University of Würzburg
Institute of Computer Science
Research Report Series

Peer-to-Peer vs. Client/Server: Reliability and Efficiency of a Content Distribution Service

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

Report No. 422 May 2007

[1] University of Würzburg
Institute of Computer Science
Department of Distributed Systems
Am Hubland, D-97074 Würzburg, Germany
{hossfeld,trangia}@informatik.uni-wuerzburg.de

[2] 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

Peer-to-Peer vs. Client/Server: Reliability and Efficiency of a Content Distribution Service

Tobias Hoßfeld
University of Würzburg
Institute of Computer Science
Department of Distributed Systems
Am Hubland, D-97074 Würzburg, Germany
{hossfeld,trangia}@informatik.uni-wuerzburg.de

Kenji Leibnitz, Naoki Wakamiya, Masayuki Murata
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

Abstract

In this paper we evaluate the performance of a content distribution service with respect to reliability and efficiency. The considered technology for realizing such a service can either be a traditional client/server (CS) architecture or a peer-to-peer (P2P) network. In CS, the capacity of the server is the bottleneck and has to be dimensioned in such a way that all requests can be accommodated at any time, while a P2P system does not burden a single server since the content is distributed in the network among sharing peers. However, corrupted or fake files may diminish the reliability of the P2P service due to downloading of useless contents. We compare a CS system to P2P and evaluate the downloading time, success ratio, and fairness while considering flash crowd arrivals and corrupted contents.

1 Introduction

The volume of traffic transported over the Internet has drastically increased over the last few years. The download of multimedia contents or software packages may consist of large files imposing high requirements on the bandwidth of the file servers. In conventional systems this means that the servers must be properly dimensioned with sufficient capacity in order to service all incoming file requests from clients. On the other hand, *peer-to-peer* (P2P) technology offers a simple and cost-effective way for sharing content. Providers offering large volume distributions (e.g. Linux) have recognized the potential of P2P and increasingly offer downloads via eDonkey or BitTorrent.

In P2P, all participating peers act simultaneously as clients and as servers, and the file is not offered at a single server location, but by multiple sharing peers. Since the load is distributed among all sharing peers, the risk of overloading servers with requests is reduced, especially in the presence of flash crowd arrivals. However, this flexibility comes at a slight risk. Since the shared file is no longer at a single trusted server location, peers may offer a corrupted version of a file or parts of it. This is referred to as *poisoning* or *pollution* [1] depending on whether the decoy was offered deliberately or not. When the number of fake peers is large, the dissemination of the file may be severely disrupted. All of this leads to a trade-off consideration between high reliability at the risk of overloaded servers and good scalability where the received data may be corrupt. In this context, we define reliability as the availability of a single file over time in a disruptive environment. This is expressed by the success ratio of downloads. While in some

structured P2P network types, the disconnection or segmentation of the network topology due to node failure may influence the availability of content, we will only focus on unstructured P2P systems and assume that each peer can contact any other peer with the same probability (epidemic model). This is valid e.g. in eDonkey networks. In this case, the availability of a file is expressed by the number of sharing peers. Hence, the availability in the P2P networks we consider is predominantly influenced by the user behavior [2], like churn, willingness to share a file, or impatience during downloading.

In this paper, we investigate the trade-off between client/server (CS) and P2P file sharing using simple models. While we assume the file structure and download mechanism to be operating like in eDonkey, the model can be easily extended to any other P2P network. Our focus of interest lies hereby on the downloading time until successful completion of the file and the number of aborted downloads due to the impatience of users. With these performance metrics we can justify under which conditions a P2P network outperforms CS. In addition, fairness of the CDS is considered as well.

The paper is organized as follows. In Section 2 we briefly summarize existing work related to evaluating content distribution systems and comparing P2P with CS. Section 3 provides the models and assumptions that we impose. In Section 4, we provide numerical results for the comparison of the performance of P2P with CS in terms of success ratio, download duration, and fairness. Finally, this paper is concluded with a summary and an outlook on future work.

2 Related Work

Most studies on the performance of P2P systems as content distribution network rely on measurements or simulations of existing P2P networks. For example, Saroiu et al. [3] conducted measurement studies of content delivery systems that were accessed by the University of Washington. The authors distinguished traffic from P2P, WWW, and the Akamai content distribution network and they found that the majority of volume was transported over P2P. A comprehensive survey of different P2P-based content distribution technologies is given in [4]. In [5] a simulation study of P2P file dissemination using multicast agents is performed and the propagation under different conditions is studied. Hoßfeld et al. [6] provide a simulation study of the well-known eDonkey network and investigate the file diffusion properties under constant and flash crowd arrivals. However, most work on P2P file diffusion as those mentioned above usually do not assume any fake files from pollution or poisoning.

Han et al. [7] study the distribution of content over P2P and consider rewarding strategies as incentives to improve the diffusion. They show that the network structure in terms of hierarchy and clustering improve the diffusion over flat structures and that compensating referrers improves the speed of diffusion and an optimal referral payment can be derived. The user behavior and an analysis of the rationale in file sharing is studied in [8] using game theory. The focus lies on free riding in the network and the authors offer suggestions on how to improve the willingness of peers to share. Qiu et al. [9] model a BitTorrent network using a fluid model and investigate the performance in steady state. They study the effectiveness of the incentive mechanism in BitTorrent and prove the existence of a Nash equilibrium. Rubenstein and Sahu [10] provide a mathematical model of unstructured P2P and show that P2P networks show good scalability

and are well suited to cope with flash crowd arrivals. Another fluid-diffusive P2P model from statistical physics is presented by Carofiglio et al. [11]. Both, the user and the content dynamics are included, but this is only done on file level and without pollution. These studies show that by providing incentives to the peers for sharing a file, the diffusion properties are improved. We include appropriate paramters in our model which capture this effect, while also considering pollution.

Christin et al. [1] measured content availability of popular P2P file sharing networks and used this measurement data for simulating different pollution and poisoning strategies. They showed that only a small number of fake peers can seriously impact the user's perception of content availability. In [12] a diffusion model for modeling eDonkey-like P2P networks was presented based on an epidemic SIR [13] model. This model includes pollution and a peer patience threshold at which the peer aborts its download attempt and retries later again. It was shown that an evaluation of the diffusion process is not accurate enough when steady state is assumed or the model only considers the transmission of the complete file, especially in the presence of flash crowd arrivals.

3 Modeling the Content Distribution Service

In the following we will consider two alternative architectures for content distribution: P2P and a traditional client/server structure (e.g. HTTP or FTP server). We include pollution from malicious peers in the P2P model offering fake content. On the other hand, the client/server system is limited by the server bandwidth. In both systems we assume that the user is willing to wait only for a limited time until the download completes. If the downloading process exceeds a patience threshold, the user will abort his attempt. We will use these models to later analyze the benefits and drawbacks of each architecture.

3.1 Peer-to-Peer Network

In the P2P model we assume that the file sharing process of a file with size f_{size} operates similar to the eDonkey network. The sharing itself is performed in units of 9.5MB, so-called *chunks*, and the data of each chunk is transfered in *blocks* of 180kB. In order to make the model more tractable, we simply consider that each file consists of M download data units. After each chunk is downloaded, it is checked using MD5 hashes and in case an error is detected e.g. due to transmission errors, the chunk is discarded and downloaded again. After all chunks of a file have been successfully downloaded, it is up to the peer if the file is kept as a *seeder* for other peers to download or if it is removed from share (*leecher* or *free rider*). In this study, we only consider a file that consists of a single chunk with $M = 53$ download units which corresponds to the number of blocks in a chunk. Thus, the terms block and download/data unit will be used interchangeably.

3.1.1 Description of the Flow Model

As mentioned above, errors may occur during the download process of a chunk rendering it useless. This mechanism is used by malicious peers deliberately introducing erroneous chunks

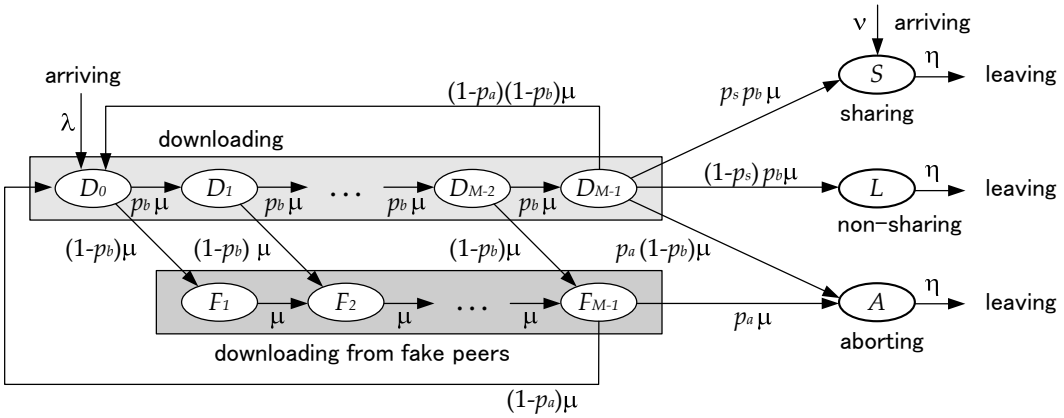


Figure 1: Flow diagram of P2P file sharing model

to the file sharing network. In order to characterize the dynamic behavior of the P2P network with K malicious peers, referred to as *fake peers*, we extend the model in [12]. This model is based on the epidemic diffusion of diseases [13] and is characterized by a differential equation system describing the transitions between each of the states a peer traverses. Initially, there are only S_0 peers in the system sharing a correct version of the file and K fake peers. Requests for downloading the file arrive with rate λ. A peer downloads M units of the file where it has the possibility of reaching a correct version of the data block with probability p_b. Since we assume an equal probability for reaching a sharing or fake peer, p_b can be given as in Eqn. (1) at time t.

$$p_b(t) = \frac{S(t)}{S(t) + K} \tag{1}$$

The population of peers with successful downloads of i units is defined as D_i. After having successfully downloaded M data units, an error check is performed and the chunk is discarded in case of an error. If the download of the entire chunk was successful, the peer either shares the file and enters population S with the *sharing probability* p_s or enters L of non-sharing peers with the complementary probability $1 - p_s$. On the other hand, if the download attempt of the chunk failed because of downloading at least one block from a fake peer, the peer aborts with probability p_a and retries the download attempt with $1 - p_a$.

The download of i data units of which at least one is corrupt is represented by state F_i. The number of fake peers K is assumed to remain constant throughout the observation period. The system model with all populations and their transitions is shown in Fig. 1. The system of

differential equations describing the dynamic behavior of each population is given in Eqns. (2-8).

$$\dot{D}_0 = \lambda + \mu\,(1 - p_a)\,[F_{M-1} + (1 - p_b)\,D_{M-1}] - \mu\,D_0 \tag{2}$$

$$\dot{D}_i = \mu\,p_b\,D_{i-1} - \mu\,D_i \qquad i = 1, \ldots, M-1 \tag{3}$$

$$\dot{F}_1 = \mu\,(1 - p_b)\,D_0 - \mu\,F_1 \tag{4}$$

$$\dot{F}_i = \mu\,(1 - p_b)\,D_{i-1} + \mu\,F_{i-1} - \mu\,F_i \qquad i = 2, \ldots, M-1 \tag{5}$$

$$\dot{S} = \nu + \mu\,p_s\,p_b\,D_{M-1} - \eta\,S \tag{6}$$

$$\dot{L} = \mu\,(1 - p_s)\,p_b\,D_{M-1} - \eta\,L \tag{7}$$

$$\dot{A} = \mu\,p_a\,[F_{M-1} + (1 - p_b)\,D_{M-1}] - \eta\,A \tag{8}$$

The other variables that have not yet been discussed are the file request rate λ and the rates for leaving the system η. Furthermore, ν is the rate of arrivals of peers that share the file which they obtained from another source than from this network. For peers in the network, we will assume *flash crowd* arrivals as $\dot{\lambda} = -\alpha\lambda$ with initial value of $\lambda(0) = \lambda_0$. Hence, the flash crowd scenario corresponds to an exponentially decreasing arrival rate with parameter α.

$$\lambda(t) = \lambda_0\,e^{-\alpha t} \tag{9}$$

For the sake of simplicity we assume that a peer decides to leave only if he either has successfully completed the download (S and L) or when he aborts the download attempt (A). In F_{M-1}, the peer may enter the population A with *abort probability* p_a or else retries the attempt. Perhaps the most important variable in the model is the download rate per data unit $\mu(t)$. We use the same approximation as in [12] which assumes that if there are enough sharers, the download bandwidth r_{dn} of a peer will be the limitation, otherwise all requesting peers fairly share the upload bandwidth r_{up} of all sharing peers, see Eqn. (10).

$$\mu(t) = \frac{M}{f_{size}} \min\left\{ \frac{r_{up}\,(S(t) + K)}{\sum_{i=0}^{M-1} D_i(t) + \sum_{i=1}^{M-1} F_i(t)}, r_{dn} \right\} \tag{10}$$

Note that all variables in the equation system are in fact functions of time resulting in a highly non-stationary behavior. Finally, it should be remarked that the continuous transition rates lead to a slight inaccuracy from non-integer population sizes which do not appear in reality, but reflect the average values.

3.1.2 Evaluation of the Download Duration

From the solution of the dynamic system in Eqns. (2-8), we can indirectly derive the transmission durations until reaching an absorbing population S, L, or A. The states S and D_i allow from Eqn. (10) the computation of the download rates per data unit $\mu(t)$. For the computation of the download duration $\delta(t)$, let us consider the start of the download attempt of a chunk at time t_0 and a series of time instants $t_1, \ldots, t_M$. Each t_i indicates the time at which the downloading of one data unit is completed. Since the transmission rates are with respect to the transmission of a

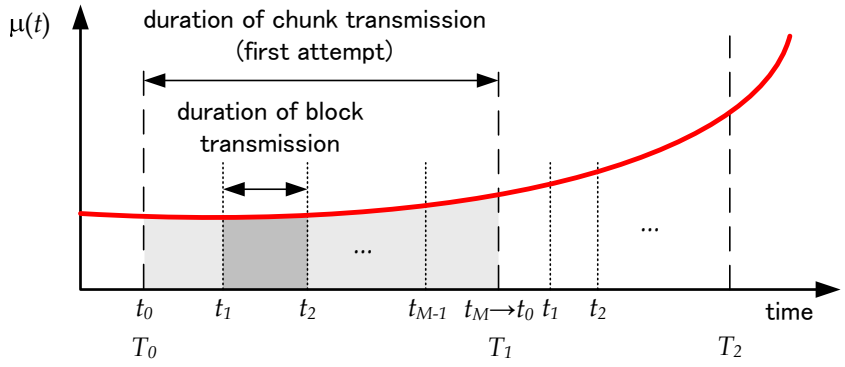


Figure 2: Computation of block and chunk transmission durations from $\mu(t)$

block, the t_i values can be computed by numerical solution of Eqn. (11) for a given t_0.

$$\int_{t_{i-1}}^{t_i} \mu(t)\, dt = 1 \qquad\qquad 1 \leq i \leq M \tag{11}$$

Once the whole chunk is downloaded, we also define this time instant as T_j, $j > 1$ indicating with j the number of attempts a download attempt was made starting at T_0. Thus, t_0 is always set to the starting time of a new chunk download and is considered only within the context of a chunk. The relationship between $\mu(t)$, t_i, and T_j is illustrated in Fig. 2.

At time instants T_j we compute the probability that the chunk was correctly received by considering the possibilities of encountering a fake source at all t_i. The probability for a correct block $p_b(t_i)$ at the start of each block download interval $[t_i, t_{i+1}]$ and the probability of the chunk being correctly received is the product over each of the correct block probabilities beginning at t_0.

$$p_c(t_0) = \prod_{i=0}^{M-1} p_b(t_i) \tag{12}$$

If the chunk was not successfully downloaded, the peer chooses to retry its attempt with probability $1 - p_a$. The average successful download duration $\delta(t)$ is then computed considering $p_c(t)$ and p_a. If we define the random variable of trials $X_s(T_0)$ needed for successfully completing the download which started at T_0 after the j-th download attempt, we obtain the probabilities in Eqn. (13).

$$\begin{aligned} P\left(X_s(T_0) = 1\right) &= p_c(T_0) \\ P\left(X_s(T_0) = j\right) &= (1 - p_a)^{j-1}\, p_c(T_{j-1}) \prod_{k=0}^{j-2} \left(1 - p_c(T_k)\right) \qquad j \geq 2 \end{aligned} \tag{13}$$

The average time until successfully completing the chunk download which the peer started at time T_0 follows then as shown in Eqn. (14). The probabilities for $X_s(T_0)$ must be normalized by

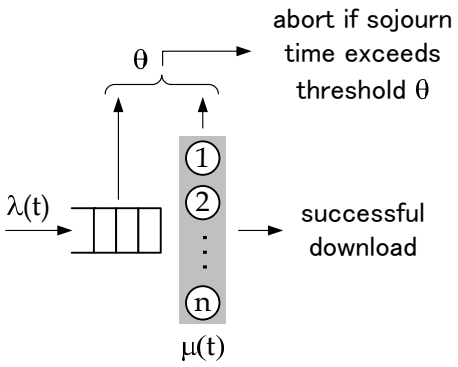


Figure 3: Model of server-based content distribution system

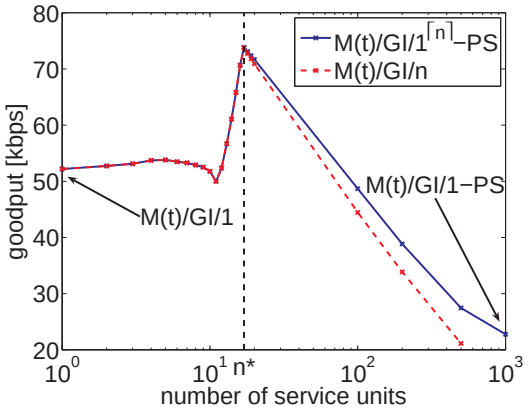


Figure 4: Influence of the number of servers on the download bandwidth of the clients

all possible realizations in order to only take the successful download completions into account.

$$\delta(T_0) = \sum_{j=1}^{\infty} (T_j - T_0) \frac{P\left(X_s(T_0) = j\right)}{\sum_{k=1}^{\infty} P\left(X_s(T_0) = k\right)} \tag{14}$$

3.2 Client/Server System

In order to compare the performance of P2P and a server-based system, we need to match the conditions like the available capacity of the system and aborted downloads. A server in the Internet, e.g. HTTP or FTP server, transfers the complete file and does not split it into chunks. Hence, the client behavior must be modeled in a different way from P2P w.r.t. aborting the download. The server model in this paper needs to consider impatient users which cancel their downloading attempt if the total sojourn time in the system exceeds an impatience threshold θ. For comparing the CS system with a P2P system, θ can be obtained from the P2P system for a given abort probability p_a using the sojourn time of peers until they abort the download. With an abuse of the Kendall-notation, we will denote the server system as $M(t)/GI/1^{\lceil n \rceil} - PS$, see Fig. 3. The queue length at the server is assumed to be infinite.

$M(t)$ means that requests arrive at the server with a non-stationary Poisson process using the flash crowd arrival rate $\lambda(t)$ described in Eqn. (9). The system itself has a total constant capacity $C = S_0\, r_{up}$ which corresponds to the total bandwidth available in the P2P system at time $t = 0$.

We assume that the complete bandwidth C is split equally among all downloading clients with the processor sharing discipline. However, the number $D(t)$ of simultaneously served clients is restricted to a maximum n. Each client is served by one virtual service unit and is guaranteed a minimal offered download bandwidth of C/n. If less than the maximum number of n service units (or parallel download slots) are actually occupied, a client receives $C/D(t)$. Thus, the average service rate of the system is then either limited by the bandwidth that each simultaneously downloading client gets or the maximum client download bandwidth r_{dn} as given in Eqn. (15).

We use $1^{\lceil n \rceil} - PS$ in the notation for the server model to describe this service behavior.

$$\mu(t) = \min\left\{r_{dn}, \frac{C}{D(t)}\right\} \tag{15}$$

The service requirement follows a general distribution GI and describes the sizes of the files to be downloaded. As we consider only a single file with a fixed size, the corresponding system is $M(t)/D/1^{\lceil n \rceil} - PS$.

The impact of the number of service units n on the average goodput each user experiences is illustrated in Fig. 4. The goodput is the ratio between the file size and the sojourn time of a user, where latter is the sum of the service time and the waiting time. This figure also shows the equivalent curve for an $M(t)/GI/n$ system with rate $\mu = C/n$. In the processor sharing model, if $n < C/r_{dn}$, the downlink of the client is the bottleneck in the system and $M(t)/GI/1^{\lceil n \rceil} - PS$ is equivalent to the $M(t)/GI/n$ system. For $n > C/r_{dn}$, both systems show a different behavior. The processor sharing discipline utilizes the entire capacity C and can therefore be seen as the best case scenario in terms of bandwidth efficiency. From Fig. 4, we can also recognize the existence of a maximum value at $n^* = \lceil C/r_{dn} \rceil$ where the highest efficiency can be found. While for $n < n^*$ the average bandwidth is limited by the client download bandwidth, for $n > n^*$ the capacity of the server is the limiting factor. Note that for $n = \frac{\lambda_0}{\alpha}$, the system results in a pure $M(t)/GI/1 - PS$ queue, as the total number of arriving users in the system is limited in the considered flash-crowd scenario: $\lim_{t\to\infty} \int_0^t \lambda(t)dt = \frac{\lambda_0}{\alpha}$.

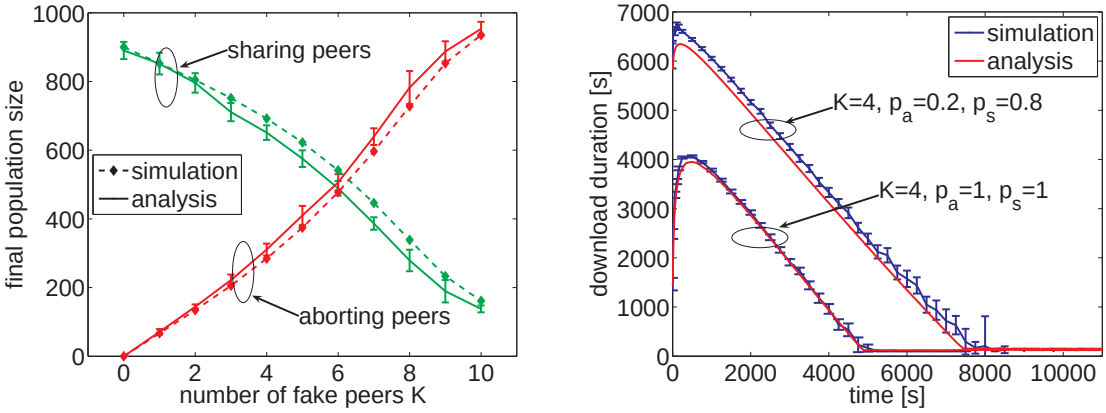
4 Numerical Results

We will now show numerical results and compare the P2P and CS system in performance. Unless stated otherwise, we will make the following assumptions as summarized in Tab. 1. Note that with $\eta = 0$, $\nu = 0$, and the limited number of arrivals ($\lim_{t\to\infty} \lambda(t) = 0$), all peers remain in the system after their either successful or unsuccessful download attempt. Therefore, the populations S, L, and A increase monotonically. The capacity of CS is C =12.8Mbps. Due to the complexity of the CS system and since we focus on the performance of P2P, we will provide numerical results for the client/server system by simulation.

We investigate the influence of the maximum number n of parallel downloads at the server, the number K of fake peers, and the user's patience θ on the number of successful downloads and the expected download time. The download has to be finished within the time θ the user is

Table 1: Default parameters for evaluation of P2P and CS system

general parameters			**P2P parameters**		
file size	f_{size}	9.5MB	initial sharing peers	S_0	100
upload bandwidth	r_{up}	128kbps	seeder arrival rate	ν	0
download bandwidth	r_{dn}	768kbps	departure rate	η	0
initial arrival rate	λ_0	1	sharing probability	p_s	0.8
flash crowd decay	α	10^{-3}	abort probability	p_a	0.2



(a) Population sizes

(b) Download durations

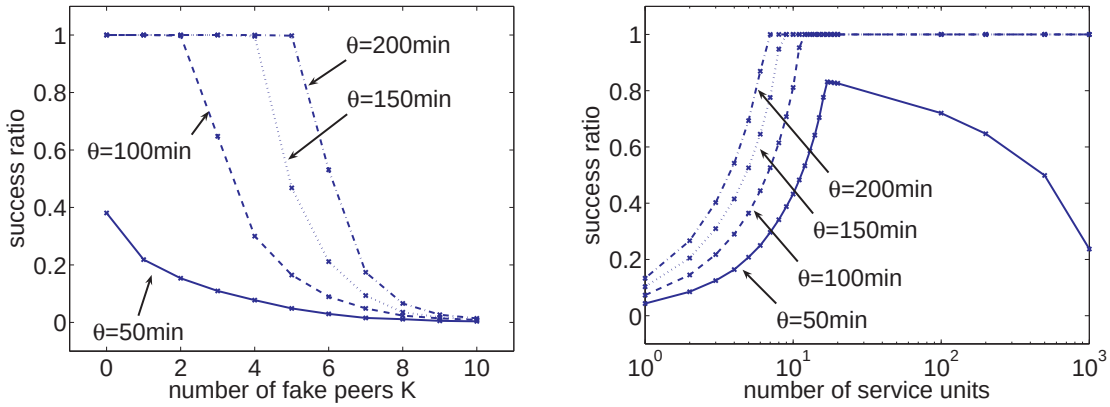
Figure 5: Comparison of simulation results with analytic flow model

willing to wait. In addition, fairness of the CDS has to be considered as well. In a fair system each user experiences a similar download duration like others.

4.1 Evaluation of the P2P Flow Model

First, we validate the analytical P2P flow model with simulation. Fig. 5(a) shows the final average population sizes of sharing peers and aborting peers over the number of fake peers K, when the whole population is in the absorbing states, S, L, A. The values are obtained from 20 simulation runs and error bars represent the 99% confidence intervals. The analysis matches the simulations well with only slight differences due to the underlying Markovian assumption at state transitions. The accuracy can be increased by inserting additional intermediate states at the cost of a higher computational complexity for solving the equations. Fig. 5(a) shows that a small number of $K = 10$ fake peers is almost sufficient to prevent any peer from completing the download.

Since we consider a non-stationary system, the download duration varies over time according to the current system state. Fig. 5(b) shows the average duration of a peer as function of the starting time of the download for $K = 4$. The analytical result is computed directly from Eqn. (14) and compared to values obtained from 20 simulation experiments. In both scenarios with different abort and sharing probabilities, the curves show a good match. The flash crowd arrival causes in both cases a strong increase with a linear decrease and in the case of no retrials and altruistic users ($p_a = 1$, $p_s = 1$), the duration is significantly smaller since peers only attempt to download the file once. On the other hand, when $p_a < 1$ the number of trials has an average greater than one resulting in longer download durations, see Eqn. (13).



(a) Peer-to-Peer (b) Client/Server

Figure 6: Comparison of success ratio between P2P and CS

4.2 Success Ratio

The performance of P2P and CS is now compared regarding the success ratio, i.e., the ratio of successful downloads to the sum of successful and aborted downloads. In order to make a fair comparison, we now use a deterministic patience threshold $\theta = 50, 100, 150, 200$ minutes after which a user in both systems cancels the download. The success ratio in P2P is 100% for $\theta > 50$ and small K, see Fig. 6(a). However, when K increases from 6 to 7, the success ratio with $\theta = 200$ reduces to about 50% and for even larger K no peer completes the download. Fig. 6(b) shows the equivalent results for CS as function of the number of service units n. Except when n is too small, the success ratio lies above that of P2P for each θ, especially when the optimal value n^* is chosen.

We conclude that the client/server system has at least the success ratio of P2P, if the client bandwidths are known a priori for dimensioning the optimal number of service units. The P2P system strongly suffers from the presence of too many fake peers.

4.3 Download Duration

The key performance indicator from the user's viewpoint is the overall download duration, i.e., the interval from the request of a file until its successful download. In Fig. 7(a), the time for successful downloads and the sojourn time of aborted downloads is depicted. Since the patience time is deterministic, the abort time is given as straight lines for each θ. The lines begin at values of K where the success ratios become less than 1. The successful download duration increases with K until impatience manifests itself in increased canceled downloads. Peers beginning their download later benefit from this effect. As a result the mean download time stays constant or even decreases again with K and the 99%-confidence intervals from the simulation runs increase due to the decreasing number of successful downloads which can be used to compute the averages.

The results in Fig. 7(b) show that well dimensioned systems show the best download perfor-

mance. However, if the optimal capacity is a priori unknown, the P2P system outperforms the server as the capacity of P2P increases with the number of sharers. If the peers behave altruistic, the P2P system has its advantages and might cope with even more extreme flash crowds, which will crash a server with fixed capacity. The P2P system mainly benefits from incentives and its multiple source technique when sharing already received chunks to other peers, thus fostering the cooperation among peers [2].

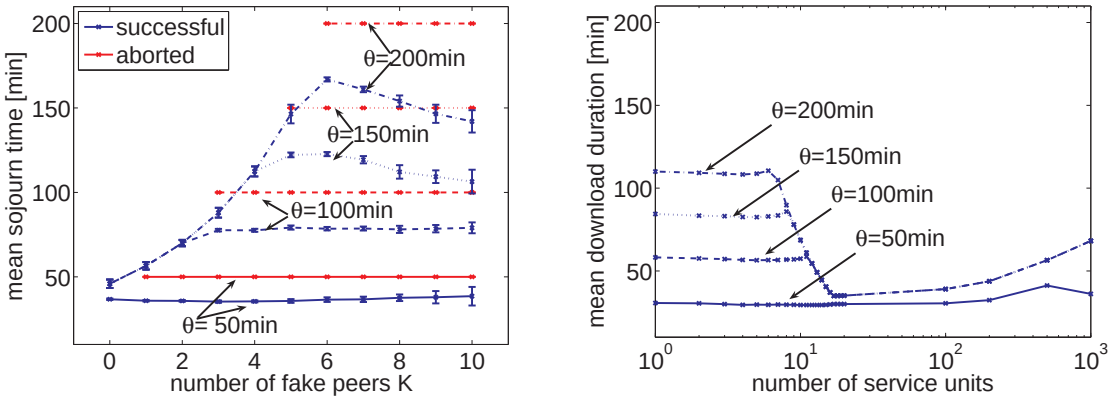
4.4 Fairness

We choose the fairness indicator $J = \left(1 + c_x^2\right)^{-1}$ given in [14] which returns values between 0 and 1. Low values of the fairness index indicate an unfair system, while a fairness index of one describes a completely fair system, where all users experience exactly the same download time. The term c_x^2 is the coefficient of variance of the considered performance measure x, which is the download time a user experiences. Independent of the number of fake peers K or the patience time θ, the P2P system is a more fair system with higher fairness index above 0.9, cf Figure 8(a). On the other hand, CS reaches such fairness only for very large n in Figure 8(b). In that case, the average download time, however, is larger than in the P2P system (for a small number of fake peers).

We can conclude that a well dimensioned CS with a priori knowledge of the clients' bandwidths outperforms P2P at the cost of fairness. Furthermore, we could see that the influence from only few fake peers is sufficient to severely cut down the performance of the P2P system.

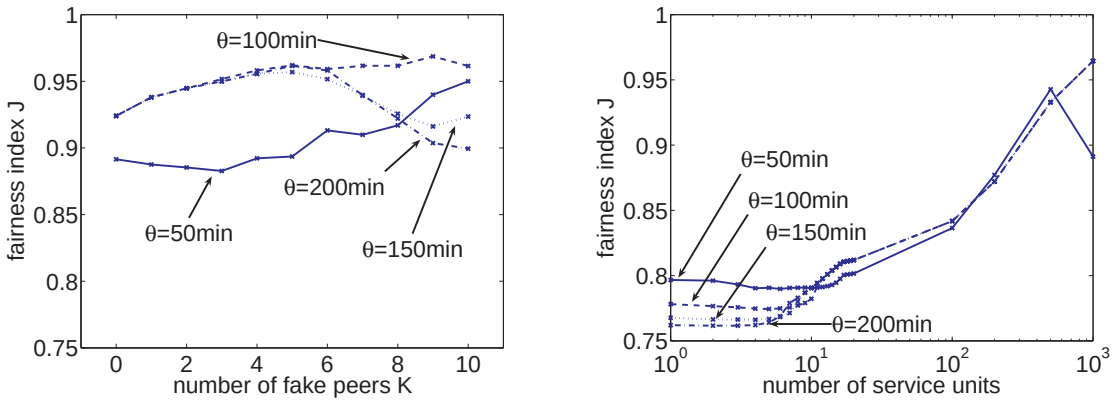
5 Conclusion

In this paper we presented a flow model for a P2P file sharing network and compared its performance to a client/server system. While in general it is not easy to compare both types of networks due to their inherently different structures, we could qualitatively investigate both architectures under comparable situations.



(a) Peer-to-Peer (b) Client/Server

Figure 7: Durations of successful downloads



(a) Peer-to-Peer (b) Client/Server

Figure 8: Fairness index of successful downloads w.r.t. download duration

Basically, when it comes to the reliability, servers are the better choice, as manipulated data is not being injected into the network. However, from the view of the end user, the same effect may be experienced when downloading from a trusted server as with P2P networks with pollution or poisoning. Especially, when the request arrival rate is high, the waiting time until the download can be processed or its duration may become too long. The problems in CS performance can be overcome by adding further server capacity.

On the other hand, P2P systems can be easily made inoperable when many fake sources exist. If the initial number of sources is small there is a risk of these peers leaving the system which would make the network lose content due to churn. For this reason, it is important that incentives are being provided to peers to increase the willingness to share the data. Enhanced error detection mechanisms must be provided to reduce the number of retransmission in case of errors. This could be done in combination with a caching peer which acts like a server but whose content is being determined by the requests of the peers.

References

[1] N. Christin, A. S. Weigend, and J. Chuang, "Content availability, pollution and poisoning in file sharing peer-to-peer networks," in *Proc. of ACM EC'05*, (Vancouver, BC), 2005.

[2] D. Schlosser, T. Hoßfeld, and K. Tutschku, "Comparison of robust cooperation strategies for P2P content distribution networks with multiple source download," in *Proc. of IEEE P2P2006*, (Cambridge, UK), Sep. 2006.

[3] S. Saroiu, K. P. Gummadi, R. J. Dunn, S. D. Gribble, and H. M. Levy, "An analysis of internet content delivery systems," in *Proc. of USENIX OSDI 2002*, (Boston, MA), Dec. 2002.

[4] S. Androutsellis-Theotokis and D. Spinellis, "A survey of peer-to-peer content distribution technologies," *ACM Computing Surveys*, vol. 36, pp. 335–371, Dec. 2004.

[5] K. G. Zerfiridis and H. D. Karatza, "File distribution using a peer-to-peer network–a simulation study," *Journal of Systems and Software*, vol. 73, no. 1, pp. 31–44, 2004.

[6] T. Hoßfeld, K. Leibnitz, R. Pries, K. Tutschku, P. Tran-Gia, and K. Pawlikowski, "Information diffusion in eDonkey-like P2P networks," in *ATNAC 2004*, (Bondi Beach, Australia), 2004.

[7] P. Han, K. Hosanagar, and Y. Tan, "Diffusion of digital products in peer-to-peer networks," in *25th Intern. Conf. on Information Systems*, (Washington, DC), Dec. 2004.

[8] J. U. Becker and M. Clement, "The economic rationale of offering media files in peer-to-peer networks," in *Proc. of HICSS-37*, (Hawaii, HI), January 2004.

[9] D. Qiu and R. Srikant, "Modeling and performance analysis of BitTorrent-like peer-to-peer networks," in *ACM SIGCOMM'04*, (Portland, OR), August 2004.

[10] D. Rubenstein and S. Sahu, "Can unstructured P2P protocols survive flash crowds?," *IEEE/ACM Trans. Netw.*, vol. 13, no. 3, pp. 501–512, 2005.

[11] G. Carofiglio, R. Gaeta, M. Garetto, P. Giaccone, E. Leonardi, and M. Sereno, "A statistical physics approach for modelling P2P systems," in *ACM MAMA'05*, (Banff, Canada), June 2005.

[12] K. Leibnitz, T. Hoßfeld, N. Wakamiya, and M. Murata, "On pollution in eDonkey-like peer-to-peer file-sharing networks," in *Proc. of GI/ITG MMB 2006*, (Nuremberg, Germany), March 2006.

[13] J. Murray, *Mathematical Biology, I: An Introduction*. Springer, 3 ed., 2002.

[14] R. Jain, D. Chiu, and W. Hawe, "A Quantitative Measure of Fairness and Discrimination for Resource Allocation in Shared Systems," Tech. Rep. DEC TR- 301, Digital Equipment Corporation, 1984.