On Pollution in eDonkey-like Peer-to-Peer File-Sharing Networks

Kenji Leibnitz[1], Tobias Hoßfeld[2], Naoki Wakamiya[1], and Masayuki Murata[1]

[1] Graduate School of Information Science and Technology
Osaka University, 1-5 Yamadaoka, Suita, Osaka 565-0871, Japan
[leibnitz,wakamiya,murata]@ist.osaka-u.ac.jp
[2] Deptartment of Distributed Systems, University of Würzburg
Am Hubland, 97074 Würzburg, Germany
hossfeld@informatik.uni-wuerzburg.de

Abstract. In this paper we propose an analytical model for file diffusion in a peer-to-peer (P2P) file-sharing network based on biological epidemics. During the downloading process, the peer shares the downloaded parts of the file and, thus, contributes to distributing it in the network. This behavior is similar to the spreading of epidemic diseases which is a well researched subject in mathematical biology. With our model we investigate the dynamics of file diffusion focusing on the effects of pollution, e.g. malicious peers sharing corrupted version of the file.

1 Introduction

The volume of traffic data transmitted over the Internet has enormously increased recently due to the upcoming of peer-to-peer (P2P) file sharing applications. The most popular applications, such as Gnutella [1], eDonkey [2], or BitTorrent [3], are often abused for illegally sharing copyrighted content over the Internet. In P2P technology, each participant (*peer*) serves simultaneously as client and server which makes the system more scalable and robust and distinguishes it from conventional client-server architectures. However, this also comes at a slight drawback when considering content distribution. Since now, no longer a single trusted server distributes the file, malicious peers (*interference peers*) can offer fake or corrupted files and disrupt the file dissemination process.

There are two common approaches for the rightful owners of the files to protect their copyrighted property from being distributed. The first is to deliberately introduce files to the networks that are not indexed correctly. Usually, indexing is performed based on the file name, which in the case of such a *poisoned* file indicates a movie or song title other than the actual file. It is then mistakenly downloaded by other peers and the propagation of the intended file is slowed down. Another well known method is *pollution*. Here, deliberate corrupt versions of a file are injected to the network, which make use of the simple error correction methods of the file sharing applications. When the received data is recognized as corrupt, it is discarded and newly requested. This delays the overall

downloading process and if the downloading delay exceeds the user's patience, he may become frustrated and abort the download.

P2P networks can be briefly classified in *pure* and *hybrid* types [4]. Unlike pure P2P networks, e.g. Gnutella, hybrid networks have additional entities which have special functions. In the eDonkey network, each peer connects to an index server which indexes all shared files and over which the search for a certain file is performed. In a similar manner, BitTorrent uses trackers accessed over WWW pages to provide the information about the peers sharing a file. *Seeders* are peers that only provide the complete file in the network for other peers to download. After a file has been downloaded, the peer may itself become a seeder or a *leecher* who does not participate in the file sharing after downloading it.

The file diffusion process itself is comparable to the spreading of a disease in a limited population. There exist many models for population dynamics in mathematical biology [5] dealing with predicting if a disease will become an epidemic outbreak or what vaccination strategy is most appropriate. In this paper we will use modeling techniques from biological epidemic models to predict the diffusion characteristics of single files shared in a P2P network. With the model we investigate the rate of diffusion, as well as the effects of pollution. We are particularly interested in the influence of these interference peers as they can greatly change the diffusion behavior of the file. By modifying the population size of interference peers we can achieve approximately the same effects as performing a vaccination of the susceptible population [6]. Additionally, our model takes the distinction between leechers and seeders into account and we show the influence of selfish peers on the file dissemination process. Especially, the ratio between seeders and interference peers and the willingness of the user to share the file are evaluated.

The paper is organized as follows. In Section 2, we will briefly overview the work related to modeling of P2P file sharing. This is followed by the description of the considered file sharing application in Section 3. The biologically motivated model and its extension to P2P is discussed in Section 4. We will give numerical results in Section 5 on the influence of the interferers. The paper is concluded in Section 6 with an outlook on future work and on mechanisms for improving the system to increase security and reliability.

2 Related Work

Epidemic methods have been considered as simple and effective protocol for disseminating data in communication networks. The main feature of these *gossip*-based protocols is that they do not require any specific topology and that any node has the same probability to contact another node. This approach has been used to devise gossip-based protocols operating in mobile ad-hoc networks [7] or in unstructured P2P networks [8–10]. The effect of the network topology on the spread of the epidemics is studied in [11].

In epidemic computing, nodes contact each other with a certain rate and depending on the rate of cure to infection a disease may become an epidemic.

Epidemic models are also well suited to model the diffusion behavior of specific information in a network, see [12], and has often been applied to forecast the spreading of worms and viruses in the Internet [13]. In a very similar manner, epidemic models can be used to model file diffusion in P2P file sharing networks. The papers found on P2P file diffusion either consider measurement studies, e.g. [14, 15], or by means of simulation [16, 17]. A theoretical model of a BitTorrent P2P network can be found in [18]. The authors use a fluid model and study the performance of the network and investigate the effects of the incentive mechanism on the network. This work is extended to considering different classes of access links in [19] and the authors show that bandwidth heterogeneity can have a positive effect on content propagation. While in [18] and [19] the steady-state network performance is investigated, we emphasize the time-dynamics of the system which requires us to consider non-stationary process, e.g. caused by flash crowd arrivals of file requests. Measurement studies on pollution and poisoning can be found in [20, 21]. Both papers show that there can be a substantial influence from introducing even a small number of interference peers to the network.

Chen et al. [22] and Thommes and Coates [23] use a model based on the classic SIR approach, which is also the fundamental idea of our work. However, as we will see later from comparison with simulations, the steady state assumptions made in many papers, e.g. in [18], are not appropriate due to the highly non-stationary behavior of the system. The transitions are made between the states after a fixed amount of data has been downloaded. Using simply a transition rate does not properly reflect the system dynamics. The focus in this work is on the time-dynamic transition phase during the diffusion process. This facilitates the consideration of flash crowd arrivals of file requests. When considering illegally shared content this often corresponds to the release date of a song or a movie as at that time the number of requests will be highest. For legally distributed software (e.g. distributions of the Linux operating system) P2P file-sharing is also much more effective for content distribution than client-server as it relieves the download servers from overloading when new software releases are available [15]. Furthermore, we investigate the influence of interference peers that share corrupt or fake content on the diffusion process.

3 P2P File-Sharing Application

In the following we will consider a file sharing application similar to eDonkey which belongs to the class of hybrid P2P architectures and comprises two separate applications: the *eDonkey client* (or *peer*) and the *eDonkey server*. The eDonkey client shares and downloads files. The eDonkey server operates as an index server for file locations and distributes addresses of other servers.

A main feature of P2P file sharing applications like BitTorrent, Kazaa and eDonkey is the ability to perform *multiple source downloads*, i.e. peers can issue two or more download requests for the same file to multiple providing peers in parallel and the providing peers can serve the requesting peer simultaneously.

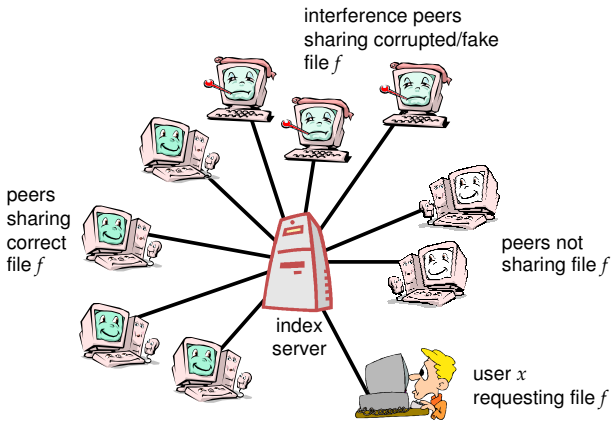


Fig. 1. Index server and peers sharing correct and fake files

3.1 Architecture of the eDonkey Network

Before an eDonkey client can download a file, it first gathers a list of all potential file providers. To accomplish this, the client connects to one of the eDonkey servers. Each server keeps a list of all files shared by the clients connected to it. When a client searches for a file, it sends the query to its main server which may return a list of matching files and their locations.

In [16] we showed from measurements that about 50% of the total number of eDonkey users are connected to the seven largest index servers with population sizes between 50,000 and 500,000. Since the connected number of peers at these index servers is so large, we can assume a Poisson arrival process for the generation of file requests.

In general we will study in this paper a scenario as shown in Fig. 1. There are N peers connected to the index server and when a peer requests a certain file, the index server responds with a list of sharing peers. However, some of the sharing peers may appear to have the file, but actually only offer a corrupted or fake version. Our goal is to investigate the influence of these interfering peers on the file diffusion process. The following sections will give more details on the file sharing process itself, further details can be found in [16].

3.2 File Structure

The general structure of an arbitrary file f that is shared in the eDonkey network is depicted in Fig. 2. The file has a size of f_{size} kB and comprises a number of $c_{max} = \lceil \frac{f_{size}}{c_{size}} \rceil$ chunks, each with a constant size of $c_{size} = 9500$ kB with exception of the final chunk c_{max} which may be smaller in size. The operator $\lceil x \rceil$ returns the next largest integer value of x. As soon as a peer has downloaded all blocks of a chunk, the peer becomes a source for this chunk and shares it to

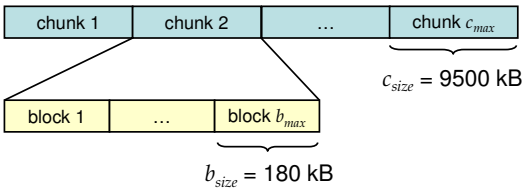


Fig. 2. Structure of a file on eDonkey application layer

the community. The full chunk is not transmitted in whole, but in units of blocks with size $b_{size} = 180$ kB. These values of b_{size} and c_{size} are taken from the source code of the eDonkey implementation eMule v0.30e.

A block is requested from a peer who shares the whole chunk containing this block. After all blocks of a chunk have been downloaded by a requesting peer, an error detection mechanism is performed. In eDonkey, this is done via comparing the hash value of the received chunk with the sender's hash value of the chunk. In case of an error, i.e., at least one block is corrupted due to malicious peers offering fake or corrupted data, the complete chunk is discarded and has to be requested and downloaded again. If the user's patience in terms of required download time exceeds a certain threshold (which we will express later with the factor Θ), the user will abort the download of the complete file.

After a peer has successfully downloaded all blocks of chunk i, he immediately acts as a *sharing peer* for this chunk and the number of sharing peers is incremented by one. Thus, all users in an eDonkey network act simultaneously as sharing peers and downloading peers. Although, the user cannot influence that each chunk is shared during downloading, he can show a different behavior after the file has been entirely downloaded. We take this into account in our model by introducing p_{share} as the probability that a user shares file f for an exponentially distributed period B. All users in the system are modeled with identical values of p_{share} and B. Hence, $p_{share} = 0$ indicates a system consisting

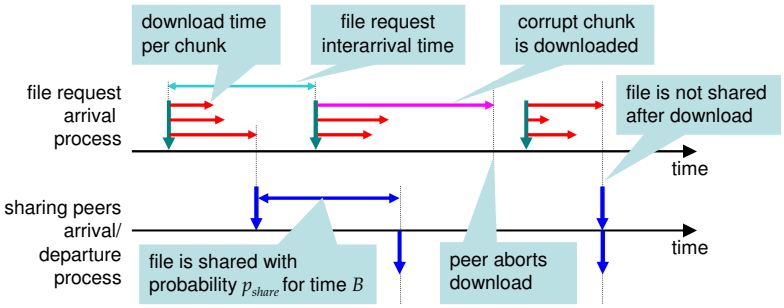


Fig. 3. Arrival process of peers sharing a file

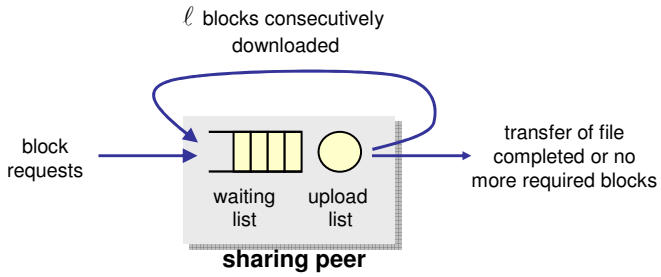


Fig. 4. Upload queue model

entirely of *leechers*, i.e. users who only share the file during the download and immediately stop sharing it once the download is completed. The arrival process and the sharing process in our eDonkey model are illustrated in Fig. 3.

3.3 Upload Queue Management

The upload management of an eDonkey peer maintains an upload queue which consists of two lists, the *waiting list* and the *uploading list*. The uploading list holds the requests which are currently served. A download request is served as soon as it obtains an upload slot, i.e. it moves from the waiting list to the uploading list, see Fig. 4. Typically, each served request gets an *equal share* of the upload capacity and we assume that only one block is downloaded at a time, i.e., $\ell = 1$. However, different modifications of the original eDonkey client exist, which may change this behavior [24]. The complex *scoring mechanism* of eDonkey decides which request is served next. One important factor of the scoring system is the "high ID/low ID" mechanism to ensure fairness among all peers. A high ID increases the score, whereas a low ID reduces it. A peer gets a low ID if the server cannot establish a new connection to the peer, e.g. the peer is located behind a firewall or a NAT. The blocking of incoming connections or an invalid IP address would hinder to contact this peer and is unfair since these peers would not answer to file requests.

Let us consider an arbitrary peer x wishing to download file f. The peer issues a request for the file to the index server and receives a list of all known peers that share the complete file or chunks of it. The peer then requests individual blocks from other peers sharing the chunk containing the desired block. All requests are appended to the waiting list of the sharing peer and a weighting mechanism handles the scheduling of the upload queue requests for transmission. The underlay network infrastructure over which the data is transmitted is assumed to be identical for all peers, e.g. DSL connections. In our model, we assume an unlimited length of the upload list and split the upload bandwidth among all requesting peers. The total available upload bandwidth is distributed according to the max-min fair share principle, cf. [16].

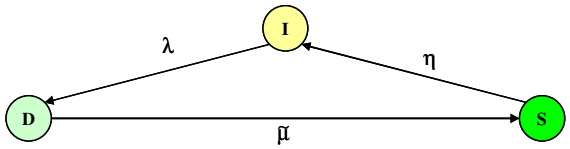


Fig. 5. Simple IDS state space

4 Epidemic Model of File Diffusion

In the following, let us consider a basic epidemic model for P2P file sharing. In general, the epidemic model categorizes the population in groups depending on their state. A commonly used approach is the SIR model [5]. SIR is an abbreviation for the states that are taken during the course of the spread of the disease. At first, there are *susceptibles*, which are users that can be possibly infected with a certain rate. When they are contacted with the disease, they move to the state of *infectives* and can pass the disease on to other members of the susceptible population. Finally, there is the *removed* population, consisting of users who have either fatally suffered from the disease or have recovered and become immune to it. In either case, they can not get infected by the disease again. An important issue is that the total population N remains constant.

4.1 Analogy of P2P to Biological SIR Model

In this section we will describe the basic underlying biological model and show the commonalities with P2P file sharing. Although there are various analogies between both models, we will see that simply applying an SIR model is insufficient due to the complexity of the P2P applications. However, the principle time-dynamic modeling technique from biology will be maintained and unlike [18] we are able to consider cases that are not in steady state.

Let us now define a model similar to SIR in the context of file sharing. We denote the number of susceptibles as *idle peers* $I(t)$ at time t. From this set, the file requests are generated with a rate of $\lambda(t)$, which can be a time dependent function or a constant reflecting the popularity of the file, see [16]. Once the peer starts to download the file, he is attributed to the set of *downloading* peers $D(t)$. The download rate $\tilde{\mu}(t)$ depends on the number of peers sharing the file and the other downloading peers, which all compete for the download bandwidth. Once downloading of the complete file with size f_{size} is finished, the peer joins the *sharing* peers $S(t)$, that offer the file to the other users. The peer shares the file only for a limited time after which he returns with rate η to the idle peers, see Fig. 5. Note that this is a rather simplified view for a generic file sharing application, as the detailed mechanism in eDonkey involves downloading and sharing chunks of the file, see Section 3.2. As we will see later, the sharing of smaller data units also has an impact on the accuracy of the model.

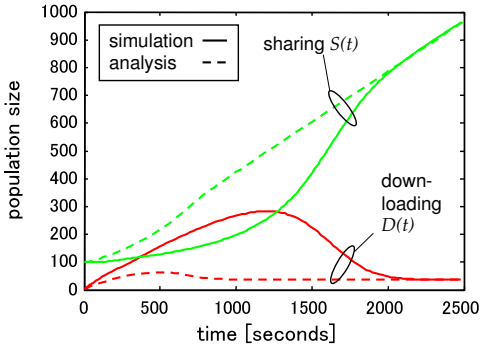


Fig. 6. Comparison of simulation results with basic IDS model

Thus, the dynamic system of the sharing process can be expressed by the equation system given in (1)-(3). In analogy to the SIR model, we will refer to it as the IDS model.

$$\frac{dI(t)}{dt} = -\lambda(t)I(t) + \eta S(t) \tag{1}$$

$$\frac{dD(t)}{dt} = \lambda(t)I(t) - \tilde{\mu}(t)D(t) \tag{2}$$

$$\frac{dS(t)}{dt} = \tilde{\mu}(t)D(t) - \eta S(t) \tag{3}$$

The initial values are $I(0) = I_0$, $S(0) = S_0$, and $D(0) = N - I_0 - S_0$.

In Eqn. (1)-(3) we can at first assume a constant request arrival rate λ which is adapted to match a Poisson arrival process and the main problem lies in the determination of the download rate $\tilde{\mu}(t)$. Let us define the upload and download rates as r_u and r_d, respectively. For the sake of simplicity, we assume homogeneous users with ADSL connections, resulting in rates of $r_u = 128$ kbps and $r_d = 768$ kbps. Since eDonkey employs a fair share mechanism for the upload rates, there are on average $S(t)/D(t)$ peers sharing to a single downloading peer and we multiply this value with r_u which gives us the bandwidth on the uplink. However, since the downloading bandwidth could be the limiting factor, the resulting effective transition rate $\mu(t)$ consists of the minimum of both terms divided by the file size f_{size}, see Eqn. (4).

$$\tilde{\mu}(t) = \frac{1}{f_{size}} \min\left\{\frac{S(t)\, r_u}{D(t)}, r_d\right\} \tag{4}$$

The dynamics of the populations of D and S are shown in Fig. 6 and compared to the mean population sizes, i.e. mean number of peers, obtained from 5000 simulation runs. We selected $S_0 = 5000$, $I_0 = 100$ and a constant λ of 1300 requests per hour. For the sake of simplification we consider at this point $\eta = 0$, i.e., all peers remain sharing peers after a completed download and do

not leave the system. The shape of the I curves is not very interesting to us in this scenario, since it just linearly decreases.

When comparing the simulation with the analytical model, we can see that the same general shape matches for $t > 2000$, whereas a problem arises w.r.t. the accuracy of the model for smaller values of time t. This can be explained as follows. The transition from D to S is performed only after the complete file with fixed size f_{size} has been downloaded. The current model using the states I, D, and S, however, is memoryless and does not take into account the number of bits that have already been downloaded. The transitions between these states are given here as rates indicating the "average" number of transitions per time unit. In reality, the average download rate changes during the downloading process of an individual peer and it is insufficient to consider it a priori as constant for the complete file. While this assumption is generally applied in epidemic modeling of diseases, we wish to provide an enhanced mathematical model by considering a finer granularity. In the following we will, therefore, minimize the error by splitting the macro state D into M smaller states corresponding to the number of bits downloaded. We expect that when M approaches infinity that the error will be reduced to zero.

4.2 Detailed File Sharing Model

For the sake of simplicity, we consider in the following the last chunk of a file which is the most interesting one, as its completion results in the completion of the entire file. The user can then decide whether the whole file is shared or not, i.e., whether the peer becomes a leecher or a seeder. In the following the terms file and last downloaded chunk have the same meaning.

Let us split the file with size f_{size} into M logical units which we will consider individually. Our model thus increases by the states $D_0, \ldots, D_M$. We can interpret the states D_i as the state where i logical units have been successfully downloaded, i.e., D_0 means that the downloaded is initiated and D_M indicates a complete download. After reception of each block, the queue mechanism of eDonkey determines the sharing peers from which the next block is downloaded. This involves an update of the download rate $\mu(t)$ after each logical unit. If we choose the logical unit as blocks, our model is exact and the obtained numerical error is acceptably small as will be shown in more detail later, cf. Fig. 8(a). The transitions from the states D_i use a rate $\mu(t)$ similar to the one described in Eqn. (4).

$$\mu(t) = \frac{f_{size}}{M} \min\left\{ \frac{S(t)\, r_u}{\sum_{i=0}^{M-1} D_i}, r_d \right\} \tag{5}$$

A further enhancement of the simple model is the introduction of p_{share} as the probability of sharing a file. We denote its complementary probability as $\bar{p}_{share} = 1 - p_{share}$. The updated state space with transitions is illustrated in Fig. 7. After the M-th logical unit has been downloaded, the peer enters the

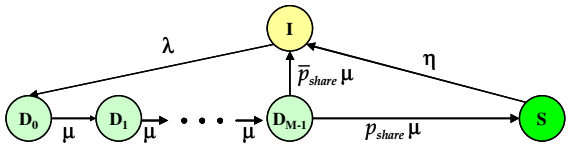


Fig. 7. Detailed IDS state space

sharing peers with probability p_{share} and returns to the idle state with $\bar{p}_{share}$. This corresponds to the user leaving the system after downloading (leecher) or downloading it another time again at a later time.

The new equation system is summarized below. The original model given in Section 4.1 corresponds to a value of $M = 1$. Obviously, the larger M is, the more accurate is the model, but the computational requirements for solving the equation system increases as well. Finding a good value of M involves a tradeoff between accuracy and computation speed.

$$\frac{dI(t)}{dt} = \bar{p}_{share}\mu(t)D_{M-1}(t) - \lambda(t)I(t) + \eta S(t) \tag{6}$$

$$\frac{dD_0(t)}{dt} = \lambda(t)I(t) - \mu(t)\,D_0(t) \tag{7}$$

$$\frac{dD_i(t)}{dt} = \mu(t)\,(D_{i-1}(t) - D_i) \quad \forall_{1\le i<M} \tag{8}$$

$$\frac{dS(t)}{dt} = p_{share}\mu(t)\,D_{M-1}(t) - \eta S(t) \tag{9}$$

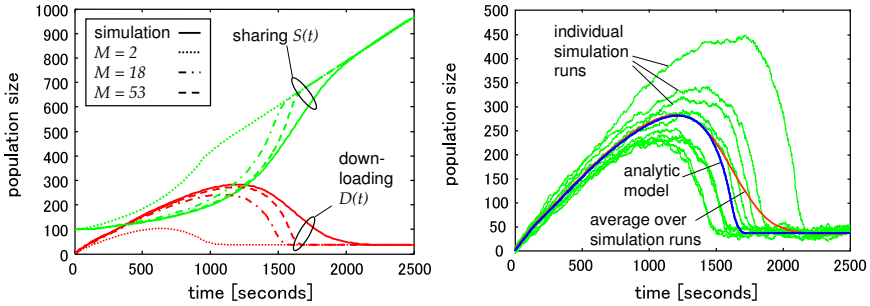
Again, we must include the condition to keep the total population at the index server constant at

$$N = I + \sum_{i=1}^{M} D_i + S. \tag{10}$$

However, since the equation system is a closed system, it is sufficient to ensure that the initial values obey this constraint. Hence, we assume that $N = I_0 + S_0$ and $D_i = 0$ for all i. The considered values for M are given in Tab. 1, thus, the largest number of equations is given for $M = 53$ which corresponds to a size of

Table 1. Considered discretization steps

pieces M	**size** [kB]	**logical unit**
1	9500	chunk
2	4750	segment
18	540	download unit
53	180	block



(a) Influence of different values of M (b) Error in average over simulation runs

Fig. 8. Extended IDS model

180 kB pieces and is the block size b_{size} as described in Section 3.2. This means that the size of the download units in our model is given in blocks.

The extended model is compared to simulation results in Fig. 8(a). We can recognize that using a large value of M greatly improves the accuracy of the model. Note that the task of comparing results averaged from simulation runs to the mathematical model is not fully appropriate. The differential equations describe the general behavior of a single evolution over time, depending on the initial values and boundary values. We can easily match the initial values, but the boundary conditions in the simulation depend for example also on the realization of each random variable. Each individual simulation run matches exactly the shape of the analytical model, however, depending on the random variables can be different in scale, see Fig. 8(b). When we average over the series of simulation runs, this leads to the different decreasing slope between time 1500 s and 2000 s in Fig. 8(a).

4.3 Influence from Corrupted Files

So far the model assumed that all peers share correct versions of the file and none is corrupted. Now we will investigate the influence caused by these interference peers, whose cardinality we will denote as X in the following. Since the model does not distinguish between specific chunks, it is sufficient to just consider an arbitrary chunk instead of the complete file in the following.

We model the user's patience by assuming that he is willing to wait Θ times the expected download duration, before aborting his attempt. He can then later attempt to download it again. Each peer has a probability of $p(t)$ of requesting a block from a correct sharing peer or $\bar{p}(t)$ from a peer sharing a corrupted version. This probability simply depends on the fraction of interferers to the total number of sharing peers.

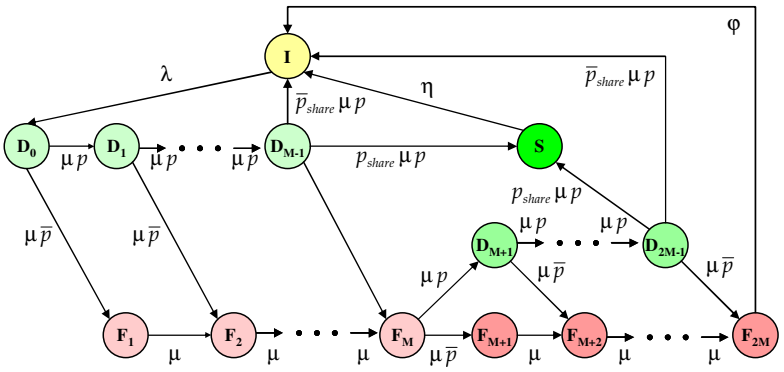


Fig. 9. Fully extended IDS model with interferers, $\Theta = 2$

$$p(t) = \frac{S(t)}{S(t) + X} \qquad \bar{p}(t) = 1 - p(t) \tag{11}$$

The state space for the enhanced model is shown in Fig. 9 for $\Theta = 2$. Beginning with the idle state, the series of D_i states is traversed, each one offering the possibility to get into a "fake" state with probability $\bar{p}(t)$. Once the file has been completely downloaded without errors, the peer may share it or return directly to the idle state, depending on the sharing probability p_{share}. On the other hand, if the download of the chunk was attempted from fake peers, the user may retrieve it successfully at the second attempt, see states D_{M+1} to $D_{\Theta M-1}$. The interpretation for the F_i states can be given in the following way. The peer has downloaded i blocks, however, one of them is corrupt. The states F_{kM} for $1 \leq k < \Theta$ are the points in time when the complete chunk has been downloaded and an integrity check is performed and an error recognized. We assume that the time required for checking the file can be neglected. A special state is given at $F_{\Theta M}$ where the patience of the peer is exceeded and he aborts the download attempt and enters the idle state. The states correspond to the following equation system.

$$\frac{dI(t)}{dt} = \bar{p}_{share}\mu(t)p\sum_{k=1}^{\Theta} D_{kM-1}(t) + \varphi\, F_{\Theta M} + \eta\, S(t) - \lambda(t)I(t) \tag{12}$$

$$\frac{dD_0(t)}{dt} = \lambda(t)I(t) - \mu(t)D_0(t) \tag{13}$$

$$\frac{dD_i(t)}{dt} = \mu(t)\left[p(t)D_{i-1}(t) - D_i(t)\right],\ \forall_{1\leq i<\Theta M}\ \forall_{1\leq k<\Theta : i\neq kM \wedge i\neq kM+1} \tag{14}$$

$$\frac{dD_{kM+1}(t)}{dt} = \mu(t)\left[p(t)F_{kM}(t) - D_{kM+1}(t)\right],\ \forall_{1\leq k<\Theta} \tag{15}$$

$$\frac{dF_1(t)}{dt} = \mu(t)\left[\bar{p}(t)D_0(t) - F_1(t)\right] \tag{16}$$

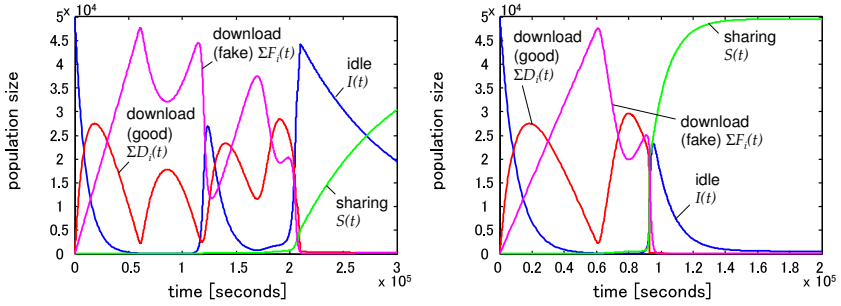
$$\frac{dF_i(t)}{dt} = \mu(t)\left[\bar{p}(t)D_{i-1}(t) + F_{i-1}(t) - F_i(t)\right],\ \forall_{2\le i<\Theta M}\,\forall_{1\le k<\Theta: i\neq kM+1} \tag{17}$$

$$\frac{dF_{kM+1}(t)}{dt} = \mu(t)\left[\bar{p}(t)F_{kM}(t) - F_{kM+1}(t)\right],\ \forall_{1\le k<\Theta} \tag{18}$$

$$\frac{dF_{\Theta M}(t)}{dt} = \mu(t)\left[\bar{p}(t)D_{\Theta M-1}(t) + F_{\Theta M-1}(t)\right] - \varphi F_{\Theta M}(t) \tag{19}$$

$$\frac{dS(t)}{dt} = p_{share}\mu(t)p\sum_{k=1}^{\Theta} D_{kM-1}(t) - \eta S(t) \tag{20}$$

Example results are shown in Fig. 10. In both cases we have the initial populations of $I_0 = 50000$, $S_0 = 100$, and $X = 10$ interference peers. The request arrival rate is $\lambda = 10^{-4}$, the repeated request rate after success is $\eta = 10^{-6}$, and the repeated request rate after failure is $\varphi = 10^{-3}$. We use $M = 53$ states and the threshold for the user's patience is $\Theta = 2$. The same values for both parameters are used throughout this entire paper. Each user has a sharing probability of $p_{share} = 0.1$ in Fig. 10(a) and $p_{share} = 0.9$ in Fig. 10(b). The general behavior of the system can be described as follows. In the beginning most users are idle but start downloading the file either from a good source or an interferer. Both downloading populations have the same shape with two successive peaks which is caused by the number of retrials Θ in our model. During this time, the number of sharing peers and interferers remains constant, so the probability of downloading a fake block $(1 - p(t))$ stays also constant. Once the first peers succeed in downloading the file, this probability becomes less and the sharing peers increase enormously. The sharing probability p_{share} influences the time when this sudden increase of sharing peers occurs and since more peers share the file, less will return to the idle state for the second or third time. We can also see in Fig. 10(a) that two cycles of the ΘM downloading states are traversed



(a) $p_{share} = 0.1$ (b) $p_{share} = 0.9$

Fig. 10. Example results with full model

before the sharing peers increase. This is caused by the low sharing probability of 0.1.

5 Numerical Results

In this section we will investigate the influence of the interference peers on the file diffusion process. We have seen that the sharing probability p_{share} plays an important role on the speed of convergence to the steady-state, i.e., where a population size becomes independent of time. In the following we will examine the number of aborted downloads due to the presence of the interference peers. Since it is a non-stationary process, the steady-state values do not give us much information. Therefore, we choose an arbitrary (but fixed) observation time instant at which we obtain the number of aborted downloads. In the following examples, we chose the time instant to be one day after the whole process starts. In order to evaluate the number of aborted downloads, we select $\varphi = 0$, i.e. aborted users do not return to the idle state. Thus, the number of aborted downloads is reflected by the state $F_{\Theta M}$.

If aborted users do not reattempt to download the file and the total population remains constant, we can recognize in Fig. 11 that already a small number of peers is sufficient to severely disrupt the diffusion process of a file. Fig. 11 shows the number of aborted downloads over the number of interference peers and the ratio of initial sharing peers over interference peers. The sharing probability was set constant to $p_{share} = 0.5$. Increasing the number of interference peers leads to an increase in aborted downloads. Especially, when the number of fake sources is greater than 15, there will be no successful downloading attempt. When the ratio between correctly sharing peers and fakes is large, this effect is greatly reduced. We can see that when this ratio is small, there is a steep incline between 10 and 15 interference peers indicating that when this ratio is small, it is very important how large the absolute number of interference peers actually is. This transition region is very critical for the proper diffusion of the file. Fig. 11 shows that a relatively small number of interference peers is sufficient to disrupt the propagation of a file in an eDonkey network.

In Fig. 12 we investigate the influence of the sharing probability p_{share} on the aborted downloads. The number of initially sharing peers is chosen as $S_0 = 100$. We can recognize in Fig. 12(a) that the number of interference peers has more dominating effect on the aborts rather than the sharing probability. The aborted downloads are nearly a step function over the interference peers X, with the transition at about $X = 13$. The sharing probability p_{share} only shows a slight influence in the phase when the number of aborts drastically increases. The reason for this behavior is that in this considered range we have a ratio of S_0/X between 0 and 5 which has a shape similar to a step function as can be seen in Fig. 11.

The influence of the sharing probability increases when we choose a higher initial value, e.g. $S_0 = 200$ as in Fig. 12(b). The step function behavior which can be briefly recognized for $p_{share} < 0.2$ gives way to a more gradual and smooth

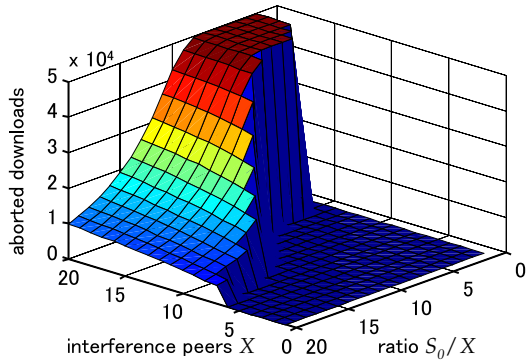
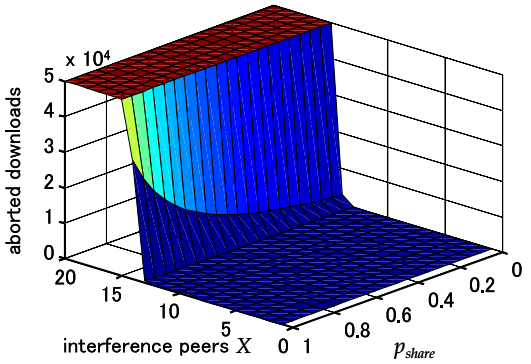
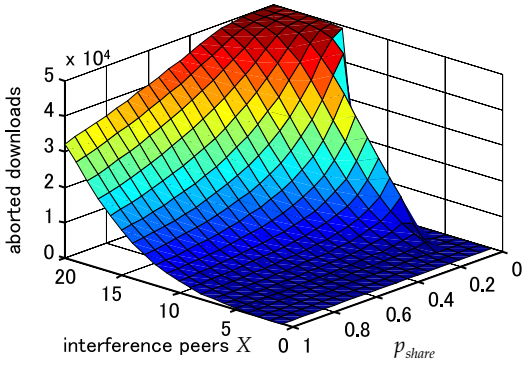


Fig. 11. Aborted downloads over sharing ratio and number of interferers



(a) $S_0 = 100$



(b) $S_0 = 200$

Fig. 12. Aborted downloads over sharing probability and number of interferers

increase. Due to the better ratio S_0/X, the abort probability does not reach the total population for $X = 20$. What we also can see from this figure is that for the same p_{share} value we have a slightly higher number of aborts in the range before the step is made in Fig. 12(a). In summary we can see that when the ratio between initially correct and fake files is rather small, the sharing probability does not show any impact on the number of aborted downloads. Instead when this ratio increases, a larger p_{share} smoothens the curve, making the increase of the aborted downloads a more gradual one.

6 Conclusion and Outlook

We presented in this paper an analytical model for the file diffusion process in a P2P file sharing network similar to eDonkey. The model is based on an epidemic model with different populations like the well-known SIR model, but in our case corresponds to the number of idle peers, peers currently downloading the file (or chunk), and those sharing it. Our focus was on the investigation of the performance of the system in the presence of malicious interference peers sharing a corrupt version of the file. We could see that using a simple SIR-like model is not very accurate, nor does the assumption of steady state which is often found in other publications yield very useful results when we study the highly dynamic behavior seen in file diffusion. We, therefore, considered separate populations for peers having downloaded certain parts of the file and could improve the accuracy of the model when we compared the results to simulation runs. The final model included states characterizing the peers having downloaded a certain number of download units from either good or malicious sharing peers. Furthermore, the user's patience was also taken into account by assuming that he is willing to wait for a download a factor Θ times of the original file size.

The numerical results showed that a small number of interference peers can greatly inhibit the propagation of a file. This fact can be used for content providers to protect their copyrighted material from being illegally distributed in the network by introducing a sufficient number of interference peers. A higher willingness of the user to share the file after successfully downloading it can reduce the number of aborted downloads, if the initial sharing ratio among good and corrupt files is sufficiently large. To overcome the influence from fake peers, more recent versions of eDonkey clients, e.g. eMule, the *Intelligent Corruption Handling* (ICH) mechanism is implemented that performs the error detection on smaller data units than chunks. Instead of discarding the complete chunk when at least one corrupted block is received, only all blocks of the damaged segment need to be re-requested. At the moment, ICH enables to retransmit only half chunks instead of complete ones. This means that ICH determines in which half of the chunk the error occurred. The extent of performance improvement due to ICH is an issue we wish to examine in future work.

Another important issue related to interference in file sharing is authenticity and reliability. In order for the content provider to ensure that the file is not being modified by other malicious sources, a client-server solution offers natu-

rally greater security by having a single trusted source offering the information. However, if the served content has a high popularity, there will be a high request rate leading to a drastic increase in server load. The tradeoff between a client-server architecture and distributed file-sharing using P2P is another interesting subject that we will study in greater detail. Another challenging task is to derive a similar diffusion model for the Avalanche [25] network which has recently gained in popularity.

Acknowledgement This research was supported by "The 21st Century COE Program: *New Information Technologies for Building a Networked Symbiosis Environment*" and a Grant-in-Aid for Scientific Research (A)(2) 16200003 of the Ministry of Education, Culture, Sports, Science and Technology in Japan.

References

1. Gnutella Protocol Development Website. (http://rfc-gnutella.sourceforge.net/)
2. eDonkey2000 Home Page. (http://www.eDonkey2000.com/)
3. The Official BitTorrent Home Page. (http://www.bittorrent.com/)
4. Schollmeier, R.: A definition of peer-to-peer networking for the classification of peer-to-peer architectures and applications. In: IEEE 2001 International Conference on Peer-to-Peer Computing (P2P2001), Linköping, Sweden (2001)
5. Murray, J.: Mathematical Biology, I: An introduction. 3 edn. Springer (2002)
6. Verriest, E., Delmotte, F., Egerstedt, M.: Control of epidemics by vaccination. In: American Control Conference, Portland, OR (2005)
7. Haas, Z., Halpern, J., Li, L.: Gossip-based ad hoc routing. In: Proc. of IEEE INFOCOM, New York City, NY (2002) 1707 – 1716
8. Eugster, P., Guerraoui, R., Kermarrec, A.M., Massoulie, L.: Epidemic information dissemination in distributed systems. IEEE Computer **37** (2004) 60–67
9. Lindemann, C., Waldhorst, O.P.: Exploiting epidemic data dissemination for consistent lookup operations in mobile applications. SIGMOBILE Mob. Comput. Commun. Rev. **8** (2004) 44–56
10. Jelasity, M., Montresor, A., Babaoglu, O.: Detection and removal of malicious peers in gossip-based protocols. In: 2nd Bertinoro Workshop on Future Directions in Distributed Computing: Survivability: Obstacles and Solutions (FuDiCo II: S.O.S.), Bertinoro, Italy (2004)
11. Ganesh, A., Massoulie, L., Towsley, D.: The effect of network topology on the spread of epidemics. In: IEEE Infocom, Miami, FL (2005)
12. Khelil, A., Becker, C., Tian, J., Rothermel, K.: An epidemic model for information diffusion in MANETs. In: 5th ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM). (2002) 54–60
13. Pastor-Satorras, R., Vespignani, A.: Epidemic spreading in scale-free networks. Physical Review Letters **86** (2001) 3200–3203
14. Tutschku, K.: A measurement-based traffic profile of the eDonkey filesharing service. In: 5th Passive and Active Measurement Workshop (PAM2004), Antibes Juan-les-Pins, France (2004)
15. Izal, M., Urvoy-Keller, G., Biersack, E., Felber, P., Hamra, A.A., Garcés-Erice, L.: Dissecting BitTorrent: Five months in a torrent's lifetime. In: 5th Passive and Active Measurement Workshop (PAM2004), Antibes Juan-les-Pins, France (2004)

16. Hoßfeld, T., Leibnitz, K., Pries, R., Tutschku, K., Tran-Gia, P., Pawlikowski, K.: Information diffusion in eDonkey-like P2P networks. In: Australian Telecommunication Networks and Applications Conference (ATNAC), Bondi Beach, Australia (2004)
17. Zerfiridis, K.G., Karatza, H.D.: Optimized dissemination of highly anticipated content over an itinerary based P2P network. In: 37th Annual Simulation Symposium (ANSS), Arlington, VA (2004)
18. Qiu, D., Srikant, R.: Modeling and performance analysis of BitTorrent-like peer-to-peer networks. In: ACM SIGCOMM'04, Portland, OR (2004)
19. Lo Piccolo, F., Neglia, G., Bianchi, G.: The effect of heterogeneous link capacities in BitTorrent-like file sharing systems. In: International Workshop on Hot Topics in Peer-to-Peer Systems (HOT-P2P'04), Volendam, The Nederlands (2004) 40–47
20. Liang, J., Kumar, R., Xi, Y., Ross, K.: Pollution in P2P file sharing systems. In: Proc. of IEEE INFOCOM, Miami, FL (2005)
21. Christin, N., Weigend, A.S., Chuang, J.: Content availability, pollution and poisoning in file sharing peer-to-peer networks. In: Proc. of EC'05, Vancouver, BC, Canada (2005) 68–77
22. Chen, H., Hai Jin, J.S., Han, Z.: Efficient immunization algorithm for peer-to-peer networks. In: 11th International Conference on High Performance Computing (HiPC 2004), Bangalore, India (2004)
23. Thommes, R., Coates, M.: Epidemiological models of peer-to-peer viruses and pollution. Technical report, Department of Electrical and Computer Engineering, McGill University (2005)
24. eMule Forum. (http://www.emuleforum.net/)
25. Gkantsidis, C., Rodriguez, P.R.: Network coding for large scale content distribution. In: IEEE Infocom 2005, Miami, FL (2005)