pieces that are distributed independently of each other [8]. Swarming allows peers to start providing data immediately after they have downloaded the first piece in contrast to alternative download protocols where sharing is possible only after obtaining the complete file.

The key decision points that influence the download efficiency of the BitTorrent protocol are: (i) which peer to trade your pieces with (peer selection strategy) and (ii) which pieces to trade for (piece selection strategy). In BitTorrent, these decisions are performed in a decentralized fashion by each peer based on a limited view of the network. To enforce fairness in sharing, BitTorrent adopts the *tit-for-tat* mechanism which ensures that each peer contributes (by uploading pieces to other peers) proportionally to how much it receives (by downloading pieces from other peers).

Before analyzing the impact of BitTorrent mechanisms on the download process, we first outline the basic terminology that will be used throughout the paper.

- **Pieces and blocks:** In BitTorrent, each file is divided into pieces, usually of size 256 KBs, which are further split into blocks of a default size of 16 KBs. Therefore, a block is a basic transmission unit in the system. However, a peer can start serving a block only after the entire piece is received and its correctness is verified through a hash function.
- **Leechers and seeds:** Peers in a BitTorrent swarm are divided in two categories: leechers and seeders. A *leecher* or *downloader* is a peer with a download in

progress. A peer that has acquired the complete file and still chooses to participate in the swarm is termed as a *seed*.

- **Neighbor set (NS):** Each peer maintains information on the pieces possessed by a limited number of peers, which is referred to as its *neighbor set*. The neighbor set is a symmetric relation meaning that if peer *A* is in the neighbor set of peer *B*, then peer *B* is in the neighbor set of peer *A*. Initially, the neighbor set contains a list of random peers obtained from the tracker while joining the swarm. The updates of the neighbor set occur during successive, periodic contacts with the tracker and when a peer is added to the neighbor set of another peer. Whenever a peer obtains a new piece, it informs everyone in its neighbor set.
- **Potential set:** The potential set of a peer consists of the subset of peers in its NS that have at least one piece to trade with the peer at a given instance of time. For example, if *B* is present in the NS of *A* and *B* has one piece that *A* does not have (and vice versa) at a given instant, then *B* belongs to *A*'s potential set (and vice versa).
- **Peer selection strategy:** In order to exchange content, a peer contacts a member of its neighbor set. We refer to this process of contacting neighbors as an *encounter*. The policy guiding the selection of peers to encounter is called *peer selection strategy*. In the BitTorrent protocol, the peer selection strategy is implemented by the *choking* algorithm that prefers peers with the highest upload rates.
- **Piece selection strategy:** After a successful encounter, the peer needs to decide which pieces to download. To this end, BitTorrent uses two *piece selection strategies*: (i) *random piece first* – the piece is selected randomly and (ii) *rarest piece first* – the piece held by the fewest number of neighbors is selected for download.

2.2. Related Work

In the recent years, there have been some notable works that have proposed analytical models for general file-swarming systems and studied their efficiency and/or stability characteristics [12, 9, 8, 2, 1, 10]. Before providing a detailed review of these works, we first clarify two important performance-related metrics. **Efficiency** is defined as the fraction of peer upload bandwidth utilized for content distribution (i.e., to upload content). **Stability** is defined as the existence of a steady state. In the context of BitTorrent, this means that the arrival rate of peers should be balanced by their departure rate.

Fluid models have been used in [12], [9], and [11], to study the performance of file-swarming networks. The general notion of fluid models is to study the aggregate behavior of the system. In the file-swarming context, fluid models discard protocol level details and study the system using aggregate parameters such as utilization of the peer upload bandwidth, the peer abort rate, the average number of seeders, etc. In [12], the authors use a fluid model to study the efficiency of file-swarming networks under two types of peer arrival patterns: (i) in a flash crowd phase where peers arrive simultaneously in a single burst and (ii) in a steady-state phase where peers arrive as a Poisson stream. The authors conclude that during the flash crowd phase, the service capacity of the network scales logarithmically with the number of peers, and during the steady state phase the capacity scales in proportion to the number of peers. In [9], the authors use a fluid model to study the stability of a BitTorrent network. The authors consider two scenarios: first, when the download bandwidth is the bottleneck for the service capacity and second, when the upload bandwidth is the bottleneck for the service capacity. The authors conclude that in both situations, the BitTorrent network is stable in the neighborhood of an equilibrium point. In [11], the authors further develop the stability results introduced in [9] to account for the heterogeneous peer bandwidths. However, the fundamental limitation of fluid models is that they hide protocol dynamics and instead rely on specific input parameters, which are not trivial to obtain. In contrast, our objective in this paper is to study the impact of protocol design on the performance of the system. Therefore, we consider models which study the system parameters in detail.

In [8], the authors model the impact of protocol dynamics on the performance of a file-swarming network, which they call a *coupon replication system*. The authors show that the efficiency of the coupon replication system depends only on the number of pieces a file is split into. Furthermore, the authors show that stability depends critically on the exogenous arrival rate of the pieces. They prove that the system is unstable when the arrival rate of a single piece is greater than the sum of the arrival rates of other pieces. However, the fundamental limitation of their work is that the design of the coupon replication system is considerably different from BitTorrent, and hence their resultant dynamics is also different. In BitTorrent, a peer maintains a neighbor set and encounters only within this limited view of the network. In the coupon replication system, a peer randomly selects its encounters from the entire swarm, and hence there is a positive probability of failed encounters if peers do not have pieces to trade. Furthermore, the coupon replication system utilizes only a single connection for an encounter, whereas, as we show later in the paper, the number of connections has a significant impact on the efficiency of BitTorrent. Other details, like the impact of tit-for-tat

mechanism, the download process dynamics, and the evolution of the system entropy, are also missing from the work in [8].

In [5], the authors propose the use of network coding for content distribution. This technique enables each peer in the swarm to generate and transmit a dynamically created piece that is significantly different from other simultaneously transmitted pieces. This improves the utilization of the upload bandwidth at the peers and the entropy of the swarm. As shown in their work, network coding is particularly useful when the network connectivity among peers is poor and the degree of outgoing connections of a peer is low.

The impact of fairness on the efficiency of the system is discussed in [2]. The authors argue that the unchoking mechanisms utilized in BitTorrent may lead to unfairness and propose a bandwidth matching mechanism to improve fairness which may decrease the efficiency. In this paper, we consider a strict tit-for-tat upload policy and homogeneous bandwidth settings and hence their results are not directly applicable to our study. However, we plan to address this issue in the future. In [1], the authors study the feasibility of BitTorrent style systems to stream content over the network and suggest that BitTorrent can be effective for streaming content provided proper upload scheduling policies are used.

3. Multiphased Download Evolution Model

In this section, we model the evolution of the peer download process as a Markov chain. For this purpose, we assume that (i) peers exchange pieces with others only in a strict tit-for-tat fashion and (ii) peers have homogeneous bandwidth connections. Even though these assumptions may seem limiting, our model captures the evolution of the download process of a BitTorrent client with reasonable accuracy. In particular, the model captures three phases in the peer download evolution that occur in practice. We discuss the impact of the two assumptions and describe how to extend our model in Section 7.

3.1. Model Details

The evolution of the download process of a single peer in BitTorrent can be described by a three-dimensional Markov chain. The state space is defined by the triplet (n, b, i) ; where n is the number of active connections, b is the number of downloaded pieces, and i is the size of the potential set. We assume that a peer joins the system without any piece. Therefore, its initial state is $(0, 0, 0)$. In addition, we assume that a peer exits the system immediately after downloading all B pieces. Therefore, the process is absorbed in state $(0, B, 0)$.

The transition probability of moving from state (n, b, i) to state (n', b', i') can be split into several functions f, g ,

and h . These functions represent the probability of making the transition to b' , i' , and n' , respectively. Thus, the transition probabilities are given as follows.

$$\begin{aligned} \Pr\{(n, b, i) \rightarrow (n', b', i')\} \\ = f(b' | n, b) \times g(i' | n, b, i) \times h(n' | n, b, i'), \end{aligned}$$

for $0 \leq n, n' \leq k$, $0 \leq b, b' \leq B$, and $0 \leq i, i' \leq s$. Note that this also reflects the order in which the variables in the process are updated: first the number of downloaded pieces b is updated, then the potential set size i , and finally the number of connections n (which depends on i'). Recall that in the equation above, B represents the number of pieces to be downloaded, k the maximum number of simultaneous uploads, and s the maximum achievable size of the neighbor set.

When a peer joins the system, it acquires its first piece either through seeds or through optimistic unchoking from other downloaders. Therefore, the process moves to a state with $b' = 1$ when $b = 0$. However, the download progress for subsequent pieces depends on the number of active connections n that was established previously. Thus, the number of downloaded pieces in the next state increases by n such that the total number of downloaded pieces does not exceed B , i.e., the total file size. Hence, the state transitions in b , determined by the function f , are given by

$$f(b' | n, b) = \begin{cases} 1, & b = 0, b' = 1, \\ 1, & b \geq 1, b' = \min\{b + n, B\}, \\ 0, & \text{otherwise.} \end{cases}$$

The evolution of the potential set is determined by the number of successful connections that are set up with peers in the neighbor set. There are various factors on which this number depends.

First, upon joining the system, thus with $b + n = 0$, the peer tries to setup a connection with each of the s peers in the neighbor set with success probability p_{init} . The number of successful connections can then be modeled as a binomially distributed random variable X_1 with parameters s and p_{init} .

As explained, after this transition, the peer will acquire its first piece, resulting in $b + n = 1$. However, if the first piece is not tradable with anyone in the neighbor set (i.e., the potential set size is 0), then the peer has to wait until a new peer with exchangeable pieces enters the neighbor set. We represent the probability that this happens by the parameter α .

The potential set size is determined by the instantaneous trading power of a peer P , which depends on the number of complete pieces $b + n \geq 1$ that P has when $i > 0$. We consider two cases: (1) other peers who already have at least $j > b + n$ pieces, and (2) those who have equal or less.

For the first group we need to determine the probability that a peer Q has at least one piece to exchange with P . Q has nothing to exchange if all of P 's $b + n$ pieces are already stored at Q . This probability is equal to $\binom{j}{b+n} / \binom{B}{b+n}$. In other words, the probability that P has something to exchange with an arbitrary node with at least $b + n$ pieces, is equal to

$$\sum_{j=b+n+1}^B \varphi(j) \left[1 - \binom{j}{b+n} / \binom{B}{b+n} \right],$$

with φ the probability distribution function of the different pieces, i.e., $\varphi(j)$ is the fraction of peers having j pieces. We return to this distribution in Section 6.

For a peer Q with $j \leq b + n$ pieces, P will not be able to exchange pieces if all of Q 's j pieces are already stored at P . This probability is equal to $\binom{b+n}{j} / \binom{B}{j}$, so that the probability that P will be able to exchange pieces is

$$\sum_{j=1}^{b+n} \varphi(j) \left[1 - \binom{b+n}{j} / \binom{B}{j} \right].$$

As a consequence, the probability $p_{(b+n)}$ that a randomly selected peer has a piece to exchange with a peer having $b + n$ pieces is given by

$$\begin{aligned} p_{(b+n)} = & \sum_{j=b+n+1}^B \varphi(j) \left[1 - \binom{j}{b+n} / \binom{B}{b+n} \right] + \\ & \sum_{j=1}^{b+n} \varphi(j) \left[1 - \binom{b+n}{j} / \binom{B}{j} \right]. \end{aligned} \quad (1)$$

Hence, in this case, the number of successful connections is given by a binomially distributed random variable X_2 with parameters s and $p_{(b+n)}$.

Finally, it could happen that a peer P has pieces to exchange, i.e., $b + n > 1$, but that the potential set size i drops to 0. This typically occurs when P has acquired a lot of pieces, so that finding another peer Q to trade pieces with becomes much harder. In this case, P will have to wait for new pieces to flow into the neighbor set. We model the probability of this event by the parameter γ .

The evolution of the potential set, determined by the

function g , can now be given as:

$$g(i' | n, b, i) = \begin{cases} \Pr(X_1 = m), & b + n = 0, i' = m, \\ \alpha, & b + n = 1, i = 0, i' = 1, \\ 1 - \alpha, & b + n = 1, i = 0, i' = 0, \\ \Pr(X_2 = m), & b + n \geq 1, i > 0, i' = m, \\ \gamma, & b + n > 1, i = 0, i' = 1, \\ 1 - \gamma, & b + n > 1, i = 0, i' = 0, \\ 1, & b = B, i' = 0, \\ 0, & \text{otherwise,} \end{cases} \quad (2)$$

Let us now look at the transitions in the number of active connections. These transitions depend on the re-encounter probability p_r , that an established connection does not fail, and on the probability p_n , that a new connection is established. Clearly, upon entry, when $b + n = 0$, the peer cannot establish new connections, since it has no pieces to exchange. When $b + n \geq 1$, the number of active connections due to re-encounters is a binomially distributed random variable Y_1 with parameters n and p_r . Since the potential set size has grown to i' , there can be at maximum $\min\{i', k\}$ connections of which $\max\{\min\{i', k\} - n, 0\}$ are new. Thus, the number of new connections is a binomially distributed variable Y_2 with parameters $\max\{\min\{i', k\} - n, 0\}$ and p_n . Therefore, the transitions in the number of active connections, determined by the function h , is given by

$$h(n' | n, b, i') = \begin{cases} 1, & b + n = 0, n' = 0, \\ \Pr(Y_1 + Y_2 = m), & b + n \geq 1, n' = m, \\ 1, & b = B, n' = 0, \\ 0, & \text{otherwise.} \end{cases} \quad (3)$$

3.2. Multiphased Evolution

The evolution of the download progress in BitTorrent can be divided into three critical phases: the bootstrap, the efficient download, and the last download phase.

Bootstrap phase In this phase, a peer acquires its first complete piece and is then ready to trade it with the members of its neighbor set. However, if the first piece is not tradable with anyone in the neighbor set (i.e., the potential set size is 0), a peer waits until a new peer with exchangeable pieces enters the neighbor set. In our model, the bootstrap phase is reflected by the transitions from $(0, 0, 0)$ to $(0, 1, i)$.

If $i = 0$, the Markov process has a self-transition in the state $(0, 1, 0)$ with probability $1 - \alpha$ and a transition to the state $(0, 1, 1)$ with probability α . The parameter α is equal

to $\frac{\lambda ws}{N}$, where λ is the arrival rate of peers, s is the neighbor set size, w is the probability that a newly arriving peer has a piece to exchange, and N is the number of peers in the swarm.

Otherwise, if $i > 0$, the process makes transitions out of the bootstrap phase into the trading phase, where the download rate depends on the efficiency of the BitTorrent mechanisms rather than the altruistic behavior of other peers.

Efficient download phase In this phase, there is always someone in the neighbor set to trade pieces with (i.e., the potential set size is always greater than one). The instantaneous size of the potential set constrains the maximum number of connections a peer can establish. In our model, this is demonstrated in Equation (3), where $Y_1 + Y_2$ (the total number of connections) is constrained by $\min(i, k)$.

The potential set size is determined by the instantaneous trading power of a peer, which depends on the number of completed pieces that a peer has. From Equation (2), the binomial variable X_2 represents the distribution of the potential set with parameters s and $p_{(b+n)}$. The parameter $p_{(b+n)}$ represents the probability that a randomly selected peer has a piece to exchange with a peer having $(b + n)$ pieces. Note that the probability $p_{(b+n)}$ increases from 0.5 for $b + n = 1$ to its maximum at $b + n = B/2$, and decreases to 0.5 for $b + n = B - 1$. Therefore, on average more than half the peers in the neighbor set are ready to exchange content. However, once the potential set size falls to 0, the process makes a transition to the last download phase. Otherwise, the process is absorbed in state $(0, B, 0)$.

Last download phase This download phase occurs when the potential set size falls to 0. In this phase, the download rate depends on γ , the rate at which new pieces flow into the neighbor set. The process makes transitions from $(0, b, 0)$ to $(0, b + 1, 0)$ with probability γ and self-transitions with probability $1 - \gamma$. The process is finally absorbed in the state $(0, B, 0)$.

4. Model Validation

In this section, we present the experimental validation of our model. We validate our multiphased download model using two methods. First, we simulate the BitTorrent protocol and measure the impact of the neighbor set size on the evolution of the potential set and hence the evolution of the download rate. As a next step, we collect real-world traces of the download process of a BitTorrent client for different kinds of swarms and validate these real-world measurements.

4.1. Validation with Simulations

We implemented a discrete-event simulator in C++ that simulates a BitTorrent swarm. In our simulator, peers ar-

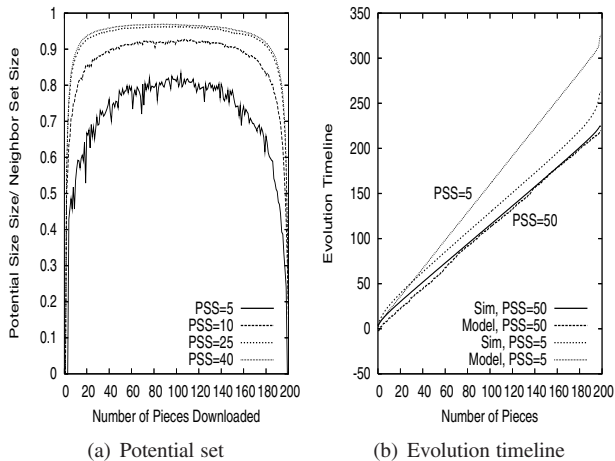


Figure 1. Effect of the peer set size on the download process.

rive to the swarm according to a Poisson process with a certain rate and depart as soon as they have downloaded all the pieces. The number of pieces in a file (B), the maximum number of connections for a peer (k), the peer set size (s), and the time to download a piece are built as configurable parameters.

We validate the model by comparing the download process described by the model with the one obtained from the simulation. As previously discussed in our model, we can identify three phases in the download process. In Figure 1 we observe that the bootstrap and the last download phase do occur when the peer set size is small (both in the simulation and the model). The reason for this is that there is a high probability to lose a significant fraction of the neighbors leading to a small potential set size. The download rate depends highly on the potential set size, as the peers in this set are the ones with whom pieces are exchanged. Hence, the potential set evolution and the download rate are highly correlated. In the efficient download phase the graphs for a peer set size of 5 do not match as accurately as for higher peer set sizes. The gap between the two plots, obtained from the simulation and the model for a peer set size of 5, can be explained as follows. The function $p_{(b+n)}$, which determines the size of the peer set in the efficient download phase, is the dominant factor in determining the duration of the efficient download phase. It is not trivial to deduce an exact expression for this function from the data. The function in our model serves as a first approximation, and exhibits the presence of the bootstrap and the last download phase, capturing the trends seen in the simulation results. As can be seen from the figure, the model validates the results with a high accuracy for higher values of the peer set size. In reality, BitTorrent clients do have peer set sizes in

the range of 40–70. This leads us to believe that our model is useful to study the BitTorrent protocol in many realistic scenarios.

4.2. Validation with Real-World Traces

In addition to evaluating our model through simulations, we have measured the real world BitTorrent swarms and used the collected data to validate our modeling approach. For the purpose of obtaining measurements we have extended a popular BitTorrent software, the BitTornado client¹, with the functionality required to collect detailed statistics on the download process. The modified BitTorrent client was injected into real-world BitTorrent swarms and actively participated in logging download progress information. Since our model assumes a strict tit-for-tat piece trading strategy, during the measurements we did not allow the modified client to interact with the seeds.

An important aspect of the measurement setup is the swarm selection criterion. To eliminate the possible influence of starvation inherent to small swarms on the obtained results, we have collected data for swarms of sizes varying from a few hundred to a few thousand peers. These swarms have been selected based on manual inspection of the statistics provided by the tracker. The tracker statistics consist of the number of peers involved in the download at a one hour resolution. We were, thus, able to filter out swarms in flash crowds (by observing rapidly increasing numbers of peers) as well as dying swarms and concentrate only on the stable ones.

In Figure 2, we plot the download rate evolution and the corresponding evolution of the potential set size for three different peers. We selected these plots to demonstrate three different instances of the download evolution process. Figures 2(a) and (b) depict a download instance without any predominant bootstrap and last download phase. As can be seen in Figure 2(b), the potential set size grows very fast in the beginning and remains greater than 15 throughout the download process, which implies that a peer always has more than $k = 7$ other peers to exchange the pieces with. This results in a smooth download rate from the beginning to the end as depicted in Figure 2(a). Figures 2(c) and (d) depict a download instance with a significant last download phase. This is because the potential set size, as depicted in Figure 2(d), drops to 1 towards the later stages of the download. Therefore, the download progress depends on whether a successful connection with this potential peer is established or some new peers join the potential set. Figures 2(e) and (f) depict a download instance where a peer is stuck within its bootstrap phase. This is because the potential set size as depicted in Figure 2(f), is equal to 0 during the initial part of the download process and hence the

¹<http://www.bittornado.com>.

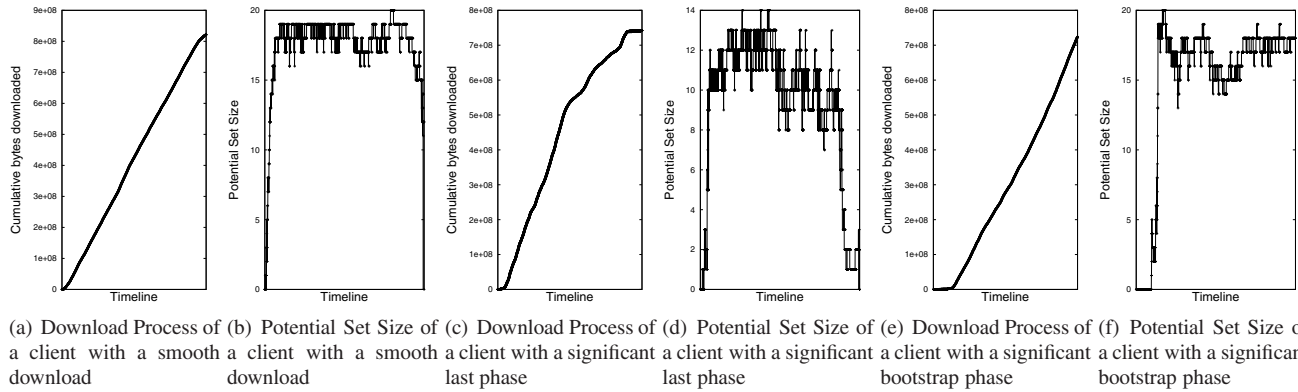


Figure 2. Plots of the download process and the evolution of potential set for different clients.

download rate remains 0 until the potential set size makes a transition out of state 0.

4.3. Conclusion

We have demonstrated that a typical peer download in a BitTorrent swarm evolves through three phases. The performance of each phase is determined by different protocol parameters and design strategies.

In the bootstrap phase, the performance is determined by two factors. First, the initial size of the potential set, i.e., the number of neighbors interested in exchanging pieces with a peer that has only one complete piece. If the initial potential set size is 0, a peer remains with this neighborhood until new peers arrive. Therefore, the design of the BitTorrent protocol should be such that the probability of remaining in the bootstrap phase is minimized. This can be accomplished either by choosing the size of the neighbor set sufficiently high or by intelligent construction of the neighbor set as described in [8]. In this paper, the authors suggest clustering of peers in terms of their download status such that the probability of successful encounters can be increased. However, feasible implementation of such a technique is still an open question and should be considered as future work. Second, the important factor that determines the performance in the bootstrap phase is the rate at which new peers are arriving into the neighbor set. To improve performance, the tracker can bias new peer arrivals into the neighborhood of the peers which are trapped in the bootstrap phase.

The second phase of the download evolution is the trading phase. Most of the pieces are downloaded in this phase. Therefore, the analysis of this phase is crucial in the study of the overall system efficiency. The potential set size within this phase is always greater than 0. In fact, as shown in Figure 1, for a suitably chosen neighbor set size, the fraction of neighbors in the potential set is close to 1. Therefore, the performance of the download process in this phase is determined by the number of active connections established with

the peers in the potential set. Thus, an important system parameter which determines the performance of the system is k (the maximum permissible number of simultaneous active connections). We will study the impact of k on the efficiency of the system in the next section.

A peer makes a transition to the last download phase with a certain probability. Once the peer enters this phase, the rate at which the download process evolves is determined by the rate at which new pieces are flowing into the neighbor set. The last download phase or the last piece problem has also been observed in the measurement study conducted in [7]. However, we are the first to model the protocol dynamics that creates such a phenomenon. Furthermore, we discuss the possible techniques that can be used to alleviate the last piece problem in Section 7.

5. Efficiency Model

In the previous section, we studied the download evolution model for a BitTorrent peer. In our model, the efficiency of the download process depends on the average utilization of connections. Therefore, if x_i represents the fraction of peers that have i active connections, for $i = 0, \dots, k$, the efficiency is defined by $\eta = (1/k) \sum_{i=1}^k i \cdot x_i$. In this section, we study the impact of k , the maximum permissible number of simultaneous connections, on the efficiency of the system.

The number of active connections at a peer evolves as a general birth/death process. The birth rates are determined by the success rate of new encounters, and the death rates are determined by the expected length of active connections. Therefore, to ensure a high download efficiency in a BitTorrent swarm, both the expected length of active connections and the success probability of new encounters should be high. The success probability depends on the fraction of peers that have at least one open connection, i.e., a connection that is not active. However, if the expected length of active connections is high, the fraction of open connections

will be low and hence the success probability will be low. Therefore, to study this intricate dynamics of connection establishment/failure, we establish a Markov chain model..

The state space of the Markov chain is given by $\{x_0, \dots, x_k\}$. Connection failures are modeled as transitions from state x_i to x_j for all $i > j$, with transition rates given by $w_{i-j}^i x_i$, where w_l^i represents the probability that l connections out of the i active connections fail. Therefore, w_l^i is given by the binomial probability $\binom{i}{l} (1 - p_r)^l p_r^{i-l}$, where p_r , as previously defined, represents the probability (averaged over all peers in the system) that an established encounter does not fail.

Connection establishments are modeled as transitions from state x_i to x_{i+1} for all $i < k$. This transition occurs when a peer from class i sets up a connection successfully with another peer j having at least one open connection, thus $j < k$. However, when $j = i - 1$, the peer in class i moves to class $i + 1$ and the peer in class $i - 1$ moves to class i , resulting in no effective change in the fraction x_i . Moreover, when $j = i$, both peers move from class i to class $i + 1$. Hence, the transition rates can be written as $(1 - x_{i-1} + x_i - x_k)x_i$.

This type of Markov chains are typical for modeling migration processes. It is well known that a finite-state Markov chain which is unichain and a-periodic has a unique steady-state (or equilibrium) distribution [3]. Even though our Markov chain satisfies the unichain property and is a-periodic, the lemma does not directly apply. In our model, we have that $x_i \in [0, 1]$, thus we have a compact state space instead of a finite state space. However, let N be the number of peers in the system, then $\tilde{x}_i = x_i N \in \{0, \dots, N\}$, and hence the lemma applies to the process \tilde{x}_i . Consequently, the lemma applies to the original process as well, since there is a one-to-one mapping between \tilde{x} and x .

The equilibrium distribution of the Markov chain can be obtained from the solution of the so-called system of balance equations. The balance equations describe the evolution of the system in steady state. For the downward transitions, the fraction x_i decreases because of transitions out of state x_i due to failures in x_i , and increases because of transitions into state x_i due to failures in x_j for all $j > i$. The change in the first case is given by $x_i \sum_{l=1}^i w_l^i$, whereas the latter is given by $\sum_{l=i+1}^k w_{l-i}^l x_l$. Hence, the evolution for the downward transitions is given by

$$x_i = x_i - x_i \sum_{l=1}^i w_l^i + \sum_{l=i+1}^k w_{l-i}^l x_l. \quad (4)$$

For the upward transitions, the fractions x_i are updated in increasing order so that we update x_0 first, followed by x_1, x_2 , etc. When peer i connects to a peer from class $l < k$ (which occurs with probability x_l), the peer from class i moves to class $i + 1$, and the peer from class l moves to

class $l + 1$. Thus, when $l = i - 1$ the number of peers in class i remains the same, and when $l = i$ two peers leave class i . Therefore, the net change in the fraction x_i is given by $(x_i \cdot N - 1 + \mathbf{1}_{\{l=i-1\}} - \mathbf{1}_{\{l=i\}})/N$. Hence, the total change is given by

$$\begin{aligned} x_i &= \sum_{l=0}^{k-1} x_l \frac{x_i \cdot N - 1 + \mathbf{1}_{\{l=i-1\}} - \mathbf{1}_{\{l=i\}}}{N} + x_k x_i \\ &= \sum_{l=0}^k x_l \frac{x_i \cdot N - 1 + \mathbf{1}_{\{l=i-1\}} - \mathbf{1}_{\{l=i\}} + \mathbf{1}_{\{l=k\}}}{N}. \end{aligned} \quad (5)$$

The changes for peers j with $i \neq j < k$ are not captured by Equation (5). Changes occur only when peer i connects to peers in class $j - 1$ and class j . In the first case, when $i \neq j - 1$, only one peer enters class j , whereas two peers enter in case $i = j - 1$. In the second case, a peer leaves class j . Thus, the net change in the fraction x_j is given by $(x_j \cdot N + \mathbf{1}_{\{l=j-1\}} + \mathbf{1}_{\{l=i\}} - \mathbf{1}_{\{l=j\}})/N$. Hence, the total change is given by

$$x_j = \sum_{l=0}^k x_l \frac{x_j \cdot N + \mathbf{1}_{\{l=j-1\}} + \mathbf{1}_{\{l=i\}} - \mathbf{1}_{\{l=j\}}}{N}, \quad (6)$$

for $j \neq i$ and $j < k$. Note that the value of x_k remains the same, since peers in this class do not have open connections.

The formulas given by Equations (5)–(6) form the system of balance equations. Due to the complex state-dependent transition rates, closed-form expressions for the equilibrium distribution are hard to obtain. Alternatively, by iterating this set of equations, the state of the system converges to the steady-state distribution [3]. Note that the order in which we iterate the equations provides an upper bound to the efficiency η . Since we start with the classes with the least number of connections first, more peers are able to migrate to classes with a high number of connections.

Figure 4(a) depicts the impact of k on the efficiency of the system. As we mentioned before, the order in which we iterate the equations in the model gives an upper bound on the efficiency. Therefore, the model overestimates the simulation results by over 8% for $k = 1$. However, as k increases, the relative difference between model and simulation results is less than 1%. Moreover, both the model and simulation results provide conclusive evidence that the efficiency of the system increases significantly by increasing k from 1 to 2. Further, increasing the value of k beyond 2 does not bring any significant improvement in the efficiency of the system. This result can be explained as follows. For $k = 1$, the duration of a connection is determined by the number of exchangeable pieces at the start of the connection. However, for $k > 2$, peers maintain multiple simul-

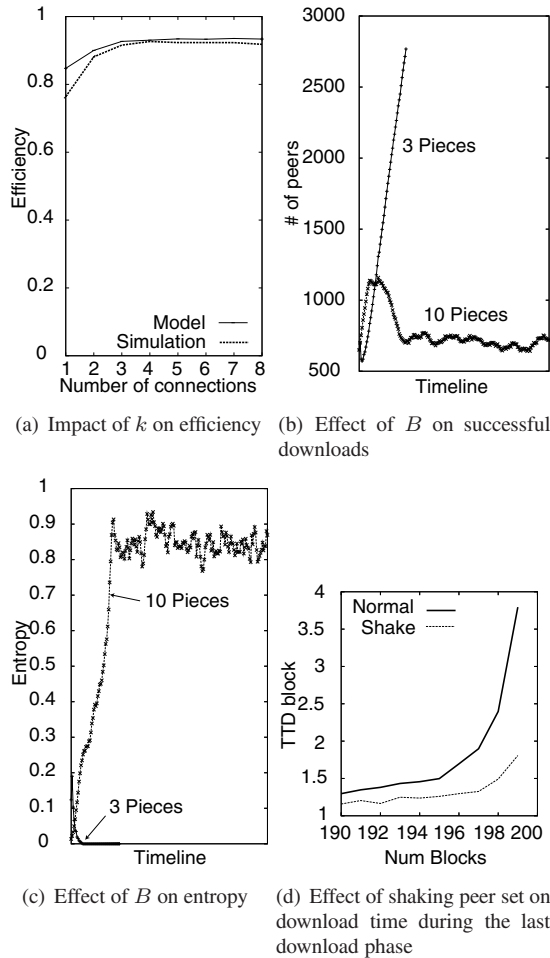


Figure 3. Effects of model parameters on the download process.

taneous connections. Therefore, new pieces are simultaneously arriving at the peers, which can also be exchanged. Thus, the expected duration of connections increases significantly by increasing k from 1 to 2. Longer duration of established connections implies low re-encounter probabilities, and hence a high efficiency of the system.

6. Stability

In this section we discuss the stability of the BitTorrent protocol, where stability is defined in terms of the entropy of the system. When d_i denotes the replication degree of the i^{th} piece in the system, then the entropy E is given by

$$E = \frac{\min\{d_1, \dots, d_B\}}{\max\{d_1, \dots, d_B\}}.$$

The entropy is a measure of the skewness of the piece distribution in the system. We call the system stable if the long-run behavior of the system is such that the entropy E goes

to 1. Otherwise, if the entropy goes to 0, the skewness of the pieces prohibits the progress of peers resulting in large download times leading to instability. We use our download evolution model and simulation experiments to study the impact of system design parameters on the stability. As discussed in the previous sections, the download process is divided into three phases. We show that each phase has an influence on the stability of the system.

In the *bootstrap phase*, the parameter α determines the probability that the potential set remains empty. The smaller the entropy E , the smaller the probability α becomes, due to the fact that the skewness in the system gives a higher probability of obtaining more replicated pieces. This leads to longer expected sojourn times in the bootstrap phase, since on average the time that a peer remains in this phase is $1/\alpha$. If the arrival rate is high enough, the skewness increases since the new peers will also encounter more replicated pieces with a higher probability. Hence, the skewness deteriorates the stability of the system.

In the *trading phase*, the BitTorrent protocol tries to stabilize the system compensating for any initial skewness in the bootstrap phase. The dynamics of the protocol is such that the least replicated pieces are exchanged at a faster rate than the more replicated pieces. Provided that the number of pieces B is large enough, this creates a drift of the entropy E towards 1 again, leading to φ being a uniform distribution. In other words, the fraction of peers having j pieces is the same for all j . When the number of pieces B is too small, peers leave the system too quickly and decrease the number of less replicated pieces. Consequently, when B is too small, the expected time peers remain in the trading phase is not sufficiently long to push the entropy back to 1.

In the *last download phase* the parameter γ affects the rate at which new peers enter the potential set when it is empty. When γ decreases, peers with almost all pieces remain longer in the system, since the expected time in the last download phase is $1/\gamma$. This improves stability of the system in the same way as explained in the efficient download phase. Hence, smaller values of γ provide a larger drift of the entropy E towards one.

In Figure 4(b) and (c), we show using simulations the effect of B on the number of peers in the system and the entropy when starting from an initial state with a high skewness. As explained in the discussion, when B is too small the system does not have sufficient time to reach stability again. The figures reflect this insight, since the number of peers grows very large when $B = 3$, whereas stability is achieved when $B = 10$. Similarly, the entropy goes to 0 in the former case, whereas the latter case pushes the entropy back to one.

An exact analysis of the stability of the BitTorrent protocol capturing the stability behavior (and its effects on all the parameters) is a nontrivial problem and is left for

future work. The difficulty arises from the complex dependence between the parameters (that affect each other) at each transition in our model. This intricate interplay requires transient methods to deal with the nonstationary state-dependent behavior of the parameters. For an overview of transient methods we refer to [6].

7. Discussion and Future Work

This section contains the lesson we learn from the analysis and models discussed in this paper.

7.1. The Last Piece Problem

In our study, we recognize that the last piece problem occurs when the pieces missing at a peer are not held by any other peer within its neighbor set. An obvious solution to this problem is to download the final pieces from seeds (as they do not enforce the tit-for-tat mechanism). However, in the event of retaining a strict tit-for-tat trading mechanism, we observe that continuous randomization of the neighbor set can alleviate the last piece problem. To demonstrate this, we ran a simulation experiment with the following modification to the BitTorrent algorithm: when a peer completes 90% of its pieces, it removes all its neighbors in its current peer set and gets a new (randomly chosen) set of peers from the tracker for populating its peer set. We call this process as *shaking the peer set*. We evaluated the impact of this modification to the traditional BitTorrent setup and plotted the download time for just the last few pieces for the sake of clarity. The results of our experiment are given in Figure 4(d). As seen in the figure, shaking the peer set significantly reduces the download time for the last few pieces. Even though this is a simple study, we believe this is a promising step to address the last piece problem.

7.2. Effects of Seeding

A seed is a peer that has acquired a complete file and still chooses to participate in the swarm. Furthermore, since seeds do not enforce the tit-for-tat piece trading downloaders can get new pieces for free. Previous models, [12] and [9], incorporate the impact of seeding by assuming seeds to be a central piece distribution source with the capacity of the source scaled by the number of seeds. In our model, we can incorporate the effects of seeds by modeling extra connections, which do not require the strict tit-for-tat policy. However, several advanced seeding techniques such as super-seeding² have been proposed and we plan to study seeding as a separate work in future.

8. Conclusion and Future Work

In this paper, we presented an analytical model of the BitTorrent protocol. In our study, we observe that to accurately model the download process of a BitTorrent client,

we need to split the download process into three phases. Our work differs from the previous works mainly in two aspects. First, we model the BitTorrent protocol specifically and not as a general file-swarming protocol. Second, we validate our model using simulations and real-world BitTorrent client traces. Our experiments show that our model validates the real-world behavior with reasonable accuracy. Using this model, we studied the efficiency of the protocol based on various protocol-specific parameters such as the maximum number of connections and the peer set size. Our analysis shows that the gain in system efficiency rapidly decreases beyond two connections. Furthermore, we studied the relationship between changes in the system parameters and the stability of the protocol. Our model suggests that the stability of the BitTorrent protocol depends heavily on the number of pieces a file is divided into and the arrival rate of clients to the network.

As an immediate next step, we plan to include the impact of seeding behavior in our model. Furthermore, we would like to do an exact analysis of the overall protocol stability including transient effects for different protocol specific parameters.

References

- [1] D. Arthur and R. Panigrahy. Analyzing BitTorrent and related peer-to-peer networks. In *SODA '06*, pages 961–969, New York, NY, USA, 2006. ACM Press.
- [2] A. Bharambe, C. Herley, and V. Padmanabhan. Analyzing and improving BitTorrent performance. MSR-TR-2005-03, Microsoft Research, Redmond, WA, USA.
- [3] K. Chung. *Markov Chains with Stationary Transition Probabilities*. Springer-Verlag, 1967.
- [4] B. Cohen. Incentives build robustness in BitTorrent. Proc. of Workshop on Economics of Peer-to-Peer Systems, 2003.
- [5] C. Gkantsidis and P. Rodriguez. Network coding for large scale content distribution. In *Proc. of IEEE Infocom*, 2005.
- [6] A. Ingolfsson, E. Akhmetshina, S. Budge, Y. Li, and X. Wu. A survey and experimental comparison of service level approximation methods for non-stationary M/M/s queueing systems. *INFORMS Journal of Computing*, 2005.
- [7] A. Legout, G. Urvoy-Keller, and P. Michiardi. Rarest first and choke algorithms are enough. under submission.
- [8] L. Massoulie and M. Vojnovic. Coupon replication systems. In *SIGMETRICS '05*, pages 2–13, New York, NY, USA, 2005. ACM Press.
- [9] D. Qiu and R. Srikant. Modeling and performance analysis of BitTorrent-like peer-to-peer networks. *SIGCOMM Comput. Commun. Rev.*, 34(4):367–378, October 2004.
- [10] Y. Tian, D. Wu, and K. Ng. Modeling, analysis and improvement for BitTorrent like file sharing networks. In *Proceedings of IEEE Infocom*, 2006.
- [11] F. Venot-Perronnin, P. Nain, and K. Ross. Multiclass P2P networks: static resource allocation for service differentiation and bandwidth diversity. *Perf. Eval.*, 62(1-4):32–49, 2005.
- [12] X. Yang and G. de Veciana. Service capacity of peer to peer networks. In *Proceedings of IEEE Infocom*, 2004.

²<http://en.wikipedia.org/wiki/Superseeding>.