An Analytical Model of a BitTorrent Peer

Mario Barbera, Alfio Lombardo, Giovanni Schembra and Mirco Tribastone
DIIT - University of Catania
Email: (mbarbera, alombardo, schembra, mtriba)@diit.unict.it

***Abstract*— In the last years peer-to-Peer (P2P) applications are emerging as potentially important structures for information, content management and distribution. The most popular P2P application in the Internet today is file sharing, which was initially popularized by the Napster system. One of the most used P2P network today allowing file sharing for Internet users is BitTorrent, which generates more than 53% of the P2P traffic in the Internet. Up to now very little efforts have been dedicated to develop analytical tools for performance evaluation of a peer in a BT network. The target of this paper is to propose an analytical model of a peer, called Tagged Peer (TP), with its peer set.**

I. Introduction

P2P users cooperate through an overlay network that allows applications to share resources by acting as both clients and servers ([1]). The P2P paradigm can be applied, for example, to access distributed file systems, to exploit distributed processing facilities, and to share stored information. Nevertheless, the most popular P2P application in the Internet today is file sharing. This kind of P2P application was popularized by the Napster system mostly for sharing music, video or software files [2]. Even though many P2P file-sharing systems have been proposed and implemented since Napster (e.g., [3], [4]), few of them have stood the test of intensive daily use by a very large user community. BitTorrent [5] is today one of the larger P2P networks allowing file sharing for Internet users. It satisfies the main requirements of a P2P network [1], which are high availability, high probability of receiving a good version of the requested content (no fake files), efficiency in dealing with flash crowds, and a relatively high download speed. Thanks to these characteristics, BitTorrent traffic accounted for 53% of all P2P traffic in the last year [6], that is, more than 40% of the traffic on high-speed IP backbones [7], [8].

The definition of analytical frameworks to evaluate and optimize the performance of both P2P protocols and underlying transport networks is therefore a challenging issue for the management of Internet resources in the near future. Very little effort has been dedicated to the analysis of P2P systems up to now. The closed queueing system model introduced in [9] was among the first attempts to provide basic insights into the stationary performance of a P2P system. In [10] two models were proposed for the analysis of both the steady-state performance and the transient characteristics of a P2P file sharing system. In [11] a stochastic fluid model is used to study the performance of P2P web caches (SQUIRREL). Starting from [12] and [10], [13] proposes for the first time an early model of a BitTorrent network.

All the previous models tend to capture the behavior of a global P2P network. However, in a BT network, peers only share their resources with the peers of a limited portion of the network, the so-called peer set. This therefore represents the communication environment in which a BitTorrent peer lives. Thus, modeling this limited region of the P2P network is very important to better analyze the performance of a single peer, and then to derive more specific information about the influence of parameter values and implemented algorithms on the performance of a single peer. The seminal work on modeling the behavior of a peer in a BT network was proposed by the authors in [14], where a Markov model is used to evaluate the behavior of free-rider peers, which are those that do not contribute to the service capacity of the network, because they download from other peers while not providing their contents. However, and fortunately, free riders constitute only a very little portion of the number of peers in a BT network. So it is crucial to have a model of cooperative peers. The target of this paper is to propose an analytical model of a cooperative peer, called Tagged Peer (TP), by means of state change of its connectios to the peer set.

The paper is structured as follows. Section II provides a brief overview of BitTorrent, to introduce only those concepts which are fundamental to better understand the model. Section III defines the analytical model of a generic peer. Section IV, after evaluating the convergence speed of an iterative algorithm needed to solve the model, proposes a case study where the obtained performance are analyzed and discussed. Finally, Section V concludes the paper.

II. Overview of BitTorrent

In this section we provide a brief introduction to BitTorrent, focusing on the most crucial aspects that are necessary for better comprehension of the model we will describe in the following sections. The interested reader is referred to Cohen's web-site [15] for a more in-depth discussion of the BitTorrent protocol.

BitTorrent (BT) is a *file level* peer-to-peer system. Users participating in such a network are not provided with file searching mechanisms. They are assumed to know the resource they wish to obtain *a priori*, by means of out-of-band communications, such as, WWW or instant messaging services. In order to download a particular resource, each user has to get a file called the *torrent file*. It is usually a small file (if compared to the resource file size) which only contains the meta-information needed to enter the BT network related to that file. In other words, we can say that each torrent file is univocally related to a BT overlay network where peers share a single resource (called the *torrent*).

The most important field in a torrent file is the *tracker* URL. The tracker is the central component of the BitTorrent architecture. It is an HTTP-based service which "tracks" the peer activities on the network. Though it can handle multiple torrents simultaneously, for our purposes we will see it as univocally associated to one torrent. A user who wants to "join the torrent" (i.e. he wants to download a file), contacts the tracker. It records the new peer identity and assigns a randomly-selected list of peers in the network, which we refer to as the *peer set*. The peer set size is an application parameter, and it can be set *only before* entering the network. Once the peer set is received, the peer establishes a peer-to-peer TCP connection with each of them. The same procedure has to be followed by the original publisher of the file, who has to transmit his identity to the tracker in order to be contacted by the other peers.

In a BT network we can distinguish between two classes of peers. Using the BT protocol author's terminology, *seeds* are peers who have a complete copy of the resource. They remain in the network only to provide the system with service capabilities. *Leechers*, on the other hand, are peers who are currently downloading the torrent. Far from being only consumers of the service, leechers actually contribute to diffusion of the torrent by uploading pieces of a file they have already obtained. We refer to these pieces as *chunks*.

Peers can easily negotiate the chunks to transfer with appropriate protocol messages. The original torrent seeder partitions the resource into chunks of sizes typically ranging between 32 KB and 512 KB. The resulting chunk index is published in the torrent file; moreover, to guarantee error protection, an SHA1 hash is associated with the content of each chunk. When a BT connection is established, there is a handshaking phase in which a peer transmits a bitfield representing the chunks it already has. When a peer finishes downloading a new chunk, it transmits a `have` message to its peer set in order to notify remote peers of its availability. Thanks to these strategies, each peer always has a consistent view of the status of its peer set.

Each peer regularly contacts the tracker and reports its status to it. More specifically, when a peer finishes downloading, it transmits a `complete` event: as a consequence, the tracker will return a peer set only made up of leechers, since there is no interest in maintaining connection with remote seeds.

While it does not strictly belong to the BT protocol, the *unchoking algorithm* is essential for the system. Each peer is able to download from any remote peer, but it can upload to a limited maximum number of leechers (typically 4–5: we call this value N_{up}). This limitation is justified by the author with the TCP congestion control [5]. Every ten seconds, each peer performs the unchoking algorithm, which ranks the remote leechers on the basis of their upload rate. It then selects the best $N_{up} - 1$ uploaders in its peer set. Clearly, in order to be unchoked, it is necessary for them to be *interested*, i.e. they should not have at least one chunk in common with the peer. This algorithm actually implements a tit-for-tattish strategy, and it represents the core of the entire BT system.

The remaining upload connection is reserved for *optimistic unchoking*: a leecher is randomly selected irrespective of its upload rate, and it remains unchoked for 30 seconds. The purpose of the optimistic unchoking strategy is twofold. On the one hand, it is necessary to allow new peers to download their first chunks, since they cannot just provide upload. On the other hand, it lets each peer to explore its peer set, in order to find better peers to download from.

Seeds also perform unchoking algorithms. In this case, the selection metric is based on the downloading speed of the remote leechers, since they are assumed to be quicker in diffusing the torrent.

Putting it all together, each leecher-to-leecher connection can be logically represented with four Boolean state variables. For the sake of clarity, let *A* be the peer we are focusing on (the *tagged peer*), and *B* be a remote leecher. We can define these variables:

- **choking**: *B* cannot download from *A*.
- **choked**: *A* cannot download from *B*.
- **interested**: *A* is interested in *B*'s chunks.
- **interesting**: *B* is interested in *A*'s chunks.

The state of a leecher-to-seed connection is greatly simplified. Data transfer is unidirectional; moreover, a leecher is always interested in a seed's chunks. So this kind of connection is completely represented by the state of the *choked* variable.

In order to improve system performance, BT employs other useful techniques. The *rarest first* policy suggests that peers download the rarest chunks in their peer set. In this way, a homogeneous diffusion of chunks is ensured. It minimizes the likelihood of not being able to complete the download due to the absence of a peer who had rare chunks. Moreover, it increases the probability of a peer being interesting in its peer set, thus capitalizing its upload capacity.

A transfer data unit is not a chunk. In order to be easily encapsulated into TCP messages, each chunk is further divided into sub-pieces of 16 KB. BitTorrent protocol messages actually deal with this unity. The *strict priority* suggests that peers first complete all the sub-pieces of a chunk, before starting to download a new one. So each peer can announce its availability more rapidly, thus letting remote peers obtain it in advance.

III. The Model

Our aim is to develop an analytical model of a leecher in a BitTorrent network; we will refer to the leecher as the *tagged peer* (TP). The state of a cooperative leecher is completely defined by the state of each peer-to-peer connection established and the nature of the remote peers.

However, before describing the model, it is worthwhile outlining some assumptions we made to derive it.

A. Assumptions and notation

Assumption 1: Though a connection state is defined by the value of four Variables, as discussed previously, in this model we disregard the *interesting* and *interested* ones, i.e. we will assume that, at any point in time, these variables are always set to `true`. Of course this is just an ideal case, but a real system

TABLE I
MODEL PARAMETERS

λ_L	Leecher arrival rate
λ_S	Seed arrival rate
θ^{-1}	Leecher mean lifetime
γ^{-1}	Seed mean lifetime
c	Download bandwidth
μ	Upload bandwidth
N_{up}	Maximum upload connections simultaneously allowed
N	Maximum peer set size
α	Unchoking algorithm frequency
β	Optimistic unchoking frequency

will tend towards this behavior if the rarest first policy works well. As a consequence, a connection is completely defined by the values of the *choking* and *choked* variables: when a connection is not choked, download can take place.

Assumption 2: The maximum peer set size (here indicated as N) is a user-level application parameter. Here, we assume that all peers in a torrent use the same value for it.

Assumption 3: The upload and download bandwidth, μ and c respectively, are equal for all the peers.

Assumption 4: The arrival process of both seeds and leechers is Poisson-distributed, with rates of λ_S and λ_L, respectively.

Assumption 5: The lifetimes of seeds and leechers are exponentially distributed with mean times of $1/\gamma$ and $1/\theta$.

Assumption 6: Peers are statistically independent of each other.

Assumption 7: All peers perform the unchoking algorithm at the same frequency α; optimistic unchoking takes place at a rate of β.

Other hypotheses are needed to deal with seeds. They actually do not have the same statistical behavior as leechers, since they construct their peer set in a different manner and base their unchoking algorithm on another metric. While an appropriate model of a seed should be adopted, we propose a simpler approach:

Assumption 8: Each seed is always connected to N leechers.

Assumption 9: Each seed always uploads to N_{up} leechers.

The model parameters introduced in this section are summarized in Table I.

B. Space State Representation

In order to model the behavior of the leecher indicated as the TP, we define a continuous-time Markov process as follows. The state S of the TP at any time t is completed represented by the 8-tuple:

$$\underline{S} = \left(n^{cds}, n^{ods}, n^{tds}, n^{cdl}, n^{odl}, n^{tdl}, n^{oul}, n^{tul}\right) \quad (1)$$

Here, n^{chs} denotes the number of choked download connections with remote seeds. Both the state variables n^{ods} and n^{tds} represent the number of unchoked connections with seeds. We need to introduce two different variables because there are two different transition rates. In fact, n^{ods} is the number of optimistically unchoked download connections, and each of these connections could vary at a rate of β. Instead, n^{tds} is the number of download connections available with the unchoking algorithm, which is executed at a rate of α. Similar considerations hold for the state variables $n^{cdl}, n^{odl}, n^{tdl}$, which, however, refer to connections between the TP and the other leechers.

Finally, in order to fully define the state of the TP-leecher, we need two additional variables, which represent the state of the TP's upload connections: n^{oul} denotes the number of optimistically unchoked connections, while n^{tul} is the number of connections unchoked at a rate of α.

According to these definitions, it is straightforward to obtain the total number of leechers and seeds connected to the TP at any time instant t:

$$N_L(t) = n^{cdl}(t) + n^{odl}(t) + n^{tdl}(t) \quad (2)$$

$$N_S(t) = n^{cds}(t) + n^{ods}(t) + n^{tds}(t) \quad (3)$$

It is worthwhile observing that n^{cdl}, n^{odl}, n^{tdl} and n^{cds}, n^{ods}, n^{tds} *completely* cover the possible states of one side of the connections between the TP and both leechers and seeds. The other side—the one directed from the TP to a remote leecher—is represented just by n^{oul} and n^{tul}. In fact, the number of choked upload connections, here denoted as n^{cul}, can easily be obtained by noting that, at any time instant, the number of upload and download connections must be equal:

$$n^{cdl}(t) + n^{odl}(t) + n^{tdl}(t) = n^{cul}(t) + n^{oul}(t) + n^{tul}(t) \quad (4)$$

We thus obtain n^{cul} directly from the system state as defined above:

$$n^{cul}(t) = n^{cdl}(t) + n^{odl}(t) + n^{tdl}(t) - n^{oul}(t) - n^{tul}(t) \quad (5)$$

Finally, we note that the state space dimension is greatly reduced by the following constraints, which must be observed in order to guarantee consistency:

$$\begin{cases} n^{cds}(t) + n^{ods}(t) + n^{tds}(t) + \\ \quad + n^{cdl}(t) + n^{odl}(t) + n^{tdl}(t) \leq N \\ n^{tul} + n^{oul} \leq N_{up} \\ n^{oul} \in \{0, 1\} \\ n^{tul} + n^{oul} \leq n^{cdl} + n^{odl} + n^{tdl} \end{cases} \quad (6)$$

C. Premise: The Iterative Approach

Our target is to derive the infinitesimal generator $Q^{(\Sigma)}$, of the above Markov chain. We now introduce the probability distribution array, the generic element of which is:

$$\begin{gathered} \underline{\pi}_{\underline{i}} = P\left\{S(t) = \underline{i}\right\}, \\ \forall \underline{i} = (n^{cds}, n^{ods}, n^{tds}, n^{cdl}, n^{odl}, n^{tdl}, n^{oul}, n^{tul}) \in \Sigma \end{gathered} \quad (7)$$

Once $Q^{(\Sigma)}$ has been calculated, it is possible to obtain the steady-state probability array as follows:

$$\underline{\pi} Q^{(\Sigma)} = \underline{0} \qquad \text{s.t.} \qquad \sum_{\underline{i} \in \Sigma} \underline{\pi}_{\underline{i}} = 1 \quad (8)$$

While it should represent our solution, we need this array just to derive the model. We circumvent this problem by means of an iterative procedure.

Let $\underline{\pi}^0$ be an arbitrary initial probability distribution. With this we are able to generate the infinitesimal generator $Q_0^{(\Sigma)}$, so as to obtain a new steady-state probability array $\underline{\pi}^1$. This procedure is iterated until the norm difference between two successive probability distributions $\|\underline{\pi}^i - \underline{\pi}^{i-1}\|$ is below a given threshold ϵ. Clearly, in order to avoid algorithm stagnation, the procedure can be stopped when the maximum number of iterations is reached.

D. Model Description

We now derive the model by examining all the possible transitions from an arbitrary state $s_1 = \{n_1^{cds}, n_1^{ods}, n_1^{tds}, n_1^{cdl}, n_1^{odl}, n_1^{tdl}, n_1^{oul}, n_1^{tul}\}$ to a state $s_2 = \{n_2^{cds}, n_2^{ods}, n_2^{rdl}, n_2^{cdl}, n_2^{odl}, n_2^{tdl}, n_2^{oul}, n_2^{tul}\}$. After giving an introductory description of each event, we show the rate at which they occur. For the sake of conciseness, we decided to emphasize only the state variables subject to changes. For example, the full transition

$$(n^{cds}, n^{cds}, n^{ods}, n^{tds}, n^{cdl}, n^{odl}, n^{tdl}, n^{oul}, n^{tul}) \rightarrow (n^{cds}+1, n^{cds}, n^{ods}, n^{tds}, n^{cdl}, n^{odl}, n^{tdl}, n^{oul}, n^{tul})$$

will be simply reduced to

$$n_2^{cds} = n_1^{cds} + 1$$

1) A Leecher Enters the Peer Set: We assume that the resulting connection is initially choked by both sides. The related state transition is therefore:

$$n_2^{cdl} = n_1^{cdl} + 1 \quad \text{at rate} \quad \lambda_L \tag{9}$$

2) A Seed Enters the Peer Set: The connection is initially choked by the seed, and the state transition is:

$$n_2^{cds} = n_1^{cds} + 1 \quad \text{at rate} \quad \lambda_S \tag{10}$$

3) A Seed Leaves the Peer Set: The original connection state of the outgoing seed is assumed to be evenly distributed over the whole space of connections with seeds. We thus obtain the following three possible state transitions:

$$\begin{array}{lll} n_2^{cds} = n_1^{cds} - 1 & \text{at rate} & N_{S1} \cdot \gamma \cdot p_{cds} \\ n_2^{ods} = n_1^{ods} - 1 & " & N_{S1} \cdot \gamma \cdot p_{ods} \\ n_2^{tds} = n_1^{tds} - 1 & " & N_{S1} \cdot \gamma \cdot p_{tds} \end{array} \tag{11}$$

where N_{S1} is the total number of seeds at the generic instant t_1 (see Eq. 3); p_{cds}, p_{ods} and p_{tds} are the probabilities of the initial connection state. They can be obtained as:

$$p_{cds} = n_1^{cds}/N_{S1} \tag{12}$$

$$p_{ods} = n_1^{ods}/N_{S1} \tag{13}$$

$$p_{tds} = n_1^{tds}/N_{S1} \tag{14}$$

4) A Leecher Aborts its Download: This event leads to 9 possible state transitions, according to the connection state at the starting time instant. Unlike the previous event, we have to consider the states of both sides of the connection. Assuming that the states of each side are independent, we easily obtain:

$$\begin{array}{lll} n_2^{cdl} = n_1^{cdl} - 1 & \text{at rate} & \gamma N_{L1} p_{cdl} p_{cul} \\ n_2^{odl} = n_1^{odl} - 1 & " & \gamma N_{L1} p_{odl} p_{cul} \\ n_2^{tdl} = n_1^{tdl} - 1 & " & \gamma N_{L1} p_{tdl} p_{cul} \\ n_2^{cdl} = n_1^{cdl} - 1, n_2^{tul} = n_1^{tul} - 1 & " & \gamma N_{L1} p_{cdl} p_{tul} \\ n_2^{odl} = n_1^{odl} - 1, n_2^{tul} = n_1^{tul} - 1 & " & \gamma N_{L1} p_{odl} p_{tul} \\ n_2^{tdl} = n_1^{tdl} - 1, n_2^{tul} = n_1^{tul} - 1 & " & \gamma N_{L1} p_{tdl} p_{tul} \\ n_2^{cdl} = n_1^{cdl} - 1, n_2^{oul} = n_1^{oul} - 1 & " & \gamma N_{L1} p_{cdl} p_{oul} \\ n_2^{odl} = n_1^{odl} - 1, n_2^{oul} = n_1^{oul} - 1 & " & \gamma N_{L1} p_{odl} p_{oul} \\ n_2^{tdl} = n_1^{tdl} - 1, n_2^{oul} = n_1^{oul} - 1 & " & \gamma N_{L1} p_{tdl} p_{oul} \end{array} \tag{15}$$

where N_{L1} is the total number of leechers at the generic instant t_1 (see Eq. 2). The probabilities p_{cds}, p_{ods} and p_{tds} can be derived in the same manner as in (12)–(14):

$$p_{cdl} = n_1^{cdl}/N_{L1} \tag{16}$$

$$p_{odl} = n_1^{odl}/N_{L1} \tag{17}$$

$$p_{tdl} = n_1^{tdl}/N_{L1} \tag{18}$$

Finally, p_{cul}, p_{tul} and p_{oul} can obviously Be inferred from the following relation:

$$p_{cul} = 1 - (p_{tul} + p_{oul}) = 1 - \left(\frac{n_1^{tul}}{N_{L1}} + \frac{n_1^{oul}}{N_{L1}}\right) \tag{19}$$

5) A Leecher Finishes Download (and Becomes a Seed): Qiu and Srikant have proposed a fluid model which provides us with the rate of service in a BitTorrent network [13]. In particular, if the system has N_L leechers and N_S seeds, the transition rate of a leecher finishing download is regulated by the following expression:

$$r_{l \rightarrow s} = \min\{c \cdot N_L, \mu(\eta N_{L1} + N_S)\}. \tag{20}$$

where η denotes the contribution by leechers to the service capacity of the system. In our model, we obtain this transition rate by relating it to the steady-state values, $\overline{N}_L$ and $\overline{N}_S$, of leechers and seeds in the TP's peer set. These values are obviously obtained from the steady-state probability distribution $\underline{\pi}$ thanks to the iterative approach.

To derive the possible transition rates, we assume that the connection state with the remote peer is not modified by this event. So, taking account only of the change in the nature of

the remote peer, we have:

$$\begin{array}{lll}
n_2^{cdl} = n_1^{cdl} - 1, n_2^{cds} = n_1^{cds} + 1 & \text{at rate} & r_{l\to s} p_{cdl} p_{cul} \\
n_2^{odl} = n_1^{odl} - 1, n_2^{ods} = n_1^{ods} - 1 & \text{"} & r_{l\to s} p_{odl} p_{cul} \\
n_2^{tdl} = n_1^{tdl} - 1, n_2^{tds} = n_1^{tds} - 1 & \text{"} & r_{l\to s} p_{tdl} p_{cul} \\
n_2^{cdl} = n_1^{cdl} - 1, n_2^{tul} = n_1^{tul} - 1, & & \\
\quad n_2^{cds} = n_1^{cds} + 1 & \text{"} & r_{l\to s} p_{cdl} p_{tul} \\
n_2^{odl} = n_1^{odl} - 1, n_2^{tul} = n_1^{tul} - 1, & & \\
\quad n_2^{ods} = n_1^{ods} + 1 & \text{"} & r_{l\to s} p_{odl} p_{tul} \\
n_2^{tdl} = n_1^{tdl} - 1, n_2^{tul} = n_1^{tul} - 1, & & \\
\quad n_2^{tds} = n_1^{tds} + 1 & \text{"} & r_{l\to s} p_{tdl} p_{tul} \\
n_2^{cdl} = n_1^{cdl} - 1, n_2^{oul} = n_1^{oul} - 1, & & \\
\quad n_2^{cds} = n_1^{cds} + 1 & \text{"} & r_{l\to s} p_{cdl} p_{oul} \\
n_2^{odl} = n_1^{odl} - 1, n_2^{oul} = n_1^{oul} - 1, & & \\
\quad n_2^{ods} = n_1^{ods} + 1 & \text{"} & r_{l\to s} p_{odl} p_{oul} \\
n_2^{tdl} = n_1^{tdl} - 1, n_2^{oul} = n_1^{oul} - 1, & & \\
\quad n_2^{tds} = n_1^{tds} + 1 & \text{"} & r_{l\to s} p_{tdl} p_{oul}
\end{array} \tag{21}$$

6) A Seed Optimistically Unchokes the TP: By Assumption 8, the probability of being optimistically unchoked by a seed is $1/N$. The possible state transitions depend on the starting connection state. Indeed, as each seed individually performs this algorithm every $1/\beta$ seconds, we straightforwardly obtain the state transitions with the related frequencies:

$$\begin{array}{lll}
n_2^{cds} = n_1^{cds} - 1, n_2^{ods} = n_1^{ods} + 1 & \text{at rate} & n_2^{cds} \beta (1/N) p_{cds} \\
n_2^{tds} = n_1^{tds} - 1, n_2^{ods} = n_1^{ods} + 1 & \text{"} & n_1^{tds} \beta (1/N) p_{tds}
\end{array} \tag{22}$$

Observe that, with a probability of p_{ods} the initial connection state would already be optimistically unchoked; in this case, however, no transition occurs.

7) A Seed Chokes the TP (from Optimistic Unchoking): By Assumption 9, in order to remain unchoked, the TP has to compete against $N_{up} - 1$ other leechers, and to win at least once. Of the N_{up} leechers he is currently serving, each seed will unchoke the worst downloader. Assuming that the download bandwidth is always equally distributed among all download connections, the TP will be choked if it has a higher number of download connections than the other leechers. If we further assume that remote leechers statistically behave as if they were in a steady-state condition, we can obtain the success (p) and failure (q) probabilities in a competition with one leecher as:

$$p = P\{n_1^{down} \leq \overline{n}_{down}\} = F_{\overline{n}_{down}}\left(n_1^{down}\right) \tag{23}$$

$$q = 1 - p \tag{24}$$

where $n_1^{down} = n_1^{ods} + n_1^{tds} + n_1^{odl} + n_1^{tdl}$ is the total number of download connections at the time instant t_1, and $\overline{n}_{down}$ is the stochastic variable denoting the number of download connections in the steady state. $F_{\overline{n}_{down}}(\diamond)$ is the cumulative distribution function of the random variable $\overline{n}_{down}$.

As the TP competes with $N_{up} - 1$ leechers, according to Assumption 6, the probability of being choked is:

$$p_{bs} = q^{N_{up}-1} \tag{25}$$

The possible state transitions consist of both cases, i.e. when the TP is choked and when he is unchoked. In the latter case, he will continue to be "tested" at a different frequency. We have:

$$\begin{array}{lll}
n_2^{ods} = n_1^{ods} - 1, n_2^{cds} = n_1^{cds} + 1 & \text{at rate} & n_1^{ods} \beta p_{bs} \\
n_2^{ods} = n_1^{ods} - 1, n_2^{tds} = n_1^{tds} + 1 & \text{"} & n_1^{ods} \beta (1 - p_{bs})
\end{array} \tag{26}$$

8) A Seed Chokes the TP (by the Unchoking Algorithm): Similar considerations to those made for Event 7 still hold for this event. If the seed continues to serve the TP, no transition occurs; otherwise, the only possible state transition is:

$$n_2^{tds} = n_1^{tds} - 1, n_2^{cds} = n_1^{cds} + 1 \quad \text{at rate} \quad n_1^{tds} \alpha p_{bs} \tag{27}$$

where p_{bs} is exactly the same as in (25).

9) A Leecher Optimistically Unchokes the TP: This event is very similar to Event 9. While in the previous case the sample space size was known a priori, in this event the number of leechers among which the TP is selected depends on the steady-state condition. The possible state transitions are:

$$\begin{array}{lll}
n_2^{cdl} = n_1^{cdl} - 1, n_2^{odl} = n_1^{odl} + 1 & \text{at rate} & n_1^{cdl} \beta p_L \\
n_2^{tdl} = n_1^{tdl} - 1, n_2^{odl} = n_1^{odl} + 1 & \text{"} & n_1^{tdl} \beta p_L
\end{array} \tag{28}$$

where $p_L = 1/N_L$ is the probability of being optimistically unchoked by a leecher. N_L is as defined in 2.

10) A Leecher Chokes the TP (from Optimistic Unchoking): When a leecher performs its unchoking algorithm, he selects the best $N_{up}-1$ peers for unchoking among all those currently uploading to him. The probability of being unchoked is therefore related to the size of the sample space size from which the TP could be extracted; this space is represented by the steady-state value of the number of download connections.

The probability of success in competition against one leecher is related to the current number of upload connections: the more they are, the greater the probability of being choked, since the upload rate per connection is lower. If we define the success and failure probabilities of being unchoked as p_{suc} and p_{fai}, respectively, we obtain their value in the same manner as in Event 7:

$$p_{suc} = P\{n_1^{up} \leq \overline{n}_{up}\} = F_{\overline{n}_{up}}(n_1^{up}) \tag{29}$$

$$p_{fai} = 1 - p_{suc} \tag{30}$$

where $n_1^{up} = n_1^{oul} + n_1^{tul}$ represents the number of TP upload connections at the time instant t_1, while $\overline{n}_{up}$ is the steady-state random variable of such connections.

Let us observe that when the TP competes with i other leechers, he must win at least $i - (N_{up} - 2)$ times in order to be unchoked. Therefore, for a given i, this probability is p_i:

$$p_i = \sum_{k=i-(N_{up}-2)}^{i} \binom{i}{k} p_{suc}^k \cdot p_{fai}^{i-k} \tag{31}$$

Clearly, when the TP competes against less than $N_{up} - 1$ leechers, the probability of being unchoked is 1. Finally, the

probability of being unchoked, p_{ul}, is defined as;

$$\begin{aligned} & p_{ul} = P\{\text{TP is unchoked}\} = \\ & = \textstyle\sum_{i=0}^{N-1} P\{\text{TP is unchoked}|\text{TP vs. } i \text{ leechers}\}\cdot \\ & \cdot P\{\text{TP vs. } i \text{ leechers}\} = \\ & = P\{\text{TP vs. 0 leechers}\} + P\{\text{TP vs. 1 leecher}\}+ \\ & \ldots + P\{\text{TP vs. } N_{up}-2 \text{ leechers}\}+ \\ & + \textstyle\sum_{j=N_{up-1}}^{N-1} p_j \cdot P\{\text{TP vs. } j \text{ leechers}\} = \\ & = \textstyle\sum_{j=0}^{N_{up}-2} P\{\overline{n}_{down} = j\}+ \\ & + \textstyle\sum_{j=N_{up-1}}^{N-1} p_j \cdot P\{\overline{n}_{down} = j+1\} \end{aligned} \tag{32}$$

The probability of being choked is therefore:

$$p_{bl} = 1 - p_{ul} \tag{33}$$

Finally, the possible state variations are:

$$\begin{array}{ll} n_2^{odl} = n_1^{odl} - 1, n_2^{cdl} = n_1^{cdl} + 1 & \text{at rate} \quad n_1^{odl}\beta p_{bl} \\ n_2^{odl} = n_1^{odl} - 1, n_2^{tdl} = n_1^{tdl} + 1 & \text{"} \quad n_1^{odl}\beta(1-p_{bl}) \end{array} \tag{34}$$

11) A Leecher Chokes the TP (by the Unchoking Algorithm): We can follow the same considerations made in the previous Event. In this case, the state transition is:

$$n_2^{tdl} = n_1^{tdl} - 1, n_2^{cdl} = n_1^{cdl} + 1 \quad \text{at rate} \quad n_1^{tdl}\alpha p_{bl} \tag{35}$$

where p_{bl} is as defined in Eq. 33.

12) The TP Performs the Unchoking Algorithm: It is easy to obtain state transitions for this event. In fact, the TP always tends to saturate the available upload connections, regardless of the remote leecher's upload rate to him. Therefore, the final state for this transaction is:

$$\begin{array}{ll} n_2^{tul} = \min\left(N_{up}-1, N_{L1}-1\right), & \\ \quad n_2^{oul} = \min\left(1, N_{L1}\right) & \text{at rate} \quad \alpha \end{array} \tag{36}$$

Clearly, when N_{L1} is 0, $n_2^{tul} = 0$.

13) The TP Optimistically Unchokes a Connection: We have two different state transitions, according to the initial connection state:

$$\begin{array}{ll} n_2^{tul} = n_1^{tul} - 1, n_2^{oul} = 1 & \text{at rate} \quad \beta p_{tul} \\ n_2^{oul} = 1 & \text{"} \quad \beta p_{cul} \end{array} \tag{37}$$

E. The Generator Matrix

From analysis of the events shown above, it is possible to derive the entire infinitesimal generator. Each generic element is shown in Table II.

TABLE II
INFINITESIMAL GENERATOR GENERIC ELEMENT

State Transition	*Transition Rate*
λ_L	$n_2^{cdl} = n_1^{cdl} + 1$
λ_S	$n_2^{cds} = n_1^{cds} + 1$
$\gamma N_{S1} p_{cds}$	$n_2^{cds} = n_1^{cds} - 1$
$\gamma N_{S1} p_{ods}$	$n_2^{ods} = n_1^{ods} - 1$
$\gamma N_{S1} p_{tds}$	$n_2^{tds} = n_1^{tds} - 1$
$\theta N_{L1} p_{cdl} p_{cul}$	$n_2^{cdl} = n_1^{cdl} - 1$
$\theta N_{L1} p_{odl} p_{cul}$	$n_2^{odl} = n_1^{odl} - 1$
$\theta N_{L1} p_{tdl} p_{cul}$	$n_2^{tdl} = n_1^{tdl} - 1$
$\theta N_{L1} p_{cdl} p_{tul}$	$n_2^{cdl} = n_1^{cdl} - 1, n_2^{tul} = n_1^{tul} - 1$
$\theta N_{L1} p_{odl} p_{tul}$	$n_2^{odl} = n_1^{odl} - 1, n_2^{tul} = n_1^{tul} - 1$
$\theta N_{L1} p_{tdl} p_{tul}$	$n_2^{tdl} = n_1^{tdl} - 1, n_2^{tul} = n_1^{tul} - 1$
$\theta N_{L1} p_{cdl} p_{oul}$	$n_2^{cdl} = n_1^{cdl} - 1, n_2^{oul} = n_1^{oul} - 1$
$\theta N_{L1} p_{odl} p_{oul}$	$n_2^{odl} = n_1^{odl} - 1, n_2^{oul} = n_1^{oul} - 1$
$\theta N_{L1} p_{tdl} p_{oul}$	$n_2^{tdl} = n_1^{tdl} - 1, n_2^{oul} = n_1^{oul} - 1$
$r_{l\to s} p_{cdl} p_{cul}$	$n_2^{cdl} = n_1^{cdl} - 1, n_2^{cds} = n_1^{cds} + 1$
$r_{l\to s} p_{odl} p_{cul}$	$n_2^{odl} = n_1^{odl} - 1, n_2^{ods} = n_1^{ods} + 1$
$r_{l\to s} p_{tdl} p_{cul}$	$n_2^{tdl} = n_1^{tdl} - 1, n_2^{tds} = n_1^{tds} + 1$
$r_{l\to s} p_{cdl} p_{tul}$	$n_2^{cdl} = n_1^{cdl} - 1, n_2^{tul} = n_1^{tul} - 1, n_2^{cds} = n_1^{cds} + 1$
$r_{l\to s} p_{odl} p_{tul}$	$n_2^{odl} = n_1^{odl} - 1, n_2^{tul} = n_1^{tul} - 1, n_2^{ods} = n_1^{ods} + 1$
$r_{l\to s} p_{tdl} p_{tul}$	$n_2^{tdl} = n_1^{tdl} - 1, n_2^{tul} = n_1^{tul} - 1, n_2^{tds} = n_1^{tds} + 1$
$r_{l\to s} p_{cdl} p_{oul}$	$n_2^{cdl} = n_1^{cdl} - 1, n_2^{oul} = n_1^{oul} - 1, n_2^{cds} = n_1^{cds} + 1$
$r_{l\to s} p_{odl} p_{oul}$	$n_2^{odl} = n_1^{odl} - 1, n_2^{oul} = n_1^{oul} - 1, n_2^{ods} = n_1^{ods} + 1$
$r_{l\to s} p_{tdl} p_{oul}$	$n_2^{tdl} = n_1^{tdl} - 1, n_2^{oul} = n_1^{oul} - 1, n_2^{tds} = n_1^{tds} + 1$
$N_{S1}\beta 1/N p_{cds}$	$n_2^{cds} = n_1^{cds} - 1, n_2^{ods} = n_1^{ods} + 1$
$N_{S1}\beta 1/N p_{tds}$	$n_2^{tds} = n_1^{tds} - 1, n_2^{ods} = n_1^{ods} + 1$
$n_1^{ods}\beta p_{bs}$	$n_2^{ods} = n_1^{ods} - 1, n_2^{cds} = n_1^{cds} + 1$
$n_1^{ods}\beta(1-p_{bs})$	$n_2^{ods} = n_1^{ods} - 1, n_2^{tds} = n_1^{tds} + 1$
$n_1^{tds}\alpha p_{bs}$	$n_2^{tds} = n_1^{tds} - 1, n_2^{cds} = n_1^{cds} + 1$
$N_{L1}\beta p_{LP} p_{cdl}$	$n_2^{cdl} = n_1^{cdl} - 1, n_2^{odl} = n_1^{odl} + 1$
$N_{L1}\beta p_{LP} p_{tdl}$	$n_2^{tdl} = n_1^{tdl} - 1, n_2^{odl} = n_1^{odl} + 1$
$n_1^{tdl}\alpha p_{bl}$	$n_2^{tdl} = n_1^{tdl} - 1, n_2^{cdl} = n_1^{cdl} + 1$
$n_1^{odl}\beta p_{bl}$	$n_2^{odl} = n_1^{odl} - 1, n_2^{cdl} = n_1^{cdl} + 1$
$n_1^{odl}\beta(1-p_{bl})$	$n_2^{odl} = n_1^{odl} - 1, n_2^{tdl} = n_1^{tdl} + 1$

IV. MODEL EVALUATION

In this section we evaluate the model by carrying out a steady-state analysis of numerical results in a case study. First, in Section IV-A, we examine the efficacy of the iterative algorithm (in terms of stability and univocity guarantees); then, in Sections IV-B and IV-C, we give some insights into the composition of the TP's peer set and the capitalization of its resources.

A. The Effectiveness of the Iterative Algorithm

Before reporting numerical results, it is worth discussing about the effectiveness of the iterative approach used to derive the steady-state probability array of the Markov chain.

If the algorithm works, we will obtain the same steady-state distribution whatever the initial probability distribution. Therefore, in order to investigate the functioning of the iterative algorithm, we have considered an exemplary scenario with an arbitrary set of the model parameters.

To this aim, we compared three arbitrary initial probability distributions: uniform, binomial and exponential ones. The parameter set which we used is summarized in Table III. We achieved the same steady-state distribution in all cases. In Fig. 1 we plot, for each iteration i, the Euclidean norm difference between the probability distributions i and $i-1$; as we can observe, those distributions converged differently. More specifically, the fastest is the uniform distribution, which converges through 10 iterations; the slowest convergence,

TABLE III

PARAMETER SET USED FOR MODEL EVALUATION.

Parameter	*Value*
λ_L	0.008
λ_S	$5 \cdot 10^{-5}$
θ	0.0013
γ	0.0013
c	0.003
μ	0.00185
N_{up}	4
N	4
α	0.1
β	$\alpha/3$
ϵ	$5 \cdot 10^{-5}$

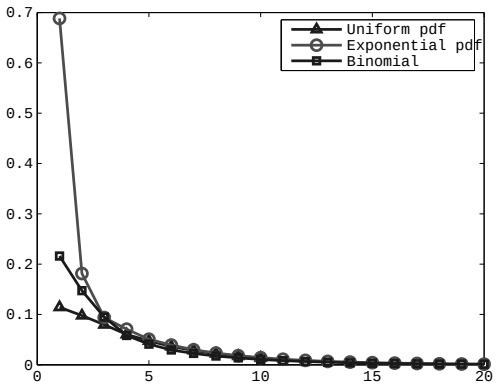

Fig. 1. Euclidean Norm Difference Used for the Iterative Algorithm.

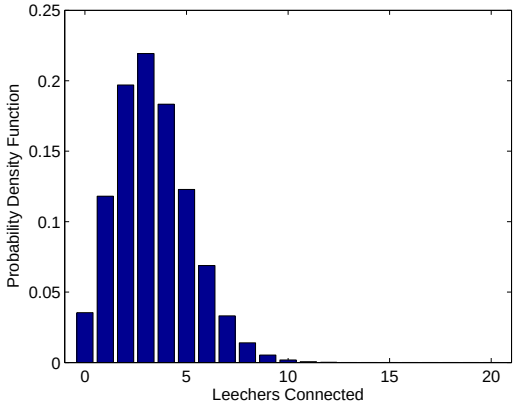

Fig. 2. Steady-state PDF of the Total Number of Leechers.

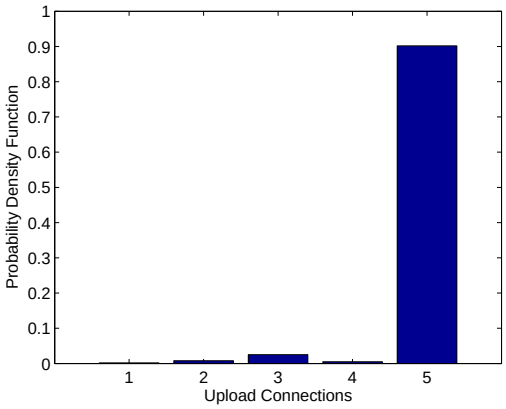

Fig. 3. Steady-state PDF of the Number of Upload Connections for $N = 20$.

instead, was obtained in 18 iterations with the exponential one. Finally, the algorithm converged in 16 iterations when it was launched with the binomial distribution.

Even though the shown results are not a formal demonstration of the algorithm convergence, many other initial distributions have been used in other tests, and we have always achieved convergence. These results made us be aware of the actual efficacy of the algorithm. However, due to its convergence rapidity, we decided to always use the uniform distribution as the initial one for the algorithm.

B. Remote Leechers Behavior Analysis

In order to evaluate the model, we used the same parameter set as it has been reported in Table III, except for N, which was set to 20 to fit a more realistic scenario.

Fig. 2 shows the steady-state probability density function of the total number of connected leechers. The expected value is 3.35 leechers. If compared to $N = 20$, this value could seem to be low; however, we have to consider the presence of seeds, and the arrival and departure rates of peers: a higher λ_L (or a lower θ) would have obviously increased that mean value.

Another important parameter for the description of the TP behavior is the number of upload connections to remote leechers. The related steady-state pdf is reported in Fig. 3. We note that the TP almost often saturates the maximum number of upload connections, since the expected value attested to 3.84. To further investigate this behavior, we can refer to Fig. 4, where we present the pdf of total leecher connections compared to the pdf of upload connections, for $N = 6$. This plot emphasizes that, even in presence of a small number of leechers, the number of upload connections still tends to the maximum available one. Finally, let us observe the prevalence of upload connections with respect to the number of leechers for a number of connections $N_L = 4$. This is due to the contribution of connections for all $N_L \geq 4$ in saturating N_{up}.

Finally, we discuss the contribution by remote leechers to the TP downloading activity. In Fig. 5, we plot the pdf of the steady-state number of optimistically unchoked connections with leechers, n^{odl}. The expected value is 0.66. To give further insights into this aspect, we calculate that mean value for N varying from 6 to 20 (see Fig. 6). We found that this value is quite independent from N: this fact can be easily explained

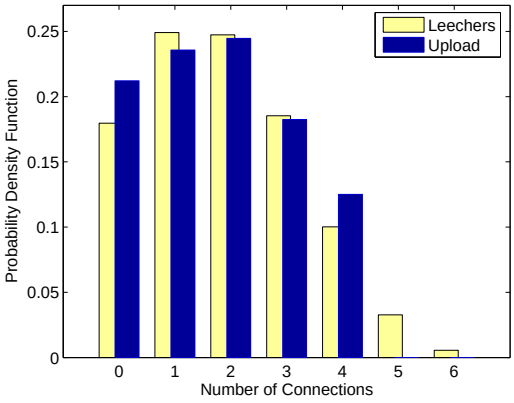

Fig. 4. Comparison between the Mean Number of Upload Connections and the Mean Number of Leechers for $N = 6$.

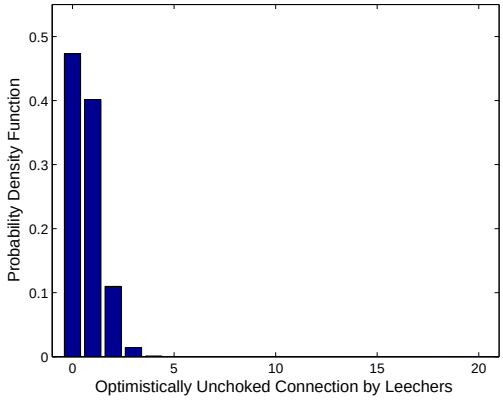

Fig. 5. Steady-state PDF of the n^{odl} State Variable.

by the increasing of this event rate and the simultaneous decreasing of the likelihood to be unchoked, since the sample space is just N.

The last contribution to the TP download activity is due to the number of download connections, represented by the state variable n^{tdl}. In this case, we found a steady-state mean value of 0.86. In Fig. 7, by means of a pie-chart, we illustrate the mean values of the number of connections to a leecher, for each possible state.

C. Seeds Behavior Analysis

Now we illustrate analogous results for connections with seeds, in the same case $N = 20$. Fig. 8 illustrates the pdf of the total number of seeds connected to the TP at the steady-state. The expected value is 6.61; so, considering that the mean number of connected leechers is 3.35 as observed so far, the mean peer-set size is only 9.96 in the case of $N = 20$. Same considerations obviously hold for the dependence of this value on the used parameter set.

As for leechers, we should obtain a value of n^{ods} *quasi* independent from N. In fact, we calculated the steady-state mean value of that state variable by varying N between 6

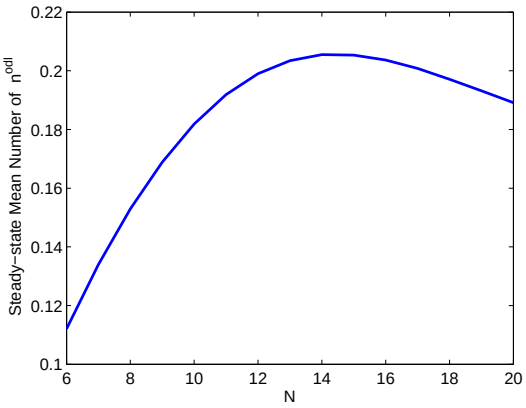

Fig. 6. Mean Value of n^{odl} for N varying.

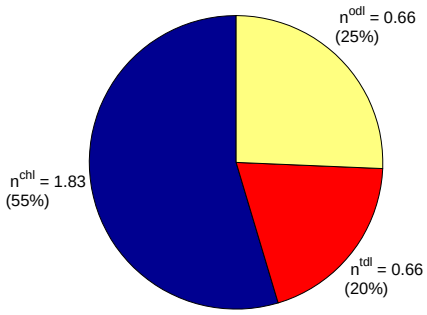

Fig. 7. Steady-state Mean Composition of Connection State with Leechers.

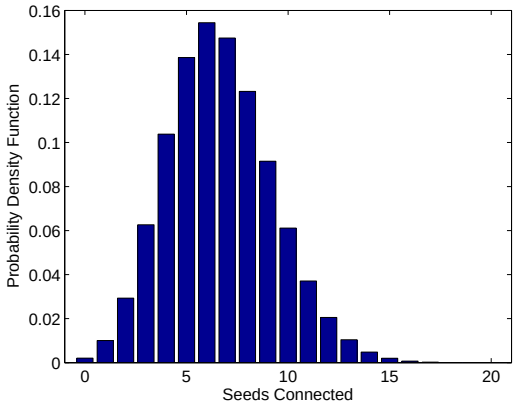

Fig. 8. Steady-state PDF of the Total Number of Seeds.

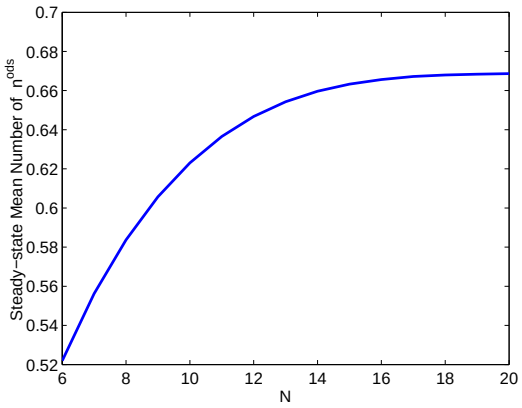

Fig. 9. Mean Value of n^{ods} for N varying.

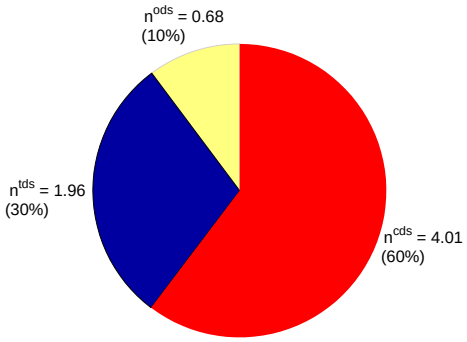

Fig. 10. Representation of All Possible Connection State with Seeds.

and 20. Fig. 9 plots these values. Again, we actually found that n^{ods} is quite constant. As for the n^{tds} state variable, we found a mean value of 1.96 connections. At last, in order to summarize these results, Fig. 10 shows a pie-chart of the number of connections to a seed, for each possible state.

V. Conclusion

The paper proposes a Markov model to evaluate performance of a peer of a BT network. This paper constitutes a complementary work to that proposed in [14], where the authors analyzed the behavior of a simpler peer, a free rider, who does not provide the other peers with upload connections, given that it does not cooperate to the evolution of the network. Instead, the model proposed in this paper takes into account this kind of connections too, and is able to evaluate performance of a cooperative peer. The model is applied to a case study, and important insights are derived on the BitTorrent protocols and algorithms. An aditional analysis is devoted to evaluate the iterative approach used to solve the model. Future works consist in apply this model in providing new insights to the BT protocols and algorithms, and to define a traffic model of P2P sources.

References

[1] D. S. Milojicic, V. Kalogeraki, R. Lukose, K. Nagaraja1, J. Pruyne, B. Richard, S. Rollins, and Z. Xu. Peer-to-peer computing. 2002.

[2] Napster. `http://www.napster.com`.

[3] FastTrack network. `http://www.kazaa.com`.

[4] Clip2. The gnutella protocol specification v0.4. Technical report. `http://www.clip2.com`.

[5] B. Cohen. Incentives build robustness in BitTorrent. 2003.

[6] A. Parker. The true picture of peer-to-peer file-sharing, 2004. `http://www.cachelogic.com`.

[7] S. Saroiu, K. Gummadi, and S.Gribble. A measurement study of peer-to-peer file sharing systems. In *Proc. Multimedia Computing and Networking (MMCN '02)*, San Jose, CA, 2002.

[8] A. Bellissimo, B.N. Levine, and P.Shenoy. Exploring the use of BitTorrent as the Basis for a large trace repository.

[9] Z. Ge, D. R. Figueiredo, S. Jaiswal, J. Kurose, and D. Towsley. Modeling peer-peer file sharing systems. In *Proceedings of IEEE INFOCOM'03*, San Francisco, CA, 2003.

[10] X.Yang and G. de Veciana. Fairness, incentives and performance in peer-to-peer networks. In *Proceedings of the 41st Allerton Annual Conference on Communication Control and Computing*, 2003.

[11] F. Clevenot and P. Nain. A simple fluid model for the analysis of SQUIRREL. Technical Report 4911, Institut National de Recherche en Informatique et en Automatique, 2003.

[12] X.Yang and G. de Veciana. Service capacity of peer-to-peer networks. In *Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), 2004*, 2004.

[13] D. Qiu and R. Srikant. Modeling and performance analysis of BitTorrent-like peer-to-peer networks. In *Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications*, Portland, Oregon, USA, 2004.

[14] M. Barbera, A. Lombardo, G. Schembra, and M. Tribastone. A Markov Model of a Freerider in a BitTorrent P2P Network, 2005. *To appear in Proc. of IEEE GLOBECOM 2005*.

[15] Bittorrent protocol specification. `http://bitconjurer.org/BitTorrent`.