# On the Impact of Incentives in eMule
## Analysis and Measurements of a Popular File-Sharing Application

Damiano Carra, Pietro Michiardi, Hani Salah, and Thorsten Strufe, *Member, IEEE*

***Abstract*—Motivated by the popularity of content distribution and file sharing applications that nowadays dominate Internet traffic, we focus on the incentive mechanism of a very popular, yet not very well studied, peer-to-peer application, eMule.**

**In our work, we recognize that the incentive scheme of eMule is more sophisticated than current alternatives (e.g., BitTorrent) as it uses a general, priority-based, time-dependent queuing discipline to differentiate service among cooperative users and free-riders. In this paper, we describe a general model of such an incentive mechanism and analyze its properties in terms of application performance. We validate our model using both numerical simulations (when analytical techniques become prohibitive) and with a measurement campaign of the live eMule system.**

**Our results, in addition to validating our model, indicate that the incentive scheme of eMule suffers from starvation. Therefore, we present an alternative scheme that mitigates this problem, and validate it through numerical simulations and a second measurement campaign.**

***Index Terms*—Dynamic Priority, Performance Evaluation**

## I. Introduction

Consumption of digital content is one of the most popular uses of the Internet, involving millions of end-hosts: content distribution services, such as direct download/streaming sites using One-Click Hosting (OCH) [1] and peer-to-peer (P2P) applications dominate Internet traffic [2], [3], [4], [5]. The popularity of such services and applications, and their impact on Internet traffic, has attracted a lot of attention in the past decade: the literature is rich of extensive studies of P2P applications – in particular of BitTorrent – and OCH [6], with the goal of measuring [7], understanding, and modeling their performance [8], [9], [10], [11].

In addition to the scaling properties and their performance, an integral part of such services is the presence of incentive mechanisms to combat "free-riders", users, who do not offer local resources (bandwidth and storage) but make the most of the contributions of the mass. Although incentive mechanisms are very important for P2P applications, they are also adopted in OCH services to create differentiation between unsubscribed and premium clients. A prominent example of incentive schemes – or variations thereof – that has received a lot of attention is that of BitTorrent [12].

D. Carra is with the Computer Science Dep., University of Verona, Italy. E-mail: damiano.carra@univr.it

P. Michiardi is with EURECOM, Sophia Antipolis, France. E-mail: pietro.michiardi@eurecom.fr

H. Salah and T. Strufe are with Computer Science Dep., TU Darmstadt, Germany. E-mail: {hsalah,strufe}@cs.tu-darmstadt.de

Manuscript received March 1, 2012; revised July 23, 2012.

This work has been partially supported by the project ANR-09-VERSO-014 VIPEER, the IT R&D program of MKE/KEIT of South Korea [10035587], and the DAAD (grant no. A/09/97120). The authors would like to thank Ghannam Aljabari from Palestine Polytechnic University for his invaluable support to the experiments, and the anonymous reviewers for their comments that allowed improving the quality of our work.

The endeavor of this work is to focus on eMule/aMule [13], [14], another file-sharing application that is very popular among users. Recent studies indeed have shown that it is used by millions [15], [16], only counting the peers that participate in the KAD network and disregarding clients that are solely connected through the E2DK network, but it has much less been studied in literature. Specifically, we investigate the built-in incentive mechanism as it is substantially different from those implemented in other P2P applications. However, the proposed model could be used in other contexts, including OCH services and various kinds of scheduling problems (e.g., operating systems and parallel processing systems). Instead of having short-term goals, as in BitTorrent, the incentive mechanism in eMule is content-oblivious: users are granted credits (using a fairly complex procedure) that are used to gain service from other peers across multiple contents.

In this paper, we first recognize that the incentive scheme of eMule is a special instance of a more general scheduling mechanism, that awards resources (in the context of eMule, upload slots) using a time-based priority queueing discipline. We call this scheme a *proportional differentiation* mechanism and propose a model to study its properties under realistic settings. That is, we assume finite-capacity queues and include churn, a characteristic trait of P2P applications where peers may join or leave the system at any time. Our model is validated both with numerical simulations and with an extensive measurement campaign on the current deployment of the eMule/aMule system.

Backed by our findings, we realize that the current implementation of the incentive scheme in eMule suffers from *starvation*: peers with little resources may have to wait for a long time before being served by other peers. We thus propose an alternative mechanism (that we call *additive differentiation*), which mitigates starvation while maintaining the flexibility of the original proportional mechanism in tuning service differentiation using a handful of parameters. Finally, we validate the additive scheme using numerical simulations and another measurement campaign in which we deploy a modified aMule client that implements our incentive mechanism.

## II. The eMule Incentive System

The motivation for eMule [13], [17] peers to use a priority scheme for awarding upload slots to remote peers stems from the fact that peers may behave selfishly and *free-ride* on system resources. As such, the priority scheme is effectively an incentive mechanism that aims at fostering peer cooperation. However, unlike other popular file-sharing applications such as BitTorrent [12], which implements an instantaneous mechanism akin to the tit-for-tat scheme, time in eMule plays an important role.

Each peer in eMule records the volume of data exchanged (download and upload) with every other peer it has interacted with in the past. The combination of these two values is referred to as *credits*. Such credits are used to assign the priority that remote peers will be granted for each content request. Note that credits are accumulated by each peer independently of the requested or served content. Credits associated to an uploading peer are stored locally, and not at the credited peer itself.

A peer in eMule implements a time-dependent priority discipline *with preemption*. For a generic request $j$ received from a remote peer, its priority over time is computed as follows:

$$q_j(t) = \big(t - T_{\text{arrival}} + T_{30}\mathbb{I}_s(t)\big) \cdot f_p \cdot C_j(t) \tag{1}$$

where $T_{\text{arrival}}$ is the arrival time of the request, $t \geq T_{\text{arrival}}$, $T_{30}$ is a constant equal to 30 minutes, $\mathbb{I}_s(t)$ is the indicator function for the service – which takes the value 1 if the request is in service, and 0 otherwise – $f_p$ is a constant value associated to each file, and $C_j(t)$ is the priority coefficient for that specific request (derived from the credits), which varies over time.

Pending requests may change priority class while they are waiting to be served (or even while they are being served). Indeed, the coefficient $C_j(t)$ is computed as follows:

$$C_j(t) = \max\left(1, \min\left(\frac{2U(t)}{D(t)}, \sqrt{U(t)+2}, 10\right)\right)$$

where $U(t)$ and $D(t)$ depict the total volume of data (expressed in MBytes) respectively uploaded and downloaded at time $t$ by the peer that issued the request $j$, as tracked by the peer currently acting as a single server queue for that particular request $j$. In eMule, the constant $f_p$ can take one of the following values: 0.2, 0.6, 0.7, 0.9, 1.8. As a result of the "min" and "max" operations, it holds that $1 \leq C_j(t) \leq 10$.

The number of requests a client can accept is limited. There are mainly two parameters that control this limit: the maximum number of connections and the maximum number of download requests. The maximum number of connections imposes a limit on the number of TCP connections (one connection per client). Once the client is connected, it can send a download request for a specific file – note that only one request at a time is accepted in the waiting queue from a given peer. Given these two parameters, it is hard to understand which one mainly limits the system, since they can be both changed by the users. As we will see in the next Section, we will consider a single limiting parameter $k$, without specifying if it represents the maximum number of connections or the maximum number of download requests. A measurement campaign will show that, in most of the cases, there is no connection slot available, therefore $k$ will be interpreted as the maximum number of connections.

## III. Performance Modeling

In this section, we provide a simple model that can be used to evaluate the impact of the eMule incentive scheme on the system performance, measured in time a request spends in the system. The model is represented by a single server queue with a finite buffer and a scheduling discipline based on a dynamic priority. For such a simple model, we provide a set of interesting, original results.

### A. Time-Dependent Priority

We consider a $M/M/1/k+1$ queue, where jobs, which hereafter we call *requests*, arrive according to a Poisson process, and their service times are exponentially distributed. Although the assumption of exponential service times is unrealistic from a practical perspective, it greatly simplifies the analysis from the theoretical point of view. We will see, using a the numerical approach, that the impact of this simplification is not significant.

The single server queue allows $P$ different priority classes (or groups): requests for group $i$ ($i = 1, 2, \ldots, P$) arrive according to an independent Poisson processes with rate $p_i\lambda$, where $\lambda$ is the total arrival rate and $p_i$ is the probability that the requests belong to group $i$, with $\sum_i p_i = 1$. The request processing time is exponentially distributed with parameter $\mu_i = \mu \forall i$. We note that this assumption of a unique service rate $\mu$ reflects a system in which the requests arriving from different priority classes concern the same set of "objects", and thus the service rate is the same, independently of class $i$.

We define:

$$\rho_i = \frac{p_i\lambda}{\mu}, \qquad \rho = \frac{\lambda}{\mu} \qquad \text{and} \quad W_0 = \frac{\rho}{\mu},$$

where $W_0$ is the expected completion time for the request (job) in service.

In contrast to the usual convention, we assume that a request $i$ has priority over another request $j$ if its priority value is larger than the priority value of request $j$. We assume that requests do not leave the system until they are served.

With a time-dependent discipline, the priority of a request depends both (i) on the specific group it belongs to and (ii) on the amount of time spent by such requests in the system. As such, these schemes have the desirable property that request *starvation* is not present (if $\rho < 1$): indeed, as time progresses, the priority of a request grows, and it is eventually served by the system. The single server queue executes a simple scheduling process that selects the next request to be served based solely on its instantaneous priority.

Let $T_{\text{arrival}}$ be the arrival time of a request and let $T_{\text{leave}}$ be the time when the request leaves the system. We consider a class of priority schemes in which the priority $q_i(t)$ at time $t$ assigned to a request belonging to group $i$ is given by the following general expression:

$$q_i(t) = b_i(t - T_{\text{arrival}})^r + a_i, \tag{2}$$

with $T_{\text{arrival}} \leq t \leq T_{\text{leave}}$. Each priority group can be identified by the coefficients $b_i$ and $a_i$, with $i = 1, 2, \ldots, P$, $0 < b_1 \leq b_2 \leq \ldots \leq b_P$ and $0 < a_1 \leq a_2 \leq \ldots \leq a_P$ .

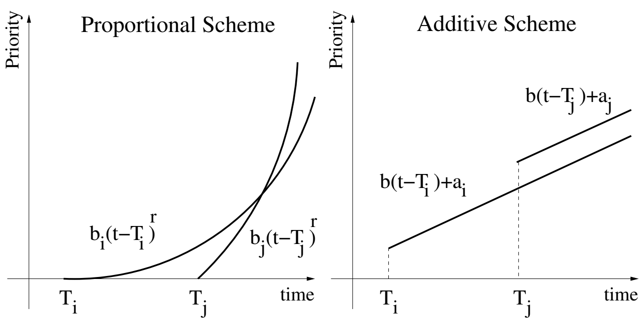


Fig. 1. Examples of the evolution of the priority in case of proportional and additive schemes.

In Figure 1, we show two different cases. On the left hand side, we show the case where $a_i = 0$ , $\forall i \in P$ and $r > 1$: the priority over time of the requests follows a convex function. In case of $r < 1$ we have a similar behaviour, with concave functions. We label this approach the *proportional scheme*.

On the right hand side, we show the case where $b_i = b$, $\forall i \in P$ and $r = 1$. The difference in terms of priority between two requests remains constant over time. We call this approach the *additive scheme* (the reasons for these names, proportional and additive scheme, will become clear later in the paper).

### B. Proportional Scheme: Main Results

We consider the case where the coefficients $a_i$ in Eq. 2 are all set to zero, *i.e.*, we consider the proportional scheme. The literature is rich of studies that consider $M/M/1$ or $M/G/1$ single server queues that execute a variety of priority queueing disciplines [18], [19], [20], [21]. However, prior works mainly focus on systems with an infinite buffer size. Instead, in this work we are interested in studying applications under the more realistic assumption of limited buffers.

We focus on *closed systems*, where the number of requests inside the system is constant (equal to $k+1$), and a new request is accepted only when the request in service leaves the system. In this case the request arrival rate equals the service rate, *i.e.*, $\lambda = \mu$. Since the request processing time is exponentially distributed, the arrival process is still Poisson. The group of the new arrival is independent from the group of the request that has completed the service.

Closed systems represent an analytically tractable approximation of the *heavy traffic* regime, *i.e.*, a regime where the offered load approaches the service rate. We are interested in the heavy traffic regime, which is the one eMule operates in: the request rate to access content approaches or is larger than the service rate a peer can offer (cf. Sec. V).

**Service without Preemption:** In case of service without preemption, once a request has been scheduled, the next request will be scheduled only when the current request has been fully served.

The authors in [22] have found an interesting relation in case of a $M/G/1$ queue (*i.e.*, with infinite buffer) and heavy traffic regime (with an additional constraint, *i.e.*, the parameter $r$ is set to one): the ratio between mean waiting times of two classes depends on the ratio of the priority coefficients, *i.e.*,

$$\frac{W_i}{W_j} \rightarrow \frac{b_j}{b_i}.$$

In the following Theorem, we extend this result in case of closed systems and without constraints for the parameter $r$. Moreover, we provide a simple way to compute the mean waiting times $W_i$ for each class.

*Theorem 1:* Given any two priority groups $i$ and $j$, the mean waiting times $W_i$ and $W_j$, in case of a non preemptive service, in closed systems, satisfies the following condition:

$$\frac{W_i}{W_j} = \left(\frac{b_j}{b_i}\right)^{1/r}. \tag{3}$$

The mean waiting times can be computed as:

$$W_i = \frac{1}{b_i^{1/r}} \frac{k}{\mu} \frac{1}{\sum_{i=1}^{P} \frac{p_i}{{b_i}^{1/r}}}. \tag{4}$$

*Proof:* See Appendix A. ■

Theorem 1 indicates that, independently from the traffic composition (*i.e.*, the values of $\rho_i$), a time-dependent priority discipline provides a *proportional differentiated service* that depends on $r$ and the coefficients $b_i$ and $b_j$.

The theorem is interesting because it also shows the relation between the mean waiting times and the parameters of the system ($k$, $\mu$, $r$ and $b_i$) that can be tuned by the system administrator.

**Service with Preemption:** We now consider the case in which the service to any request can be suspended by a new request that, as time progresses, has gained a higher priority than the currently scheduled one. The suspended request can be resumed if its priority returns to be the highest.

Let $T_i$ be the mean time spent in the single server queuing system by a request belonging to priority class $i$, *i.e.*, $T_i = E[T_{\text{leave}} - T_{\text{arrival}}]$. Clearly, we have that $T_i = W_i + 1/\mu$, where $W_i$ is the mean waiting time for a request in the class $i$.

The following result holds:

*Theorem 2:* Given any two priority groups $i$ and $j$, the mean times spent in the system, $T_i$ and $T_j$, in case of a preemptive service, in closed systems, satisfies the following condition:

$$\frac{T_i}{T_j} = \left(\frac{b_j}{b_i}\right)^{1/r}. \tag{5}$$

The mean time spent in the system can be computed as:

$$T_i = \frac{1}{b_i^{1/r}} \frac{k+1}{\mu} \frac{1}{\sum_{i=1}^{P} \frac{p_i}{{b_i}^{1/r}}}. \tag{6}$$

*Proof:* See Appendix B. ■

To the best of our knowledge, this result, or part of it, has not been found before, not even in the case of infinite buffers.

### C. Relevance of the Main Results

The main results discussed above can be applied to characterize the performance of eMule. In particular, we shall consider Theorem 2, next, since eMule clients offer a service with preemption.[1]

Considering Eq. 1, let's neglect the term $T_{30}\mathbb{I}_s(t)$ to simplify the expression, and assume $f_p = 0.7$ for each file (this is the default value in eMule). In this case, Theorem 2 indicates that if the mean download time for a request with top priority is $T_H$, then the mean download time for a request with the lowest possible priority will be $T_L = 10T_H$ (since the ratio between the maximum possible value and the minimum possible value of $C_j(t)$ is 10). The parameter $k$ represents the maximum number of connections, and it may be used to tune the absolute values of the mean time spent in the system.

In practice, however, the eMule system is more complex than the model we presented so far. A more realistic model should include the ability of eMule to allow parallel uploads among $Q$ slots; moreover, peer churn (dynamic departure of peers, along with their requests) should also be accounted for in the model. In the following, we show that the simple model presented in this Section is able to predict the performance of a more complex model (which can be solved only numerically). In other words, we show that the impact of some parameters, such as $Q$ of the churn, is negligible; this in turn suggests that we can use the simple model to study the eMule system.

## IV. Numerical validation of the model

We consider the following three modifications to the single server model described in Sec. III-A:

1. We allow the service rate to be generally distributed. This means that, with a closed system, the arrival rate is generally distributed too.
2. The system serves $Q$ requests in parallel, giving to each of them a service rate equal to $\mu/Q$. The system has a waiting line of $k$ positions, therefore the total number of requests in the system is $k + Q$.
3. The requests in the waiting queue can leave the system at any time. In particular, the requests are active for a random interval, which is generally distributed. When they become inactive, they leave the waiting queue – in practice, when the client that has issued the request disconnects, its request is discarded by the system. We refer to this behavior as "churn." Since we have a closed system, if a position becomes available, it is immediately occupied by a new request. Note, that this has an impact on the arrival process. The new arrival belongs to class $i$ with probability $p_i$, independently from the class of the request that has left the system.

The enhanced model, is hard, if not impossible, to solve analytically. We therefore revert to a numerical solution applying a dynamic Monte Carlo simulation. In particular, we make use of the Stochastic Simulation (also known as Gillespie algorithm). In practice, the model (which is a Markov process) is simulated for a sufficiently long time, and then the statistics of interest are taken. The approach is interesting since, given a performance index, it is possible to estimate not only the mean, but also the whole probability distribution; the error in the estimation can be decreased to a desired level with the usual statistical techniques (multiple runs, with evaluation of confidence intervals).

[1] For the sake of completeness, we also presented the non-preemptive case, which can be useful to model other incentive schemes such as the ones used in OCH services.

In the following, we show the numerical solutions obtained with stochastic simulations and compare them with the theoretical results obtained in Sec. III-B. We observe that part of the theoretical results are able to predict the performance of the enhanced model with the three modifications explained above. We will show only some representative results for the preemptive case, but we have obtained similar results for the non-preemptive case and with many different settings (e.g., with many different service time distributions), that we omit from this paper due to lack of space.

We consider four classes with parameters $b_i$ equal to 1, 2, 4 and 10 respectively, and equal probability, i.e., $p_i = p = 0.25$. We set the parameter $r$ (see Eq. 2) to 1. The buffer size is $k = 1000$ and the service rate for requests is Weibull distributed with scale parameter $\mu = 10$ and shape parameter $s$. We consider a Weibull distribution since, by changing the shape parameter, it is possible to obtain a light tailed distribution ($s > 1$), a heavy tailed distribution ($s < 1$) or an exponential distribution ($s = 1$).

We start validating our stochastic simulation solver against the main theoretical results (Theorem 2), i.e., we use a shape parameter $s = 1$ to obtain an exponential distribution, the system serves $Q = 1$ request at a time, and we have requests that are always active (no churn). Table I shows the mean time spent in the system by the requests belonging to different classes. The second and the third column show the absolute values of the $T_i$s, theoretical and simulated (with the corresponding 98% confidence interval). Moreover, the last column shows the ratio between $T_1$ and $T_i$: considering class 1 as the reference class, the ratio should be equal to $b_i/b_1$. Since $b_1 = 1$, the ratio should be equal to $b_i$, i.e., the last column should be equal to the first column. The results show a clear match between theoretical and numerical results, thus validating our numerical solver.

TABLE I
MEAN TIME SPENT IN THE SYSTEM: VALIDATION OF THE NUMERICAL SOLVER OF THE $M/M/1$ CLOSED SYSTEM MODEL (FOR THE NUMERICAL RESULTS, 98% CONFIDENCE INTERVALS ARE SHOWN).

| $b_i$ | $T_i^{\text{theor}}$ | $T_i^{\text{numeric}}$ | $T_1^{\text{numeric}}/T_i^{\text{numeric}}$ |
|---|---|---|---|
| 1 | 2162.16 | $2157.99 \pm 6.30$ | – |
| 2 | 1081.08 | $1080.26 \pm 3.02$ | $2.00 \pm 3 \cdot 10^{-5}$ |
| 4 | 540.54 | $540.86 \pm 1.58$ | $3.99 \pm 0.01$ |
| 10 | 216.21 | $216.93 \pm 0.64$ | $9.96 \pm 0.05$ |

We now consider the case where the distribution of the service time is heavy tailed ($s = 0.5$), the system serves $Q = 6$ requests in parallel, and there is churn: we assume that the requests are active for a random interval, which is Weibull distributed, with scale parameter equal to 600 and shape parameter equal to 0.7 (heavy tailed distribution). Note that, with this level of churn, approximately 33% of the requests leave the system while waiting to be served.

The main difference with respect to the basic model is that it is not possible to compute the absolute values of the $T_i$s. On the other hand, the proportional property, i.e., $\frac{T_i}{T_j} = \left(\frac{b_j}{b_i}\right)^{1/r}$, is still valid, as shown in Table II

TABLE II
MEAN TIME SPENT IN THE SYSTEM IN CASE OF CHURN, MULTIPLE PARALLEL UPLOADS AND SERVICE TIME WEIBULL DISTRIBUTED (98% CONFIDENCE INTERVALS ARE SHOWN).

| $b_i$ | $T_1^{\text{numeric}}/T_i^{\text{numeric}}$ |
|---|---|
| 1 | – |
| 2 | $1.99 \pm 0.001$ |
| 4 | $3.97 \pm 0.002$ |
| 10 | $9.86 \pm 0.006$ |

Table II shows the case where all the three modifications – service time Weibull distributed, $Q$ requests served in parallel, and churn – are applied. We have tested the impact of each modification alone: none of them has an impact greater than the others, therefore all of them agree in the slight deviation with respect to the proportional property.

In summary, the basic $M/M/1$ closed system model represents a good approximation even for more complex systems, with different service time distributions, number of requests served in parallel and levels of churn.

## V. MEASUREMENTS

In this section, we provide the results of an extensive measurement campaign on the eMule system in order to further validate our theoretical results.

The eMule system differs from the model in many details. For instance the credits depend on the amount of data downloaded and uploaded, therefore they change over time: this behavior can not be modeled with simple tools, therefore we can only observe its impact on the main performance metric.

### A. Heavy Traffic Regime

The first step before starting to study the eMule performance is to check if the system is indeed working in the heavy traffic regime. To this aim, we have performed a specific measurement campaign.

The eMule clients are designed to report the position in the queue of the download requests. If peer $i$ sends a download requests to peer $j$, peer $i$ is able to visualize the status of such a request; we have modified an aMule client such that this status is written in the logs. The possible values that the status can assume are the following: On Queue (OQ), Queue Full (QF) and Too Many Connections (TMC). The status OQ says that the download request of peer $i$ has been placed in the waiting queue by peer $j$; in this case, it is possible to know the position in the queue. The status QF says that the download request of peer $i$ has been discarded by peer $j$ because there are no positions available. The status TMC indicates that peer $j$ is receiving too many connection requests, therefore the connection request is denied (and consequently the download request can not be sent).

For the measurement campaign we have taken 64 popular contents and sent the request for downloading the content. This translated into approximately 5400 individual requests, i.e. the client tried to connect and to send the download request to 5400 different peers. The results of these requests have been recorded in the log file.

The results of the measurement campaign shows that 87.4% of the connection requests are denied (status: TMC), and 3.2% of the download requests are denied (status: QF). Therefore, more than 90% of the contacted peers are saturated, i.e., they can not accept either a new incoming connection or a new incoming download request. Since in most of the cases connection slots are not available, we can interpret the buffer size $k$ in the model and the maximum number of connections.

### B. Measurement Setup

For the evaluation of the performance of eMule, we take the perspective of a single node that serves the requests for non copyrighted content issued by other peers. As such, we have instrumented an aMule client (version 2.2.6, [14]) to log different information. Among them, we consider all the events related to aMule's incentive system: in particular, we record when a node issues a request (*i.e.*, the request enters the waiting queue) when the request is served (*i.e.* it leaves the waiting queue and takes a serving slot), when the request has been completely served, or when it is sent back in the waiting queue (e.g., as a result of preemption). Additionally, our instrumented client reports all the incentive-related numerical values, such as bytes uploaded to other peers, bytes downloaded from other peers, and computed credits.

The log traces we obtain require post-processing, since they contain data that may affect the analysis. For instance, when we compute the total time spent in the system, we consider peers that have left after downloading the content, *i.e.*, we filter partial sessions due to churn.

Another issue is related to the time-varying nature of eMule credits: since the credits depend on the amount of data uploaded and downloaded, the credits of a generic peer may change over time. To simplify our analysis, we have divided the possible credits in ten classes: class $i$ contains the peers with credits greater or equal to $i - 0.5$, but smaller than $i + 0.5$. The only exceptions are class 1, which contains peers with credits between 1 and 1.5, and class 10, which contains peers with credits between 9.5 and 10, since eMule imposes a minimum and a maximum value for $C_j(t)$ (equal to 1 and 10 respectively). We have verified that most of the peers remain in the same class during our experiments, and we have filtered out the (very) few exceptions.

In our measurement campaign, we have tested our instrumented client in different periods of time. In the following, we will show some representative results in which we tested two values of available bandwidth for serving requests (240 and 360 kbit/s), and three values of uploading (serving) slots $Q$: 10, 4, and 1 slot.

### C. Measurement Results

Table III shows the results (time spent in the system, expressed in minutes) with bandwidth 240 kbit/s and different offered uploading slots. For each class (first column, which provides the values of the coefficients $b_i$s) we show the number of samples that contributed to provide the mean download time, along with the mean download time itself and the 95% confidence interval. Note that, differently from the numerical solution presented in Sec. IV, where we have performed multiple short runs, here we can only analyze a single, long observation (the experiment covers approximately 12 days).

Since we have a single observation, we can not compute the ratio with the corresponding confidence interval as we have done with the numerical solution in Sec. IV. We therefore consider an alternative approach: we use the download time of the lowest priority class (for which we also have the highest number of samples) as a reference $T_j$, and we compute the download times of the other classes, $T_i$ applying the proportional property of Theorem 2, *i.e.*, $T_i = \frac{b_j}{b_i} T_j$. The fourth column shows the results of this computation, while the last column shows the absolute relative error (difference) between the measured and the expected means.

TABLE III
MEASUREMENT RESULTS WITH BANDWIDTH 240 KBIT/S AND DIFFERENT SERVING SLOTS.

| Class | # samples | Mean (minute) with 95% Conf. Int. | Expected Mean | Error |
|---|---|---|---|---|
| Uploading slots Q=10 | | | | |
| 1 | 3758 | 511.29 ± 4.5 | - | - |
| 2 | 36 | 269 ± 17.9 | 255.65 | 13.35 |
| 3 | 59 | 182.19 ± 12.3 | 170.43 | 11.76 |
| 4 | 13 | 148.75 ± 13.5 | 127.82 | 20.93 |
| 10 | 416 | 49.5 ± 1.1 | 51.13 | 1.63 |
| Uploading slots Q=1 | | | | |
| 1 | 4289 | 357.6 ± 4.33 | - | - |
| 2 | 61 | 158.8 ± 19.18 | 187.8 | 29 |
| 3 | 13 | 145 ± 22.74 | 125.5 | 19.5 |
| 4 | 127 | 73 ± 4.71 | 93.9 | 20.9 |
| 10 | 287 | 35.2 ± 1.71 | 37.6 | 2.4 |

The confidence interval of each class overlaps or is close to the mean theoretical value. Considering the approximations made to compute the performance indexes during the measurements, we can observe a good match between measurements and theoretical values. In some cases, the low number of samples (second column) translates into a higher confidence interval and higher absolute error.

Note that the two measurement campaigns (corresponding to different values of $Q$) have been performed under different conditions, i.e., in different times and with a different set of shared files, therefore the results are not directly comparable. However, it is important to note that, independently from the value of $Q$, the proportional property holds, therefore the model is able to predict accurately the performance of the eMule system.

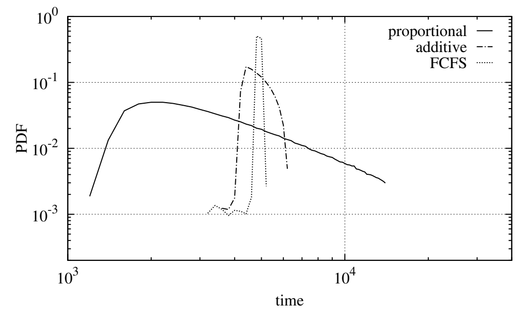


Fig. 2. PDF of the waiting times for different priority schemes.

## VI. An Alternative Incentive System

In this section, we illustrate a possible problem with the credit system currently implemented in eMule:the distribution of the request waiting times with a scheme akin to proportional differentiation exhibits *heavy tails*. As such, we propose, analyze and validate a possible alternative time-dependent priority discipline that can be obtained from Eq. 2 by setting the coefficients $b_i = b$, $\forall i \in P$ and $r = 1$: we call this approach *additive scheme*.

### A. Distribution of the Waiting Times: Results From a Numerical Analysis

In this section we are interested in understanding some basic properties of the complete probability distributions of the request waiting times. Since it is hard to derive such distributions analytically, we take a numerical approach which is similar to that developed in [23]. In particular, we have used our numerical solver based on Stochastic Simulations (cf. Sec. IV) to obtain the results. We assume a finite buffer of size $k = 5000$.

We compare the distribution obtained by three service disciplines: (i) the basic First Come First Serve (FCFS) discipline, (ii) the time-dependent proportional scheme (Sec. III-B) and (iii) the time-dependent additive scheme.

Specifically, for the proportional scheme we generate a large set of requests whose priority class is triangularly distributed in the interval $b_i \in \{1, 10\}$, with $r = 1$ and $a_i = 0 \, \forall i \in P$. We choose the triangular distribution, with its peak in the lowest value of $b_i$, since the measurement campaign has shown that most of the requests come from peers with low credits. Similarly, we evaluate the additive scheme for a set of requests whose priority class is identified by coefficients chosen at random (following a triangular distribution with peak in the lowest value of $a_i$) in the set $a_i \in \{1, 2500\}$, with $r = 1$ and $b_i = 1 \, \forall i \in P$. The results of our experiments consist in the empirical probability density function (PDF) of the request waiting times in the system, and are depicted in Figure 2.

Our results indicate that, for the FCFS scheme, the PDF of the waiting times exhibits a peak around the mean waiting time, as expected. Figure 2 illustrates that, for the proportional case, the PDF exhibits *heavy tails*, a result also observed in [24]. We performed another experiment with $b_i \in \{1, 50\}$ to study the sensitivity of the proportional scheme to the range in which the coefficient $b_i$ can take value: also in this case, the results (that we do not report here for the sake of clarity) show a PDF with heavy tails.

In [24], the authors consider also the additive scheme: they show that the additive scheme exhibits heavy tails if the coefficients $a_i$ are selected from a probability distribution that has heavy tails. This means that, if the coefficients are bounded, i.e., $a_i < a_{max}, \forall i \in P$, the waiting time distribution does not have heavy tails, as our numerical results confirm (see Fig. 2). We note that the PDF is relatively tight and centered around the mean waiting time of the FCFS scheme, and has a support that is correlated to the difference between the maximum and minimum values of the coefficients $a_i$.

In summary, a system based on the proportional scheme exhibits heavy tails in the distribution of the waiting times. Instead, the additive scheme, independently from the coefficients $a_i$, does not exhibit heavy tails.

### B. Main Results for the Additive Scheme

For the additive scheme, we are able to find general results which are valid for both the non-preemptive and the preemptive cases. We assume, as in Sec. III-A, a $M/M/1/k{+}1$ closed system.

*Theorem 3:* Given any two priority groups $i$ and $j$, the mean waiting times $W_i$ and $W_j$, for both the non preemptive and the preemptive cases, in closed systems, satisfies the following condition:

$$(W_i - W_j) \to \frac{a_j - a_i}{b} \tag{7}$$

The mean waiting times can be computed as:

$$W_i = -\frac{a_i}{b} + \frac{k}{\mu} + \frac{1}{b}\sum_{i=1}^{P} p_i a_i. \tag{8}$$

*Proof:* See Appendix C. ∎

As for the proportional case, Theorem 3 provides a relation between the mean waiting times independently from the traffic composition (i.e., the values of $\rho_i$). Moreover, having a simple expression for the absolute values of $W_i$, allows us to easily evaluate the impact of the system parameters on the waiting times. To the best of our knowledge, this result, or part of it, has never been found before, not even in the infinite buffer case.

### C. Results

As previously done for the proportional case, we first evaluate the accuracy of the simple model in predicting the perfomance of an enhanced model where we introduce the three modifications explained in Sec. IV: the service time is Weibull distributed, $Q$ requests are served in parallel, with churn. The parameters of the service time distribution and of the request online time remain the same used in Sec. IV. The coefficient $b$ is set to 1, while the coefficients $a_i$ are set to 14, 28, 56 and 140 – note that such coefficient are the values 1, 2, 4 and 10 all multiplied by 14, the reason of which will become clear below. The number of requests served in parallel is $Q = 6$.

TABLE IV
MEAN TIME SPENT IN THE SYSTEM WITH THE ADDITIVE (98% CONFIDENCE INTERVALS ARE SHOWN).

| $a_i$ | $a_i - a_1$ | $W_1^{\text{numeric}} - W_i^{\text{numeric}}$ |
|---|---|---|
| 14 | – | – |
| 28 | 14 | $15.11 \pm 0.56$ |
| 56 | 42 | $44.75 \pm 0.51$ |
| 140 | 126 | $130.79 \pm 0.51$ |

As previously noted, it is not possible to compute the theoretical absolute values, therefore we will consider the main property of Theorem 3, i.e. $(W_i - W_j) \to (a_j - a_i)/b$. Table IV shows the results obtained for the additive scheme. The second column shows the difference between the coefficients $a_i$s and the last column shows the difference between the mean waiting times, showing a good match. We obtained similar results with different settings (service time distribution, distributions of the churn, different values of $Q$). This means that, even for general distributions, the single server queue model represents a good approximation of the system.

Once tested that the model maintains the properties for the general case, we have performed a new measurement campaign with a modified aMule client. In particular, we have implemented the additive scheme by modifying the computation of the instantaneous priority, i.e., Eq. 1: the instantaneous priority is set to:

$$q_j(t) = \big(t - T_{\text{arrival}}\big) + f_p \cdot C_j(t) \cdot \alpha \tag{9}$$

We have not modified the values of the coefficients $C_j(t)$, but we have introduced a parameter $\alpha$ to differentiate better the classes. In particular, we set $\alpha = 20$. The value of $f_p \cdot C_j(t) \cdot \alpha$ corresponds to the coefficient $a_i$ in Eq. 2. Since the default value of $f_p$ is 0.7, and the minimum and the maximum values of $C_j(t)$ are 1 and 10 respectively, then the minimum and the maximum values of $a_i$ are 14 and 140. In general, class $i$ will have coefficient $a_i = i \cdot 14$, $i = 1, \ldots, 10$.

The measurement setup is similar to the one used for the proportional case (cf. Sec. V). In particular, we have a server bandwidth equal to 360 kbit/s. We have tested only $Q = 1$ since, as already noted, the value of $Q$ does not affect the proportional or differential properties. Table V shows the measurement results. For each class, we show the value of the coefficient $a_i$, the mean waiting time (in minutes) with the 95% confidence interval, and the expected mean computed according to the main property of Theorem 3. In particular, we have used class 1 as reference, and we have computed the mean waiting time as $W_i = W_1 + a_1 - a_i$.

Also in this case, we can observe a good match between measurements and theoretical values, especially considering the difference between class 1 and 10. The small number of samples that we had in these experiments explains the wider confidence interval values when compared to the results of previous experiments (Table III).

In a generic uncontrolled environment, the additive scheme is then able to provide service differentiation that depends solely on the parameters of the incentive scheme.

TABLE V
MEASUREMENT RESULTS WITH BANDWIDTH 360 KBIT/S AND DIFFERENT BUFFER SIZES.

| Class | $a_i$ | # samples | Mean (minute) with 95% Conf. Int. | Expected Mean | Error |
|---|---|---|---|---|---|
| 1 | 14 | 3431 | 365.7 ± 5.3 | - | - |
| 2 | 28 | 12 | 286.8 ± 23.57 | 351.7 | 64.9 |
| 4 | 56 | 11 | 291.3 ± 22.7 | 323.7 | 32.4 |
| 10 | 140 | 63 | 245.4 ± 14.91 | 239.7 | 5.7 |

### D. Discussion

The proposed additive scheme has some benefits – it avoids the heavy tails in the mean time spent in the system – but it has also some drawbacks. In a peer-to-peer system it is difficult to distinguish between a free-rider or a newly arrived user without the help of additional components (e.g., a reputation system for identifying free-riders). A scheme that tries to improve the performance of a newly arrived peer will automatically help free riders.

The question then becomes if the gain that may come from newly arrived users (if they can be involved in the exchange faster than the actual scheme) is greater than the cost of providing resources to free-riders. This remains an open question that is out of the scope of this paper.

From our experience, we believe that the additive scheme is a actually a better option: in [25] the authors show that in eMule many users are willing to contribute to the system, therefore providing a better service to newly arrived users helps in spreading the content and alleviate the burden on other users.

Despite the application of the additive scheme to eMule, the main theoretical results provided in Sect. VI-B can be applied to other context, i.e., system with time-dependent scheduling disciplines under heavy traffic, where the problem of free-riders may not be present – e.g., job scheduling in operating systems.

## VII. Related Work

Incentives in P2P systems have been the subject of many studies in the past few years – see [26], [27], [28] and the references therein. Such studies are focused on BitTorrent. There are some notable exceptions [29], [30], [31] that focused on eMule. In [29] the authors analyze the effect of incentives on the download time; nevertheless they use a simplified model for the credits assigned to peers, and they do not focus on the relation of the waiting times of the requests as we do in our work. The aim of the work in [30] is to improve the fairness in terms of downloaded and uploaded content, while we consider the waiting time of the requests and the proportional property given by the scheduling discipline. The authors in [32] study the average download time of the peers, without distinction among different classes (with respect to the credits) of peers, while our work is focused on such a distinction.

With respect to the model, single server queues with time-dependent priority disciplines have been studied originally in [19] for the linear case (i.e., $r = 1$), and in [20], [21], [33] for more general cases ($r > 0$). None of such works considers a finite buffer and closed systems, as we do in this work. Only [22] considered the heavy traffic regime for the linear time-dependent priority scheme ($r = 1$) and infinite buffer, so our results for the proportional scheme represent a generalization of the results in [22].

The heavy traffic regime for the linear time-dependent priority (i.e., $r = 1$), and some of the properties related to the proportional scheme, has been also studied within the *Proportional Delay Differentiation* (PDD) framework [34], [35]. As previously pointed out, we consider the general case with any value of $r > 0$ and finite buffer.

Also the authors in [34] study the properties of the additive scheme under heavy traffic, in the specific case with $b_i = b = 1$; however, they do so using a simulation-based approach. Instead, we consider the general additive scheme with $b_i = b$ and we provide analytical results of its properties for closed systems.

Finally, all the above works consider systems and applications with *no preemption*, *i.e.*, a single server queue in which, once a request has been scheduled, it will be served before any other request will be considered for scheduling. In contrast, we provide results also for the preemptive work conserving case.

## VIII. Conclusion

In this work we considered the incentive scheme adopted by eMule / aMule, and studied its impact on the application by modeling it as a time-dependent priority discipline. We showed that service differentiation – that is, peers are granted upload slots as a function of their contribution – is achieved with a sophisticated combination of a "tit-for-tat"-like discipline, that materializes in credits assigned to peers, and a time-dependent priority scheme, where priority is assigned to peers based on their credits. Essentially, the incentive mechanism of eMule / aMule takes into account both the level of contribution of a peer and the time it has spent waiting to be served.

Our analysis showed that it is possible to derive simple laws that govern the service differentiation achieved by a range of priority mechanisms, including that of eMule. In practice, the relative performance of peers can be determined by configuring a handful set of parameters. We validated our model and an extension thereof (which accounts for general service rate and churn) both numerically and with a measurement campaign on the live eMule / aMule network.

Moreover, we identified an area in which the current eMule incentive scheme may be improved: instead of using a proportional service differentiation, in which some peers suffer from starvation, we proposed an additive scheme that mitigates this problem. We analyzed and validated our scheme through numerical simulations and an additional measurement campaign, and showed that our approach maintains the property of the proportional scheme in that a handful set of parameters is sufficient to regulate service differentiation.

In conclusion, we remark that our model could be applied to other applications – e.g., OCH services, scheduling systems – that necessitate service differentiation, with or without the component that accounts for the level of contributions of the entities involved. However, this requires an in-depth analysis that we leave as future work. Other future works include the evaluation of the system with multiple servers, in order to examine the performance of the total download time using multiple sources.

## Appendix A
## Proof of Theorem 1

We consider a generic request coming from group $p$, and its mean waiting time, $W_p$. We start from the $P$ simultaneous equations used to derive the time spent in the system defined in [19]. Let $N_i$ be the mean number of requests of group $i$ in the queue, and let $f_{ip}$ be the expected fraction of such requests which receive service before the newly arrived request from group $p$.

Let $M_i$ be the mean number of requests of group $i$ which arrive during $W_p$, and let $g_{ip}$ be the expected fraction of such requests which receive service before the generic request of group $p$ we are considering.

Given these definitions, for a generic class $p$ we have:

$$W_p = W_0 + \sum_{i=1}^{P} \frac{N_i f_{ip}}{\mu_i} + \sum_{i=1}^{P} \frac{M_i g_{ip}}{\mu_i}. \tag{10}$$

We need to compute the different parameters. In case of $N_i$, we can use Little's theorem, obtaining $N_i = \lambda_i T_i$. In case of $M_i$, when observing the system for $W_p$ seconds, we see $M_i = \lambda_i W_p$ arrivals. In both cases, $N_i$ and $M_i$, the mean arrival rate $\lambda_i$ is equal to $p_i \mu$ (recall that the group of the new arrival is independent from the group of the request that has completed the service).

For the parameters $f_{ip}$ and $g_{ip}$, we note that the derivation obtained in [19] and [33] are based only on the Little theorem, which is still valid in our case with a closed system. Therefore, we can use those results and arrive at the following expressions:

$$f_{ip} = \begin{cases} (b_i/b_p)^{1/r} & i < p \\ 1 & i \geq p \end{cases}$$

$$g_{ip} = \begin{cases} 0 & i \leq p \\ 1 - (b_p/b_i)^{1/r} & i > p \end{cases}$$

Combining all the information, we obtain

$$W_p = \frac{W_0 + \sum_{i=1}^{p-1} \rho_i W_i \left(\frac{b_i}{b_p}\right)^{1/r} + \sum_{i=p}^{P} \rho_i W_i}{1 - \sum_{i=p+1}^{P} \rho_i \left(1 - \left(\frac{b_p}{b_i}\right)^{1/r}\right)}. \tag{11}$$

At this point, [19] invokes the Kleinrock's conservation law to simplify the expression. Since we are considering a closed system, we analyze Eq. 11 without using the Kleinrock's conservation law. For the lowest priority group ($p = 1$), noting that $\sum_{i=2}^{P} \rho_i = \rho - \rho_1$, in case of a closed system ($\rho = 1$), from Eq. 11 we obtain

$$W_0 + \sum_{i=1}^{P} \rho_i W_i = b_1^{1/r} W_1 \sum_{i=1}^{P} \frac{\rho_i}{b_i^{1/r}}. \tag{12}$$

For the group with $p = 2$, Eq. 11 becomes, after some manipulation,

$$W_2 = \frac{W_0 + \sum_{i=1}^{P} \rho_i W_i - \rho_1 W_1 \left(1 - \left(\frac{b_1}{b_2}\right)^{1/r}\right)}{1 - \sum_{i=3}^{P} \rho_i + b_2^{1/r} \sum_{i=3}^{P} \frac{\rho_i}{b_i^{1/r}}}. \tag{13}$$

The numerator of the fraction, with the help of Eq. 12, can be transformed into

$$b_1^{1/r} W_1 \sum_{i=1}^{P} \frac{\rho_i}{b_i^{1/r}} - \rho_1 W_1 + \rho_1 W_1 \left(\frac{b_1}{b_2}\right)^{1/r} =$$
$$b_1^{1/r} W_1 \left(\frac{\rho_1}{b_2^{1/r}} + \sum_{i=2}^{P} \frac{\rho_i}{b_i^{1/r}}\right).$$

The denominator of the fraction can be transformed into

$$\rho_1 + \rho_2 + b_2^{1/r} \sum_{i=3}^{P} \frac{\rho_i}{b_i^{1/r}} = b_2^{1/r} \left(\frac{\rho_1}{b_2^{1/r}} + \sum_{i=2}^{P} \frac{\rho_i}{b_i^{1/r}}\right).$$

Equation 13 then becomes

$$b_2^{1/r} W_2 = b_1^{1/r} W_1. \tag{14}$$

With the help of Eqs. 14 and 12 we can compute $W_3$; repeating this process for all groups we obtain the result of the first part of the Theorem.

The absolute values of the $W_i$s can be found considering that the number of requests in the queue is constant ($k$) and equal to the sum of requests belonging to each group, which can be derived from $W_i$ using Little's theorem.

$$\sum_{i=1}^{P} \lambda_i W_i = k \tag{15}$$

Since $W_i = \left(\frac{b_1}{b_i}\right)^{1/r} W_1$, we can derive the expression for $W_1$ and, consequently, for all $W_i$s.

## Appendix B
## Proof of Theorem 2

In case of service with preemption, we consider the mean time spent in the system by a generic request coming from group $p$, $T_p$. With similar arguments used in Appendix A we have the following relation:

$$T_p = \frac{1}{\mu_p} + \sum_{i=1}^{P} \frac{N_i f_{ip}}{\mu_i} + \sum_{i=1}^{P} \frac{M_i g_{ip}}{\mu_i}. \tag{16}$$

It is easy to show that the values of $N_i$, $M_i$, $f_{ip}$ and $g_{ip}$ remain the same as in Appendix A. We obtain:

$$T_p = \frac{\frac{1}{\mu} + \sum_{i=1}^{p-1} \rho_i T_i \left(\frac{b_i}{b_p}\right)^{1/r} + \sum_{i=p}^{P} \rho_i T_i}{1 - \sum_{i=p+1}^{P} \rho_i \left(1 - \left(\frac{b_p}{b_i}\right)^{1/r}\right)}. \tag{17}$$

Comparing Eqs. 17 and 11 we notice that they have the same structure, with $1/\mu$ instead of $W_0$, and $T_i$ instead of $W_i$. Thus the proof follows exactly the same scheme used in Appendix A.

## Appendix C
## Proof of Theorem 3

We consider first the non-preemptive case: the starting point remains Eq. 10, and the value of $N_i$, $M_i$ are the same, while $f_{ip}$ and $g_{ip}$ change.

Let's assume that the newly arrived request (which we call the tagged request) belongs to group $p$. As said before, $f_{ip}$ represents the expected fraction of group $i$ requests (already in the queue at the arrival of the tagged request) which receive service before the tagged request. Clearly, if $i \geq p$, then $f_{ip} = 1$. If $i < p$, the request arrived at time $Y_i$ seconds before the tagged one, with $W_i > Y_i$ such that

$$bY_i + a_i = a_p,$$

will receive service before the tagged request. So, the group $i$ request should arrive at most $Y_i = (a_p - a_i)/b$ seconds before the tagged one. Let $P[w_i > t]$ be the probability that the waiting time $w_i$ (whose mean is $W_i$) is greater than $t$, we obtain

$$f_{ip} = \begin{cases} \int_{(a_p - a_i)/b}^{\infty} \lambda_i P[w_i > t] \, dt & i < p \\ 1 & i \geq p \end{cases}$$

The parameter $g_{ip}$ represents the expected fraction of group $i$ requests which arrive during $W_p$ and receive service before the tagged request. If $i \leq p$, then $g_{ip} = 0$. If $i > p$, the request will receive service if it arrives before $w_p$, and $V_i$ seconds after the tagged request, with:

$$bV_i + a_p = a_i.$$

Therefore, $g_{ip} = \lambda_i \min\big((a_i - a_p)/b, w_p\big)$. Following the same approach used in [21] it is possible to show that

$$\min\big((a_i - a_p)/b, w_p\big) = \int_0^{(a_i - a_p)/b} P[w_p > t] \, dt.$$

We then obtain

$$g_{ip} = \begin{cases} 0 & i \leq p \\ \lambda_i \int_0^{(a_i - a_p)/b} P[w_p > t] \, dt & i > p \end{cases}$$

Combining all the information, we obtain

$$W_p = W_0 + \sum_{i=1}^{p-1} \rho_i \int_{(a_p - a_i)/b}^{\infty} P[w_i > t] \, dt + \sum_{i=p}^{P} \rho_i W_i + \sum_{i=p+1}^{P} \rho_i \int_0^{(a_i - a_p)/b} P[w_p > t] \, dt. \quad (18)$$

Note that

$$\int_x^{\infty} P[w_i > t] \, dt = W_i - \int_0^{x} P[w_i > t] \, dt.$$

In case of heavy traffic, as done in [21], we can assume that $w_i > (a_j - a_i)/b$, for any $j$, and approximate the integrals by $\int_0^x P[w_i > t] \, dt \approx x$. Equation 18 becomes

$$W_p = W_0 + \sum_{i=1}^{P} \rho_i W_i - \sum_{i=1}^{p-1} \rho_i \frac{a_p - a_i}{b} + \sum_{i=p+1}^{P} \rho_i \frac{a_i - a_p}{b}. \quad (19)$$

Since $\rho = 1$, we obtain

$$W_p + \frac{a_p}{b} = W_0 + \sum_{i=1}^{P} \rho_i W_i - \sum_{i=1}^{P} \rho_i \frac{a_i}{b} = \text{constant}. \quad (20)$$

In case of service with preemption, we consider the mean time spent in the system by a generic request coming from group $p$, $T_p$. With similar arguments used for the non preemptive case, we arrive at the following relation:

$$T_p = \frac{1}{\mu_p} + \sum_{i=1}^{P} \rho_i T_i - \sum_{i=1}^{p-1} \rho_i \frac{a_p - a_i}{b} + \sum_{i=p+1}^{P} \rho_i \frac{a_i - a_p}{b}, \quad (21)$$

which leads to:

$$T_p - \frac{1}{\mu_p} + \frac{a_p}{b} = \sum_{i=1}^{P} \rho_i T_i - \sum_{i=1}^{P} \rho_i \frac{a_i}{b} = \text{constant}. \quad (22)$$

Recalling that $T_p - \frac{1}{\mu_p} = W_p$, we have completed the proof (the absolute values of $W_i$s can be obtained following through with almost identical arguments used in Appendix A).

## References


[1] T. Lauinger, E. Kirda, and P. Michiardi, "Paying for piracy? an analysis of one-click hosters controversial reward schemes," in *International Symposium on Research in Attacks, Intrusions and Defenses*, 2012.

[2] Sandvine, Global Internet Phenomena Report, Spring 2011 . http://www.sandvine.com/downloads/documents/05-17-2011_phenomena/Sandvine%20Global%20Internet%20Phenomena%20Report.pdf/.

[3] J. S. Otto, M. A. Sanchez, D. R. Choffnes, F. E. Bustamante, and G. Siganos, "On blind mice and the elephant," in *Proc. of ACM SIGCOMM*, 2011.

[4] G. Maier, A. Feldmann, V. Paxson, and M. Allman, "On dominant characteristics of residential broadband internet," in *Proc. of ACM IMC*, 2009.

[5] H. Schulze and K. Mochalski, "ipoque: Internet study 2008/2009," http://www.ipoque.com/media/internet_studies, 2009.

[6] D. Antoniades, E. P. Markatos, and C. Dovrolis, "One-click hosting services: A file-sharing hideout," in *Proc. of ACM IMC*, 2009.

[7] A. Legout, G. Urvoy-Keller, and P. Michiardi, "Rarest first and choke algorithms are enough," in *Proc. of ACM IMC*, 2006.

[8] D. Qiu and R. Srikant, "Modeling and performance analysis of bittorrent-like peer-to-peer networks," in *Proc. of ACM SIGCOMM*, 2004.

[9] L. Guo, S. Chen, Z. Xiao, E. Tan, X. Ding, and X. Zhang, "Measurements, analysis, and modeling of bittorrent-like systems," in *Proc. of ACM IMC*, 2005.

[10] J. A. Pouwelse, P. Garbacki, D. H. J. Epema, and H. J. Sips, "The bittorrent p2p file-sharing system: Measurements and analysis," in *Proc. of USENIX IPTPS*, 2005.

[11] R. Cuevas, N. Laoutaris, X. Yang, G. Siganos, and P. Rodriguez, "Deep diving into bittorrent locality," in *Proc. of IEEE INFOCOM*, 2011.

[12] B. Cohen, "Incentives build robustness in BitTorrent," in *First Workshop on Economics of Peer-to-Peer Systems*, 2003.

[13] "The emule project," http://www.emule-project.net/.

[14] A-Mule, http://www.amule.org/.

[15] Q. Wu and X. Chen, "Advanced distributed crawling system for kad network," *Jour. of Computational Information Systems*, vol. 17, pp. 677–684, 2011.

[16] J. Yu and Z. Li, "Active measurement of routing table in kad," in *Proc. of IEEE CCNC*, 2009.

[17] Y. Kulbak and D. Bickson, "The emule protocol specification," Leibniz Center TR-2005-03, School of Computer Science and Engineering, The Hebrew University, Tech. Rep., 2005.

[18] A. Cobham, "Priority assignment in waiting line problems," *Operations Research*, vol. 2, pp. 70–76, 1954.

[19] L. Kleinrock, "A delay dependent queue discipline," *Naval Research Logistics Quarterly*, vol. 11, pp. 329–341, 1964.

[20] A. Netterman and I. Adiri, "A dynamic priority queue with general concave priority functions," *Opserations Research*, vol. 27, pp. 1088–1100, 1979.

[21] U. Bagchi and R. Sullivan, "Dynamic, non-preemptive priority queues with general, linearly increasing priority function," *Operations Research*, vol. 33, pp. 1278–1298, 1985.

[22] R. Nelson, "Heavy traffic response times for a priority queue with linear priorities," *Operations Research*, vol. 38, pp. 560–563, 1990.

[23] A.-L. Barabási, "The origin of bursts and heavy tails in humans dynamics," *Nature*, vol. 435, pp. 207–211, 2005.

[24] P. Blanchard and M. Hongler, "Modeling human activity in the spirit of barabasi's queueing systems," *Phys. Review E*, vol. 75, p. 026102, 2007.

[25] P. Brown and S. Petrovic, "A new statistical approach to estimate global file populations from local observations in the eDonkey P2P file sharing system," *Ann. Telecommun.*, vol. 66, pp. 5–16, 2011.

[26] G. Neglia, G. L. Presti, H. Zhang, and D. Towsley, "A network formation game approach to study bittorrent tit-for-tat," in *Proc. of ACM NETCOOP*, 2007.

[27] D. Levin, K. LaCurts, N. Spring, and B. Bhattacharjee, "Bittorrent is an auction: Analyzing and improving bittorrents incentives," in *Proc. of ACM SIGCOMM*, 2008.

[28] M. Feldman and J. Chuang, "Overcoming free-riding behavior in peer-to-peer systems," *ACM SIGecom*, vol. 5, 2005.

[29] D. G. Yunzhao Li and C. Scoglio, "Reward only is not enough: Evaluating and improving the fairness policy of the p2p file sharing network emule/edonkey," *PEER-TO-PEER NETWORKING AND APPLICATIONS, Volume 5, Number 1*, pp. 40–57, 2012.

[30] F. D. Luca Caviglione, Cristiano Cervellera and F. A. Grassia, "Optimization of an emule-like modifier strategy," *Computer Communications, 31(16), Performance Evaluation of Communication Networks (SPECTS 2007)*, pp. 3876–3882, 2008.

[31] P. B. Sanja Petrovic and J.-L. Costeux, "Unfairness in the e-mule file sharing system," *Lecture Notes in Computer Science, vol. 4516*, pp. 594–605, 2007.

[32] S. Petrovic and P. Brown, "Large scale analysis of the edonkey p2p file sharing system," in *Proc. of IEEE INFOCOM*, 2009.

[33] L. Kleinrock and R. Finkelstein, "Time dependent priority queues," *Operations Research*, vol. 15, pp. 104–116, 1967.

[34] C.Dovrolis, D. Stiliadis, and P. Ramanathan, "Proportional differentiated services: Delay differentiation and packet scheduling," *IEEE/ACM Transactions on Networking*, vol. 10, pp. 12–26, 2002.

[35] M. Leung, J. Lui, and D. Yau, "Adaptive proportional delay differentiated services: Characterization and performance evaluation," *IEEE/ACM Transactions on Networking*, vol. 9, pp. 801–817, 2001.


PLACE PHOTO HERE

**Damiano Carra** received his Laurea in Telecommunication Engineering from Politecnico di Milano, and his Ph.D. in Computer Science from University of Trento. He is currently an Assistant Professor in the Computer Science Department at University of Verona. His research interests include modeling and performance evaluation of peer-to-peer networks and distributed systems.

PLACE PHOTO HERE

**Pietro Michiardi** received his M.S. in Computer Science from EURECOM and his M.S. in Electrical Engineering from Politecnico di Torino. Pietro received his Ph.D. in Computer Science from Telecom ParisTech (former ENST, Paris). Today, Pietro is an Assistant Professor of Computer Science at EURECOM. Pietro currently leads the Distributed System Group, which blends theory and system research focusing on large-scale distributed systems (including data processing and data storage), and scalable algorithm design to mine massive amounts of data. Additional research interests are on system, algorithmic, and performance evaluation aspects of computer networks and distributed systems.

PLACE PHOTO HERE

**Hani Salah** is a PhD student in the P2P Networks Group at the Computer Science Department at Technische Universität Darmstadt, Germany. His research interests include monitoring and measurements of distributed systems, graph theory, and security. From 2007 to 2010, he worked as a lecturer at Information Technology Department at Palestine Polytechnic University (PPU), Hebron, Palestine. He obtained his M.Sc. in Information Technology from the Royal Institute of Technology (KTH), Stockholm, Sweden, in 2007, and his B.Sc. degree in Computer Systems Engineering from PPU, in 2003.

PLACE PHOTO HERE

**Thorsten Strufe** is professor for peer-to-peer networks at Technische Universität Darmstadt. His research interests lie in the areas of large scale distributed systems and social networking services, with an emphasis on privacy and resilience. Recently, he has focused on studying user behavior and security in online social networks and possibilities to provide privacy-preserving and secure social networking services through P2P technologies. He was appointed visiting professor for Dependable Distributed Systems at University of Mannheim, Germany, throughout 2011. Previously, he took a post as senior researcher at EURECOM and at TU Ilmenau, working on analysis and the security of online social networks, as well as resilient networking. He received his PhD degree from TU Ilmenau in 2007.