Stochastic Analysis of Chain Based File Distribution Architectures with Heterogeneous Peers*

Damiano Carra and Renato Lo Cigno
Dip. di Informatica e Telecomunicazioni, Università di Trento, Trento, Italy
{carra, locigno}@dit.unitn.it

Abstract

Performance analysis of P2P systems is necessary to understand the real impact on the network of such applications. In this paper we study the performance that can be achieved by a simple file distribution architecture in a heterogeneous environment, i.e., when the access links of peers have randomly distributed capacities. The distribution architecture is a chain, where each peer downloads the content from exactly one node and uploads the content to exactly one node. Our analysis starts from a complete knowledge about peers, so that we can derive analytically the deterministic behavior and use the results as reference. We then remove part of the knowledge a peer has about its neighbors and derive the performance that can be obtained in such an environment. Results show that, if peers have sufficient information about neighbors, they can be organized in such a way that slow peers obtain near optimal performance without affecting faster peers. On the other hand, if peers do not know neighbor characteristics, slow peers have a significant impact on global performance, and other, more sophisticated, distribution architectures are required to maintain proper scalability.

1 Introduction

The Peer-to-Peer (P2P) paradigm is becoming a predominant data communication model in recent years. P2P networks form an overlay structure and peers interact one another without the presence of centralized servers. Peers can act as clients, as servers, and as application level routers interchangeably [3].

The dominant traffic observed by Internet Service Providers (ISPs) in P2P overlays is related to file sharing and distribution [2]. Due to the increasing traffic related to P2P, the study on P2P systems performance helps in understanding the influence on the network of such applications. Most P2P systems optimize the search phase, considering that, once the content is localized, P2P networks are implicitly scalable and the download phase is only a matter of the local peer bandwidth. In some contexts this assumption may be not true: during flash crowds the main problem becomes the download phase, independently from the object localization. Another example is when P2P systems are used for file distribution, e.g., anti-virus updates or critical software patch distribution. In these contexts, the point of interest becomes the efficiency of the download phase. Only very recently P2P research focused on the download performance [8, 10, 6, 9]. In many cases, analyses are based on observation of traffic measurements and discrete event simulations. In this paper we are interested in fundamental theoretic results.

We analyze the performance that can be achieved by a BitTorrent-like application [7] used for file distribution in a heterogeneous environment. Without loss of generality, we consider a given content or file and we study the time necessary for this content to be distributed to all peers. We assume a simple distribution architecture, where peers are organized in chains: as general rule, a node downloads (uploads) the content from (to) one node. When multiple parallel chains are used, one node belongs to one chain only.

Each peer has a list of peers that participate in the distribution process, and we consider two opposite situations: i) each peer knows exactly all other peer characteristics (e.g., bandwidth and position), or ii) there is no knowledge about other peer characteristics and position. In the former case, peers can be organized in smart ways, exploiting the knowledge they have. We call this case "deterministic" and use the results as reference. In the latter case, the peer to which the content is uploaded is chosen randomly among those that still do not have the content. We analyze the impact of such an uncertainty on performance.

The results show that, if peers have sufficient information about their neighbors, it is possible to define collaboration policies among peers with different bandwidths that

*This work has been partly supported by the European Union under the E-NEXT project FP6-506869. The authors wish to thank Prof. Ernst Biersack at EURECOM, who, within E-Next collaboration, provided initial ideas and many useful discussions.

allow slow peers to obtain near optimal performance without affecting faster peers. An interesting result is that the optimal policy is robust to the different ratios between number of slow peers and number of fast peers. On the other hand, if peers do not know neighbor characteristics, even a small number of slow peers can have a significant impact on global performance.

2 Deterministic Analysis: Two Classes

A chain based distribution architecture is a model where each peer can have an indefinite number of neighbors[1], but it downloads from one peer only and uploads to one peer only. The neighbors chosen for downloading/uploading could be different for different files, but remains the same for the whole file exchange. See [4] for details.

In file distributions, a server starts to upload the file to the first peer and, as soon as it terminates, starts with another peer; this behavior is adopted indefinitely until all the peers have downloaded the content. Each file is subdivided in pieces called "chunks" and each chunk can be sent separately. As soon as a peer receives a single chunk it can start to upload the chunk to the next peer, without waiting for the whole file to be downloaded.

We consider two classes of users, class 1 (fast peers) with bandwidth b_1 and class 2 (slow peers) with bandwidth b_2 ($b_1 > b_2$). Each class is symmetric, i.e., peers that belong to a class have equal upload and download capacity. The number of peers in class 1 and class 2 is equal to N_1 and N_2 respectively; the total number of peers is $N = N_1 + N_2$. We consider the distribution of a single file $\mathcal{F}$ divided into C chunks. The only bottleneck is the access bandwidth. The server has sufficient bandwidth to serve concurrently a fast peer and a slow peer. Moreover, no transmission errors or packet losses occur.

The main metrics of interest are the total time necessary to N users to complete the download of a file, and the number of peers that have completed the download at time t.

We assume that all the peers know the characteristics of all their neighbors, including their bandwidths. In such an environment, it is possible to define distribution *policies* whereby fast and slow peers collaborate to obtain the best possible overall result. We analyze different possibilities to build chains in order to minimize the distribution time. The policies are decentralized, and the chains are built node-by-node. Each node has a list of peers involved in the distribution process, and it selects the peer to upload to, provided that the contacted node has not yet received any chunk[2]. We define the following four policies:

- **Independent**: each class evolves independently. This scheme, with no interaction between classes, serves as a reference.
- **Generous**: each fast peer uploads in parallel the chunks to exactly one fast peer and one slow peer; slow peers do not upload.
- **Generous with Collaboration**: as in the previous case, but each served slow peers uploads to a slow peer, that in turn uploads to another slow peer and so on. This scheme exploits the unused capacity of slow peers.
- **Altruistic**: each fast peer, after uploading the content to exactly one fast peer, stays on-line and serves in parallel as many slow peers as possible; slow peers do not upload.

We only give some basic result; details on the complete analysis can be found in [5], where also other, tree based, distribution architectures (not covered here) are considered. This work instead introduces the stochastic analysis.

2.1 Deterministic Analysis Results

When classes evolve independently, the total number of peers of class i that complete the download at time t is

$$N^{\mathrm{Cl.i}}_{\mathrm{Ind}}(b_i, C, t) = \frac{1}{2}\frac{1}{(\mathcal{F}/b_i)}(Ct^2 - (C-2)t) \qquad (1)$$

where $\mathcal{F}\ /b_i$ is the time necessary to download the file $\mathcal{F}$ with rate b_i (details on the formula derivation can be found in [4] and [5]). Note that the number of reached peers grows linearly with the number of chunks. With respect to the variable t, the quadratic growth is intuitive since the number of chains increases linearly in time and the number of peers that complete downloading the file in each chain also evolves linearly in time. From (1) we can also derive the total time necessary to complete N peers [5]:

$$T^{\mathrm{Cl.i}}_{\mathrm{Ind}}(b_i, C, N_i) = \frac{\mathcal{F}}{b_i}\frac{(C-2) + \sqrt{(C-2)^2 + 8N_iC}}{2C}\ . \qquad (2)$$

We want to compare the different proposed policies and, concurrently, the impact of the ratio between the number of peers belonging to different classes. We only present result relevant for the following part of the paper, referring readers to [5] for details.

Figure 1 shows the total download time necessary to complete the download in two different cases, $N_1 = N_2$ and $N_2 = 10N_1$, as a function of the collaboration policy; in both cases $N = 10^4$. We normalize $\mathcal{F}\ /b_2 = 1$, and use different bandwidths b_1 with $b_1/b_2 = 5$, 10, 100. For each

[1] A peer is considered a neighbor in the overlay architecture if it is possible to contact it, i.e., its address is known.

[2] Results do not change with the dual chain building strategy, where each node selects the peer from which downloading.

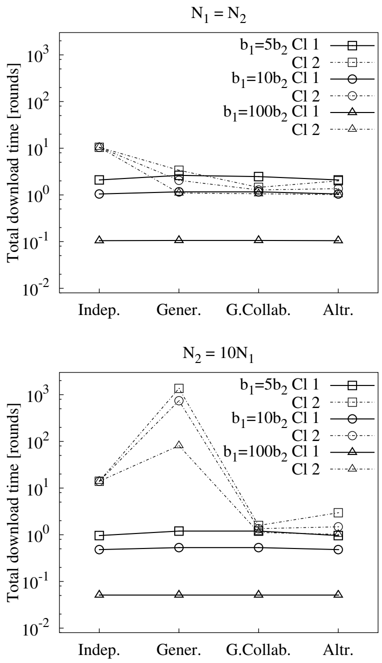

Figure 1. Total download time with chain based architecture ($N = 10^4$, $C = 10^2$)

policy (reported on x-axis) the figure shows the total download time of each class with different bandwidth ratios.

Considering fast peers (class 1, solid lines), independently from the bandwidth ratio, performances are not affected by any helping policy, thanks to the fairly high values of b_1/b_2 we use. For $b_1 \sim b_2$ things may change, but the scenario is less interesting. For slow peers (class 2, dashed lines), the Generous with Collaboration policy greatly improves the performances: in fact, by construction, $\mathcal{F}\ /b_2 = 1$ round, so the best results that can be achieved is one round. The figure shows that, regardless of bandwidth ratio and number of peers ratio, this policy performance nears optimality. In the case of $b_1 = 5b_2$ and $N_1 = N_2$ (Fig. 1, top), slow peers finish before fast peers. This is because a lot of slow chains are started almost at the same time (i.e., one slow chain every new fast peer reached by a chunk) allowing slow peers to finish nearly in one round. In the meanwhile the download time of fast peers increases marginally from 1.5 (Independent) to 1.9 rounds (Generous), because they use part of their capacities to help slow peers.

2.2 Limits and Use of the Deterministic Analysis

The deterministic analysis, when the knowledge about neighbor characteristics is complete, shows that it is possible to define simple collaboration policies that allow to obtain near optimal results for slow peers without affecting significantly fast peers. The analysis of more than two classes becomes only a matter of cumbersome calculations, but does not increase the insight. Additionally, the hypothesis of complete knowledge is unrealistic, not to mention that there are a lot of situations where the knowledge about neighbor characteristics is not available or is changing too fast to be used. Most probably in real scenarios nodes will be randomly chosen with respect to the considered characteristic.

3 Analysis with Randomly Selected Peers

Let all peers be independent and identical, so we can describe the system through a unique probability distribution of the random variable b for the peer bandwidth. The probability distribution of b summarizes the fact that in the network peers dedicate only part of the bandwidth for file distribution and also the fact that in the network there could be peers with different access technologies.

In the lack of global knowledge it is not possible to devise smart cooperation policies. If a peer starts to download/upload to a peer with less bandwidth (so there is a fraction of the download/upload bandwidth available) the remaining capacity is wasted.

Without loss of generality, we suppose that heads of the chains are always fast peers with full bandwidth available. So a new chain is started every $\mathcal{F}\ /b_{\text{fast}}$ seconds, where b_{fast} is the highest capacity in bit/s.

3.1 Single Chain Analysis

Similarly to a normal multi-link transmission, the total download time, for a given n, number of peers, can be divided into two terms: the time necessary to reach the n-th peer and the time necessary to upload the whole file to that peer. We obtain

$$t^{(n)}_{\text{total}} = t^{(n)}_{\text{reach}} + t^{(n)}_{\text{dwnl-file}} \tag{3}$$

From the probability distribution of the random variable b we can derive the probability distribution of the download time for a single peer, i.e., $f_{t_{\text{dwnl-file}}}(\tau)$, with $t_{\text{dwnl-file}} = \frac{\mathcal{F}}{b}$, where the file size $\mathcal{F}$ is a constant. The file is divided in C chunks and we can derive $f_{t_{\text{dwnl-chunk}}}(\tau)$, with $t_{\text{dwnl-chunk}} = \frac{\mathcal{F}}{Cb}$.

The time to reach the n-th peer is the time to transmit the single chunk through the chain. The transmission rate at each step is determined by the minimum capacity between the uploading and the downloading peer and corresponds to the maximum transfer time. The probability distribution of a single transfer can be found through the cumulative

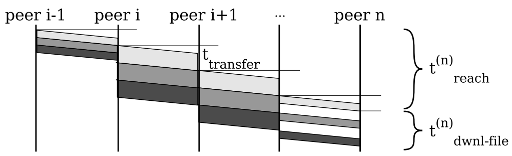

Figure 2. File transfer over a single chain

distribution of a single chunk download time:

$$\begin{aligned} t_{\text{transfer}} &= \max_{node-u, node-d}(t_{\text{dwnl-chunk}}) \\ F_{t_{\text{transfer}}}(\tau) &= F^2_{t_{\text{dwnl-chunk}}}(\tau) \end{aligned} \quad (4)$$

where $F_{t_{\text{dwnl-chunk}}}$ is the cumulative distribution function (CDF) of $t_{\text{dwnl-chunk}}$. Equation (4) is correct as far as peers are i.i.d. Since we suppose that the transfer time on each link is independent with respect to previous links, the probability distribution of n transfers is the convolution of the distributions:

$$\begin{aligned} t^{(n)}_{\text{reach}} &= \overbrace{t_{\text{transfer}} + t_{\text{transfer}} + \cdots + t_{\text{transfer}}}^{n \text{ times}} \\ f_{t_{\text{reach}}|n}(\tau) &= \overbrace{f_{t_{\text{transfer}}}(\tau) * \cdots * f_{t_{\text{transfer}}}(\tau)}^{n \text{ times}} \end{aligned} \quad (5)$$

where the symbol '$*$' defines convolution.

The time necessary to transmit the whole file, once a peer is reached, depends on the capacity of all the previous nodes. Let b^i be the capacity of node i and $b^{(i)}$ be the capacity used to transmit. By construction $b^i \geq b^{(i-1)}$, since the transmit rate $b^{(i-1)}$ includes the capacity of node i and node i receives a chunk every $\frac{\mathcal{F}}{Cb^{(i-1)}}$. The transfer rate to the node $i+1$ depends on the capacity b^{i+1}. Node i will upload chunks with a rate that is the minimum between the rate it receives the chunks and the rate node $i+1$ can accept chunks. In formulas

$$b^{(i)} = \min(b^{(i-1)}, b^{i+1})$$

or, equivalently, the time necessary to transmit the file at step i, $t^{(i)}$, is

$$\begin{aligned} t^{(i)} &= \max(t^{(i-1)}, t^{i+1}) \\ F_{t|i}(\tau) &= F_{t|i-1}(\tau) F_{t_{\text{dwnl-file}}}(\tau) = F^{i+1}_{t_{\text{dwnl-file}}}(\tau) \end{aligned} \quad (6)$$

where $F_{t|i}(\tau)$ defines the conditional CDF of the file distribution after i peers have been reached, and $F_{t_{\text{dwnl-file}}}(\tau)$ is the CDF of the file download of a single peer and the last equality is obtained by iteration.

With (5) and (6) we have the distributions of $t^{(n)}_{\text{reach}}$ and $t^{(n)}_{\text{dwnl-file}}$ respectively. Unfortunately these variables are not independent; however, since we are interested in large n and large C to exploit parallelism, the distribution of $t_{\text{dwnl-file}}$ tends rapidly to the maximum download time T_d, so that

$$f_{t_{\text{total}}|n}(\tau) \xrightarrow{n \gg 1} f_{t_{\text{reach}}|n}(\tau - T_d) . \quad (7)$$

Eqs. (2) and (7) have been validated by simulation. With the probability distribution we can calculate the mean time necessary to reach n peers and then we can build a graph of time versus the number of peers and study how the chain evolves with different bandwidth distributions as input. For notation simplicity we set $t = t_{\text{total}}$.

Starting from Eq. 7, with simple stochastic manipulation we can also obtain $f_{n|t}(\eta)$, which describes, at a given time instant, the distribution of the number of reached peers.

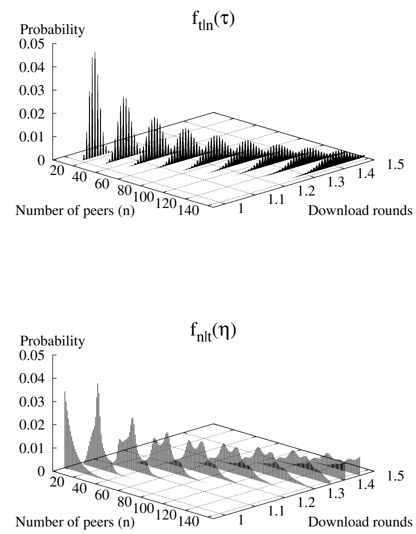

Figure 3. Example of different conditional distributions obtained with the bandwidth distribution in Table 1

Figure 3 shows the conditional distributions, $f_{t|n}(\tau)$ and $f_{n|t}(\eta)$, for different values of n and t respectively, obtained with the bandwidth distribution in Table 1.

3.2 Multiple chains

With a chain based architecture, the server, as soon as it finishes uploading the file to a peer, starts to upload to another peer, creating a new chain. A new chain is created every $\Delta = \frac{\mathcal{F}}{b_{\text{fast}}}$ seconds. Considering that the system has started the distribution at time $t = 0$, given a instant of observation t, we know exactly how many chains are present in the system and the probability distribution $f_{n|t}(\eta)$ of the number of peers in each chain. The total number of peers in the system at time t is the sum of the peers in each chain and, from the probability distribution of the number of peers in each chain, we can find the distribution of the total number of peers. Since all the chains are independent, we have

$$f_{n_{total}|t}(\eta) = f_{n|t}(\eta) * f_{n|t-\Delta}(\eta) * f_{n|t-2\Delta}(\eta) * \cdots \quad (8)$$

where the convolution is repeated until $t - i\Delta > 0$. If we are interested only in the mean number of peers, since the sum is a linear operation, we can find it from the mean number of peers of a single chain at time t, $t - \Delta$, $t - 2\Delta$, ...

4 Numerical Examples and Discussion

We start from a typical bandwidth distribution (Table 1). We set the file size $\mathcal{F}$ such that the download time with minimum rate (dial-up connection) is $\frac{\mathcal{F}}{b_{\text{slow}}} = 1$ rounds. Other rates are 10 and 20 times greater. We vary the percentage of slow users from 13% to 1%, considering that the number of cable users increases of the same percentage, leaving DSL users unmodified.

Table 1. Reference bandwidth distribution

Type	$\mathcal{F}$ / b	%
dial-up	1	13%
DSL	0.1	23%
Cable	0.05	64%

We implemented and solved the system with Octave [1]. Our program accepts as input any (discrete) probability distribution and the number of chunks that composes the file and gives as output the final distributions and some statistics such as mean values and quantiles.

Figure 4 shows the results obtained for the analysis of a single chain with the distribution of Table 1. Figure 4 (top) illustrates how the distribution of the time necessary to reach n-th peer varies with the number n of peers. As the number of peer increases, the probability distribution becomes smooth and spread, approximating a Gaussian as the central limit theorem predicts. Figure 4 (bottom) shows the distribution of the download rate as a function of the number of reached peers. Since each transmission is limited to the rate at which it receives chunks, the distribution rapidly converges on the maximum time. The explanation of this effect is simple: with a probability p_{slow} to find a slow peer, the probability that at step n the chain contains at least one slow peer is $p^{(n)}_{\text{slow}} = 1 - (1 - p_{\text{slow}})^n$.

Let's now focus on the distribution of peers given t. From this distribution we calculate the mean number of completed peers reached at time t (Fig. 5, top) with different

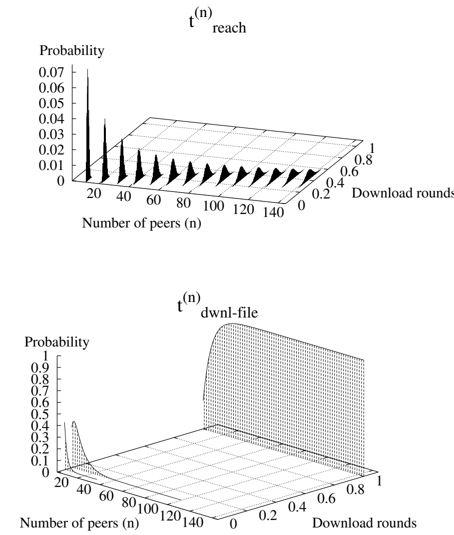

Figure 4. The distribution of the time necessary to reach n-th peer and to download the file with a single chain using the bandwidth distribution in Table 1 and C=100

fractions of slow peers. It is possible to see that, after a time lapse necessary to reach a number of peers approximately equal to $1/p_{\text{slow}}$, where p_{slow} is the probability to find a slow peer, the mean number of peers remains constant until the time reaches one round, that is the time necessary to upload the file with lowest bandwidth.

Figure 5 (bottom) shows the mean total number of completed peers in case of multiple chains. The quadratic behavior predicted in the deterministic analysis by Eq. 1 starts only after 1 round and is governed by the slow peers capacity.

The same behavior can be observed with different values of C, number of chunks (Fig. 6). As noted in Sect. 2, as the number of chunks increases, the number of peers reached at any observation time t increases linearly; C appears both in the coefficients of the linear and quadratic term, thus its influence is indeed dominant.

5 Conclusions and Future Work

We have analyzed a simple distribution architecture in P2P networks with a heterogeneous environment. We have first considered a network with complete knowledge about neighbor characteristics. In this case, we have found the best distribution policy that exploits the peer knowledge. This policy allows to obtain near optimal performance for

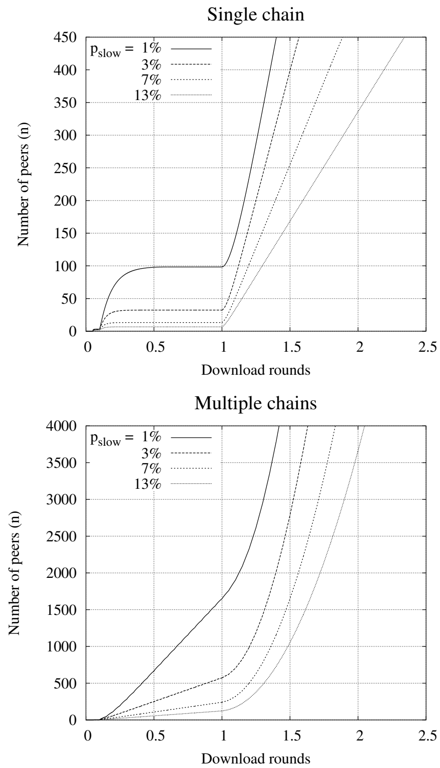

Figure 5. Mean number of completed peers as a function of t for different probability distributions (C = 100)

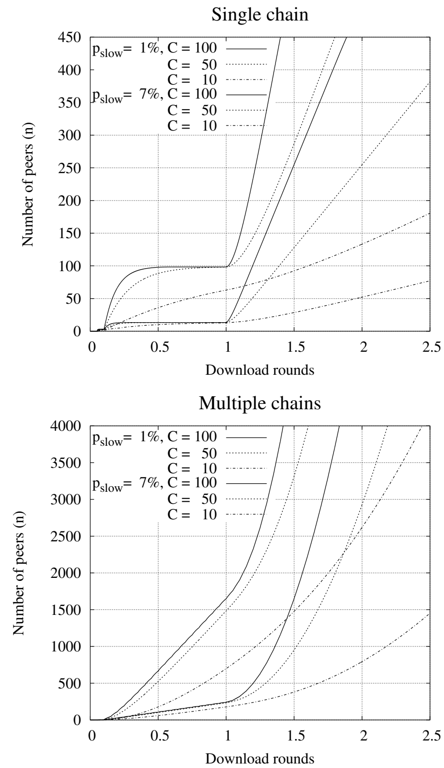

Figure 6. Mean number of completed peers for different probability distributions and different number of chunks C per file

slow peers, without affecting fast peer performance. Then, we have studied the case of peers that select randomly peers to which upload. We have found that even a small fraction of slow peers have a disruptive impact on system performance. A simple chain based architecture is therefore not effective in presence of uncertainties about peer bandwidth. Nevertheless, in contexts where bandwidth is reserved for these application and peers have complete knowledge about other peers, collaborative policies may obtain near optimal results.

We are now studying the impact of heterogeneity in tree based distribution architectures to verify how uncertainty affects these architectures and possibly to overcome the problem with flexible and adaptive collaboration policies.

References

[1] The Octave Web Page. Available: http://www.octave.org.

[2] Top applications (bytes) for subinterface: Sd-nap traffic, in CA/DA workload analysis of SD-NAP data. Available: http://www.caida.org/analysis/workload/byapplication/ sd-nap/index.xml, 2002.

[3] H. Balakrishnan, M. F. Kaashoek, D. Karger, R. Morris, and I. Stoica. Looking Up Data in P2P Systems. *Communications of the ACM*, 46 (2):43–48, Feb. 2003.

[4] E. W. Biersack, P. Rodriguez, and P. Felber. Performance Analysis of Peer-to-Peer Networks for File Distribution. In *Proc. 5th International Workshop on Quality of Future Internet Services (QofIS'04)*, Barcelon, Spain, Sept. 2004.

[5] D. Carra, R. Lo Cigno, and E. W. Biersack. Introducing Heterogeneity in Performance Analysis of P2P Networks for File Distribution. Technical Report DIT-T4-113, Univ. of Trento, Dec. 2004.

[6] F. Clevenot and P. Nain. A Simple Fluid Model for the Analysis of the Squirrel Peer-to-Peer Caching System. In *Proc. IEEE INFOCOM*, Hong Kong, Mar. 2004.

[7] B. Cohen. Incentives Build Robustness in BitTorrent. Available: http://www.bittorrent.com, 2003.

[8] Z. Ge, D. Figueiredo, S. Jaiswal, J. Kurose, and D. Towsley. Modeling Peer-Peer File Sharing Systems. In *Proc. IEEE INFOCOM*, San Francisco, California, USA, Mar. 2003.

[9] D. Qiu and R. Srikant. Modeling and Performance Analysis of BitTorrent-Like Peer-to-Peer Networks. In *Proc. ACM SIGCOMM*, Portland, OR, Sept. 2004.

[10] X. Yang and G. de Veciana. Service Capacity of Peer-to-Peer Networks. In *Proc. IEEE INFOCOM*, Hong Kong, Mar. 2004.