Peer-assisted VoD Systems: an Efficient Modeling Framework

Delia Ciullo, Valentina Martina, Michele Garetto, Emilio Leonardi, Giovanni Luca Torrisi

Abstract—We analyze a peer-assisted Video-on-Demand system in which users contribute their upload bandwidth to the redistribution of a video that they are downloading or that they have cached locally. Our target is to characterize the additional bandwidth that servers must supply to immediately satisfy all requests to watch a given video. We develop an approximate fluid model to compute the required server bandwidth in the sequential delivery case, as well as in controlled non sequential swarms. Our approach is able to capture several stochastic effects related to peer churn, upload bandwidth heterogeneity, non-stationary traffic conditions, which have not been documented or analyzed before. At last, we provide important hints for the design of efficient peer-assisted VoD systems under server capacity constraints.

Index Terms—Video-on-demand, peer-to-peer, performance evaluation.

1 Introduction

The efficient distribution of video contents will be one of the main challenges of the future Internet. According to Cisco forecasts [2], the combination of all forms of video (live streaming, video-on-demand and P2P file sharing) will be in the range of 80 to 90 percent of global consumer Internet traffic by 2017, posing a tremendous challenge to both content providers and network operators. In particular, Video-On-Demand traffic has been increasing tremendously during the last few years. For example, it has been reported that, on a normal weeknight, Netflix alone accounts for almost one third of all Internet traffic entering North American homes.

Traditional (client-server) Content Delivery Networks (CDN) help alleviate the traffic on the transport infrastructure by "moving" contents close to the users, however they do not solve the scalability problem of data centers and server farms, whose resources (bandwidth/storage/processing) increase linearly with the user demand and the data volume.

Peer-assisted video distribution architectures, in which users contribute their upload bandwidth to the system while viewing the requested video, have been advocated as a viable alternative to traditional CDNs to reduce the server workload and guarantee the scalability to large populations of users [3], [4].

Several peer-assisted systems have already been deployed, such as PPLive, GridCast, PPStream, TVU, SopCast, Xunlei Kankan [5].

Despite the wide popularity gained by existing applications, several fundamental questions remain unanswered about the design of video streaming systems and the potential benefits of the peer-assisted approach. Indeed, the unpredictable nature of users cooperation, that cannot always guarantee the strict quality-of-service requirements of online video; the added complexity on the control plane due to signalling and chunk scheduling; and the need to provide incentive mechanisms to the users, tend to discourage the content providers to adopt peer-assisted solutions.

In our work, we focus on peer-assisted Video-on-Demand (VoD) systems, for the on-line distribution of movies to a large audience of users, such as Netflix. Users can browse a catalog of available movies, and asynchronously issue requests to watch a given content, which are ideally immediately satisfied by the system, with the optional support for limited VCR actions such as pause and jump backward/restart. Notice that Video-on-Demand (VoD) systems are quite different from live video streaming applications, in which users join the distribution of a given TV channel at random points in time, but peers connected to the same channel watch the content almost synchronously.

In peer-assisted VoD systems, users interested to a specific video can retrieve it from servers (CDN modality), from peers downloading/watching it, and from users storing a copy of it in their computer/Internet TV memory or in dedicated set-top-boxes remotely controllable by the network operator [6]. The content is typically divided into small chunks (typically one or few video frames) which are retrieved from other peers (or from servers) in a fully distributed (swarming) fashion based on the exchange of chunk bitmaps. However, differently for traditional file-sharing, chunk and peer selection strategies for peer-assisted video distribution must account for the fact that users watch while downloading. In particular, to avoid service interruptions/degradations: i) a minimum average download rate equal to the video playback rate must be guaranteed; ii) an "almost in order" delivery of chunks is required.

Our main contribution is a stochastic fluid framework that allows to approximately estimate the additional bandwidth that servers must provide to satisfy all requests to watch a given video. Our methodology can account for several stochastic effects related to peer churn, upload bandwidth heterogeneity, non-stationary traffic conditions, which have not been analyzed before, providing a useful tool for the analysis and design of

- *D. Ciullo is with EURECOM, Sophia Antipolis, France. E-mail: delia.ciullo@eurecom.fr*
- *V. Martina and E. Leonardi are with Dipartimento di Elettronica, Politecnico di Torino, Torino, Italy. E-mail: {valentina.martina, emilio.leonardi}@polito.it*
- *M. Garetto is with Dipartimento di Informatica, Università di Torino, Torino, Italy. E-mail: michele.garetto@unito.it*
- *G. L. Torrisi is with Istituto per le Applicazioni del Calcolo, CNR, Roma, Italy. E-mail: torrisi@iac.rm.cnr.it*

A preliminary version of this paper appeared at IEEE INFOCOM Mini-Conference 2012 [1].

VoD systems.

We consider both the simple basic sequential delivery scheme, in which users download the video chunks (almost) sequentially, as well as, schemes that tolerating higher play-out delay permit to exploit non-sequential delivery to improve the overall system performance.

The analytical approach described in this paper complements the analysis presented in [7], in which we obtain rigorous bounds for the sequential delivery scheme (under stationary traffic conditions) and asymptotic results as the number of users increases. With respect to [7], we extend the analysis to non-sequential delivery schemes and non-stationary traffic conditions, with a different goal in mind, *i.e.*, to provide a performance evaluation tool that can be readily used for system design and optimization.

We emphasize that in our work we do not consider issues related to optimal replication strategies of heterogeneous contents (in size and popularity) or optimal peer resource allocation (in terms of storage and upload bandwidth) in the presence of multiple videos (e.g., universal streaming architectures). This because we focus on the bandwidth requested from the servers to distribute a given video, assuming that the peer resources allocated to it (i.e., number of copies available in the system and the amount of upload bandwidth devoted to the considered video) are given. Our analysis can be combined with optimal resource and replication strategies for universal streaming architectures.

For an overview of related work, see Section 10 in [8].

2 MODEL

2.1 System assumptions

We model a fairly general peer-assisted VoD system. Users[1] run applications that allow them to browse an online catalog of videos. When a user selects a video, we assume that the request is immediately satisfied and the selected video can be watched uninterruptedly till the end (*i.e.*, a continuity index equal to 1 is guaranteed). This is possible only if the system is able to steadily provide to each user a data flow greater than or equal to the video playback rate.

Users contribute their upload bandwidth to the video distribution, thus they can retrieve part of the video (or even the entire video) from other peers, saving servers resources. The main goal of our analysis is to characterize the additional bandwidth that servers must supply (in addition We provide a fundamental tool to properly dimension the CDN infrastructure supporting the VoD system.

We focus on a given video of duration T_v seconds and size L bytes, which is played back by the users' applications at rate $d_v = L/T_v$ bytes/s. Clearly, to guarantee continuous playback each user must *at least* receive video chunks sequentially at rate d_v. As a widely adopted strategy to mitigate bandwidth fluctuations, applications pre-fetch and buffer video chunks before playback (notice that we consider VoD systems, hence we assume, in contrast to live streaming systems, that all chunks are immediately available, at least at the servers). In our model, we assume that the system provides to each user a fixed target download rate $d \geq d_v$ (we assume that the download bandwidth on the access links of the users is large enough that it does not constitute a bottleneck).

In general, the target download rate of a peer could be adapted to the portion of video being downloaded, or even depend on some peer's characteristics (such as its upload bandwidth). By imposing a constant target download rate $d \geq d_v$ at each user we simplify the analysis, while obtaining a conservative prediction with respect to the case in which the target download rate is adapted over time and to the peer characteristics. Notice that the target download rate d can be chosen by the system. We will show that in some cases, unexpectedly, the optimal value of d (*i.e.*, the one that minimizes the average bandwidth requested from the servers) is actually larger than d_v.

The amount of upload bandwidth with which peers contribute to the redistribution of the video that they are downloading may or may not be under the control of the system. In our analysis, we assume that the upload bandwidth available at a peer is a random variable with given distribution. This way, we encompass both the realistic case of users with heterogeneous Internet connections (*i.e.*, ADSL, fiber, LAN) and cross-traffic fluctuations, and the case in which the peer upload bandwidth allocated to the given video is tuned by the system (such as in universal streaming architectures). More specifically, the amount of upload bandwidth with which users contribute at a given time to the redistribution of the considered video is modeled by a random variable U with cumulative distribution function $F_U(w)$, mean $\overline{U}$ and variance σ_U^2. The random variables denoting the instantaneous upload bandwidths of the users are assumed to be i.i.d. (identically and independently distributed). See Section 11 in [8] for a critical discussion on our system assumptions.

2.2 Peers dynamics

We need to incorporate in our analysis a model describing how peers join the distribution of the considered video, and when and how they leave the system, stopping to contribute their upload bandwidth. To this aim, we adopt a very flexible model that allows to consider a non-stationary video request process, and general peer churn, which is an another crucial feature of any realistic P2P system.

In particular, we assume that the arrival process of requests for the considered video follows a time-varying Poisson process of intensity $\lambda(t)$. By so doing, we are able to capture effects related to content popularity variation, while maintaining analytical tractability [9], [10].Indeed, assuming that at a given time the arrival process is Poisson is reasonable, since users behave independently of each other, and their requests are immediately satisfied. On the other hand, a video (*e.g.*, a typical movie) can be long enough that the rate at which it is requested can vary significantly during the playing time, due to daily traffic fluctuations, or rapidly-changing video popularity. We account for this fact assuming that the video request process is described by an non-homogeneous Poisson process with time-varying intensity $\lambda(t)$, which can possibly

1. In this paper we use the terms peer and user interchangeably.

be equal to zero before a given time t_0, to model a newly introduced video inserted into the catalog at time t_0.

The dynamics of peer participation in the distribution of a given video must account for the fact that activity periods of the users are highly heterogeneous, as observed in several measurement studies [4]: some users stop watching the video after a very short time since the beginning, because they realize they are no longer interested in it; most users who decide to watch the video shut down the computer/Internet-TV towards the end of it; some of them keep the application running for prolonged time after the end of the video; those running set-top-boxes can be considered to be always active and serving other peers (until they stop contributing to the distribution of the considered video). We account for general user behavior assuming that the activity period of a user (*i.e.*, the interval during which a user contributes its upload bandwidth, starting from the instant at which the video has been requested) is described by an arbitrary random variable T with finite mean $\overline{T}$ and complementary cumulative distribution function $G_T(x)$. The activity periods of the users are assumed to be i.i.d.

It follows from our assumptions that the number of active users $N(t)$ at time t is distributed as the number of customers in an M/G/∞ queue with time-varying arrival rate, hence it follows a Poisson distribution with time-varying mean $\overline{N}(t)$ given by

$$\overline{N}(t) = \int_0^\infty \lambda(t-x) G_T(x)\,\mathrm{d}x \tag{1}$$

In our analysis we need to distinguish two classes of active users: those who are still *downloading* the video, and those who have completed the download (referred to as *seeds* in the following). Let $\tau_d = L/d$ be the time needed to download the whole video, and $\overline{T}_d = \int_0^{\tau_d} G_T(x)dx$ the average time spent by peers downloading the video. The number of *downloading* peers at time t, denoted by $N_d(t)$, follows a Poisson distribution of mean $\overline{N}_d(t)$ given by

$$\overline{N}_d(t) = \int_0^{\tau_d} \lambda(t-x) G_T(x)\,\mathrm{d}x \tag{2}$$

Then standard properties of Poisson processes allow to say that the number of *seeds* at time t, denoted by $N_{\text{seed}}(t)$, follows a Poisson distribution of mean $\overline{N}_{\text{seed}}(t) = \overline{N}(t) - \overline{N}_d(t)$.

We define as instantaneous system load $\gamma(t)$ the quantity

$$\gamma(t) = \frac{d \cdot \overline{N}_d(t)}{\overline{U} \cdot \overline{N}(t)} \tag{3}$$

which is the ratio between the average data rate requested at time t by downloading peers and the average upload rate provided by all active users at time t. Borrowing the terminology adopted in previous work [3], [11] we say that at time t the system operates in *deficit* mode if $\gamma(t) > 1$, in *balanced* mode if $\gamma(t) = 1$, and in *surplus* mode if $\gamma(t) < 1$.

We also introduce the per-user system load $\gamma_p = \frac{d \cdot \overline{T}_d}{\overline{U} \cdot \overline{T}}$, which is the ratio between the average amount of data that are downloaded by a peer, and the average amount of data that a peer is able to offer to other peers. Note that by construction γ_p is equal to the (constant) instantaneous system load in the case of a stationary user arrival process. In ergodic systems, γ_p can be regarded as the time average of $\gamma(t)$.

TABLE 1: Notation

Symbol	Definition
L	video size (bytes)
d_v	video playback rate (bytes/s)
T_v	video playback duration (s), $T_v = L/d_v$
d	target download rate (bytes/s)
$\overline{U}$	average user upload bandwidth (bytes/s)
$\overline{T}$	average user activity period (s)
$\overline{T}_d$	average time spent downloading the video (s)
$\lambda(t)$	arrival rate of requests for the video at time t
$\overline{N}(t)$	average number of active users at time t
$\overline{N}_d(t)$	average number of *downloading* users at time t
$\overline{N}_{seed}(t)$	average number of *seeds* at time t
$\overline{S}_d(t)$	average bandwidth requested by downloaders at time t
$\overline{S}_{\text{seed}}(t)$	average bandwidth offered by seeds at time t
$\overline{S}(t)$	average bandwidth requested from the servers at time t

2.3 Performance metrics

The main goal of our paper is to characterize the additional bandwidth required from the servers (i.e., in addition to the aggregate bandwidth provided by the peers) to guarantee an optimal quality of service (*i.e.*, continuity index equal to 1) to the users.

Let $S(t)$ be the random variable denoting the additional bandwidth that the servers must supply at time t to satisfy all active downloads of the considered video. We denote by $\overline{S}(t)$ and $\sigma_S^2(t)$ the mean and variance of $S(t)$, respectively.

Since in practice there are multiple videos to be served concurrently by the system, statistical multiplexing arguments suggest that a good design goal is to minimize the mean value $\overline{S}(t)$ of the server bandwidth required by a single video. Therefore, this will be the main metric that we will look at in our performance analysis. Table 1 summarizes the notation of our model.

3 Analysis

3.1 Sequential delivery

We start considering the simple case in which users download the video chunks sequentially. This scheme is simple to implement, as it does not require complex chunk/peer selection mechanisms such as those needed in BitTorrent-like chunk swarming schemes. More importantly, the sequential delivery scheme is analytically tractable and provides an upper bound to the server bandwidth requested by non-sequential schemes.

Let $S_d(t)$ be the aggregate bandwidth requested by the downloading users at time t, and $S_{\text{seed}}(t) = \sum_{i=1}^{N_{\text{seed}}(t)} U_i$ be the aggregate upload rate offered by the seeds at time t. Then the bandwidth requested from the servers at time t is given by

$$S(t) = \max\{0, S_d(t) - S_{\text{seed}}(t)\}. \tag{4}$$

Focusing on $S_d(t)$, we first condition this quantity on the number of downloading users k, defining

$$S_d(t,k) \triangleq (S_d(t) \mid N_d(t) = k)$$

After characterizing $S_d(t,k)$, the evaluation of $S(t)$ is easy, since the distribution of $N_d(t)$ is known (a Poisson distribution of mean $\overline{N}(t)$), while $S_{\text{seed}}(t)$ is a compound Poisson random variable which does not depend on k [12].

To evaluate $S_d(t,k)$ under sequential download, we start observing that, if all peers download the video sequentially at common rate d, a peer can only redistribute video pieces to peers arrived later on in time.

Proposition 1: Quantity $S_d(t,k)$ satisfies the following recursive equation:

$$S_d(t,k) = \begin{cases} d & k=1 \\ d + \max\{0, S_d(t,k-1) - U_k\} & k>1 \end{cases} \quad (5)$$

The proof of Proposition 1 is reported in Appendix A of [8]. Expression (5) provides the key to the analytical approximation developed in the next section.

Alternate formulations of quantity $S_d(t,k)$ exist (see [3], [11], [7]). Here, we just mention that in [7] we find a connection between the stochastic process described by (5) and a random walk with increments $d-U$, which allows to obtain analytical upper bounds to the server bandwidth and to characterize its asymptotic behavior for large number of users.

3.2 Gaussian approximation

In the sequential delivery case, we can characterize the distribution of the server bandwidth using a second-order approximation [12], [13].

The idea is to approximate the distribution of quantity $S_d(t,k-1) - U_k$ in (5) (for each $k \geq 2$) by a normal distribution matching the first two moments of this quantity. We can then apply standard formulas of the truncated normal distribution to derive the first two moments of $S_d(t,k)$ as a function of the first two moments of $S_d(t,k-1)$. This provides a recursive technique to compute the first two moments of $S_d(t,k)$ for any k, starting from the exact values known for $k=1$. Our gaussian approximation is motivated by the fact that significant excursions of $S_d(t)$ (*i.e.*, away from the lower limit d) result from the accumulation of several random contributions $d-U$, thus the central limit theorem can be invoked to justify the convergence to a normal distribution. A similar approximation is subsequently applied to take into account the effect of the *seeds*, whose aggregate contribution $S_{\text{seed}}(t)$ can be well described by a gaussian distribution for sufficiently large number of seeds. Notice that an exact evaluation of the distributions (or just the first few moments) of $S(t)$ and $S_d(t,k)$ is difficult, due to the presence of barriers at zero and d, respectively.

More in detail, let us start introducing some notation and standard results related to the normal distribution. Let $\mathcal{N}(w)$ be the probability density function of the standard normal distribution (having mean 0 and variance 1), and $Q(w)$ its complementary cumulative distribution function.

Let y be a random variable distributed according to a normal distribution $N(\mu,\sigma)$ of mean μ and standard deviation σ. Then it can be proved that the first moment of the random variable $y' = \max\{0,y\}$ has the following expression:

$$\mathbb{E}[y'] = \sigma\mathcal{N}\left(-\frac{\mu}{\sigma}\right) + \mu\, Q\left(-\frac{\mu}{\sigma}\right) \quad (6)$$

while the second moment is given by

$$\mathbb{E}[y'^2] = \sigma\mu\mathcal{N}\left(-\frac{\mu}{\sigma}\right) + (\sigma^2+\mu^2)Q\left(-\frac{\mu}{\sigma}\right). \quad (7)$$

Let $\overline{S}_d(t,k)$ and $\sigma^2_{S_d}(t,k)$ be the mean and variance of $S_d(t,k)$. Our recursive procedure to approximately compute $\overline{S}_d(t,k)$ and $\sigma^2_{S_d}(t,k)$ for all k starts from the initial known values $\overline{S}_d(t,1) = d$ and $\sigma^2_{S_d}(t,1) = 0$ (see (5)). For a given $k \geq 2$ we approximate $S_d(t,k-1) - U_k$ by a normal random variable y of mean $\mu = \overline{S}_d(t,k-1) - \overline{U}$ and variance $\sigma^2 = \sigma^2_{S_d}(t,k-1) + \sigma^2_U$. Defining the random variable $y' \triangleq \max\{0,y\} \simeq \max\{0, S_d(t,k-1)-U\}$, from (5) we obtain:

$$\overline{S}_d(t,k) \simeq d + \mathbb{E}[y'] \quad (8)$$

$$\sigma^2_{S_d}(t,k) \simeq \mathbb{E}[y'^2] - \mathbb{E}[y']^2. \quad (9)$$

Applying (6) and (7) we can compute the first and second moment of variable y' in (8,9). This provides the recursion to compute $\overline{S}_d(t,k)$ and $\sigma^2_{S_d}(t,k)$ for all k.

To account for the effect of the *seeds* (if any), we apply once more the normal approximation, as follows. Let $S(t,k) = \max(0, S_d(t,k) - N_{\text{seed}}(t))$ be the server bandwidth necessary at time t, assuming that there are k downloading users and $N_{\text{seed}}(t)$ seeds. Moreover, let $\overline{S}(t,k)$ and $\sigma^2_S(t,k)$ be the mean and variance of $S(t,k)$.

We observe that $S_{\text{seed}}(t)$ is a compound Poisson random variable, whose moments can be computed exactly in closed-form. In particular, the mean of $S_{\text{seed}}(t)$ is equal to $\overline{N}_{\text{seed}}(t)\overline{U}$, whereas its variance is equal to $\overline{N}_{\text{seed}}(t)(\sigma^2_U + \overline{U}^2)$. We approximate $S_d(t,k) - S_{\text{seed}}(t)$ by a normal distribution y of mean $\mu = \overline{S}(t,k) - \overline{N}_{\text{seed}}(t)\overline{U}$ and variance $\sigma^2 = \sigma^2_S(t,k) + \overline{N}_{\text{seed}}(t)(\sigma^2_U + \overline{U}^2)$, and apply again (6) and (7) to compute the first and second moment of $y' = \max\{0,y\} \simeq S(t,k)$.

Finally, the mean server bandwidth $\overline{S}(t)$ (and similarly its variance) can be obtained deconditioning with respect to k:

$$\overline{S}(t) = \sum_{k\geq 1} \overline{S}(t,k)\mathbb{P}(N_d(t)=k) \quad (10)$$

We observe that each step of the iterative procedure requires a constant number of operations. Hence the computational complexity of the model solution is linear in the number of steps (k), which equals the number of downloading users in the systems. This number is theoretically unbounded (it has a Poisson distribution), however we can reasonably limit the iteration to a maximum number of steps $k_{\max}$ such that $\mathbb{P}(N_d(t) > k_{\max})$ is negligible (i.e., smaller than a given small constant ϵ; in our results we set $\epsilon = 10^{-6}$). The computational complexity is then $\Theta(k_{\max})$.

3.3 Extension to non-sequential delivery

So far we have restricted the analysis to the case in which users receive the video chunks sequentially. Although conceptually simple, this scheme is clearly sub-optimal when users download data at a rate larger than the playback rate: recall that in

this case users can download in advance video chunks needed in the future, and this prefetching does not necessarily have to be done sequentially. Actually, by allowing out-of-sequence delivery the system can better exploit the upload bandwidth of the peers. Chunk-based, swarming approaches like those commonly used in P2P bulk transfers (*e.g.*, BitTorrent) can be applied to VoD systems with the additional constraint that individual chunks must be downloaded before specific deadlines to avoid interrupting the video playback.

A common approach to combine the efficiency of P2P swarming with the strict delay constraints of VoD is to allow users to receive also out-of-sequence chunks of the video within a limited "sliding window" of data starting from the point currently played [5].

For simplicity, instead of considering an actual sliding window, we divide the video into a fixed number W of non-overlapping segments of size $L_W \triangleq L/W$. We denote by $T_W \triangleq L_W/d = \tau_d/W$ the time needed to download a segment. Users who are concurrently downloading the same segment, besides helping users downloading previous segments can help each other in a swarming fashion, *i.e.*, we assume that, within a segment, we can exploit also chunk-based, out-of-sequence distribution. This model is able to capture the behavior of realistic sliding window applications, while keeping the analysis simple.

Below we show how the analysis developed in Section 3.1 can be adapted to study this scheme as well, permitting us to assess the performance gain achievable by non-sequential schemes.

Indeed, by aggregating all users belonging to the same segment into a sort of 'meta-peer' we can essentially apply the same analysis as before to a chain of W meta-peers downloading the video segments sequentially. Let $N_v(t)$, $1 \leq v \leq W$, be the random variables representing the number of peers concurrently downloading segment v at time t. Notice that this number is Poisson-distributed [12] with mean

$$\overline{N}_v(t) = \int_0^{T_W} \lambda(t-(v-1)T_W - x) G_T(x + (v-1)T_W)\,\mathrm{d}x$$

Let $S_d(t,v)$ be the bandwidth that the servers (or the seeds) must supply at time t to all users downloading segments of index smaller than or equal to v. Using the same reasoning as in the proof of Proposition 1, quantity $S_d(t,v)$ can be computed through the following recursive equation,

$$S_d(t,v) = \max\Big\{0, S_d(t,v-1) - \sum_{i=1}^{N_v(t)} U_i + \sum_{i=1}^{N_v(t)-1} d\Big\} + d\cdot \mathbb{I}_{N_v(t)>0} \qquad 1 \leq v \leq W \quad (11)$$

with $S_d(t,0) = 0$ and the convention that summations are equal to zero if $N_v(t) = 0$. Notice that (11) is analogous to (5), if we consider all peers within a segment v (if any) as a single meta-peer having virtual upload bandwidth $\tilde{U}(t,v) = d + \sum_{i=1}^{N_v(t)}(U_i - d)$.

The first two moments of $S_d(t) = S_d(t,W)$ can be computed using a second-order approximation similar to the one adopted for the case of sequential delivery in Section 3.2, *i.e.*, by assuming that quantity $S_d(t,v-1) - \tilde{U}$ (whose moments can be computed exactly) has a normal distribution, and then using (6) and (7) to compute the first and second moments of the positive part of it [12], [13].The analysis is made slightly more complicated by the fact that we need to consider also the case in which there are no users downloading a segment ($N_v(t) = 0$), which requires some care. Details are reported in Appendix B of [8].

At last, the impact of seeds is taken into account in a way analogous to the sequential case. Note that the computational procedure for the non-sequential case is again linear in the number of steps, hence it is $\Theta(W)$.

The extreme case $W = 1$ of just one segment (*i.e.*, the entire video) corresponds to a scheme in which chunks can be downloaded in any order, and the system can exploit the upload bandwidth of any peer irrespective of its arrival time. In this case we have,

$$S_d(t,1) = \max\Big\{d, \sum_{i=1}^{N_d(t)} (d - U_i)\Big\} \quad (12)$$

which plugged into (4) (in the place of $S_d(t)$) provides a lower bound to the server bandwidth required by any chunk distribution scheme. In the following, we will refer to the server bandwidth obtained in this way as the completely non-sequential case, or simply the *lower bound*.

In Sect 7 of [8], it is shown how our model can be extended to analyze systems providing limited VCR functionalities as well as, systems in which peers have limited storage capabilities.

4 Performance under stationary conditions

In this section we report a selection of the most interesting results that we have obtained by our analysis under stationary user arrival process. Since in this case all averages do not depend on t, we will omit for simplicity the indication of time. We normalize to 1 the video playback rate d_v, which thus serves as unit for all other bandwidth figures. We assume that users stay in the system for a time at least equal to the watching time, hence $\overline{T} \geq T_v$. Unless otherwise specified, we assume that users' upload bandwidth U is exponentially distributed.

The results obtained by our analytical approximation in Section 3.2 are compared against those obtained by an event-driven simulator derived from P2PTVsim[2]. In particular, we adapted P2PTVsim, originally developed for live-streaming, to represent VoD systems. Our simulator permits considering a general mesh-based pull system: the content is segmented into small fixed-size chunks, corresponding to 100 ms of video; peers independently retrieve individual chunks from other peers and/or the servers adopting a simple trading mechanism that involves a periodic exchange of signaling messages containing buffer maps between pairs of peers. To guarantee an almost in-order delivery of chunks (as in the case of VoD) we

2. P2PTVsim is available at http://www.napa-wine.eu/cgi-bin/twiki/view/Public/P2PTVSim

implemented a sliding window mechanism: each peer playing chunk c is allowed to retrieve only the chunks $[c, c+H]$ where H is the window parameter. A sequential delivery scheme is then approximately achieved by choosing a small value of H (compared to the entire video). In our simulations (if not otherwise specified) we have set $H = 40$, such that a window corresponds to only 4 seconds of video. Chunks are, in the first instance, retrieved from other peers; a peer is allowed to retrieve directly from servers the immediately following chunk to play (i.e., chunk $c+1$ can be retrieved from the servers only while playing chunk c). Servers are assumed to have unlimited bandwidth.

We recognize that our simulator may appear rather simplified with respect to the behavior of real systems, as it shares most of the system assumptions of the model (see Section 11 in [8]).We use it primarily to validate the accuracy of the Gaussian approximation introduced in Section 3.2, that enables us to assess the system performance at very low computational cost (i.e., with negligible effort as compared to detailed simulations or measurement campaigns).

We emphasize that the ultimate goal of our model and analysis is not to produce accurate quantitative results about the performance of a realistic system (that necessarily depends on the specific system implementation), but to allow for an efficient qualitative prediction of the impact of various parameters and design choices on the resulting system performance.

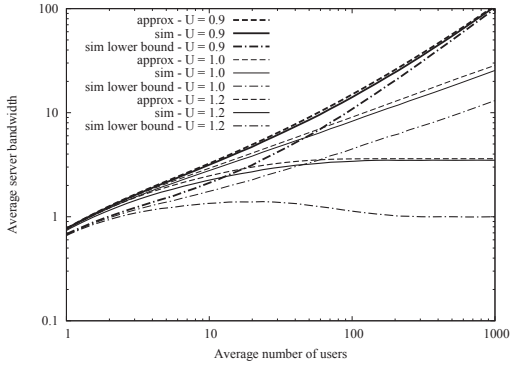

Fig. 1: Comparison of average server bandwidth in the case $d = d_v$, as function of the average number of users $\overline{N}$, for different values of $\overline{U}$, in the absence of seeds.

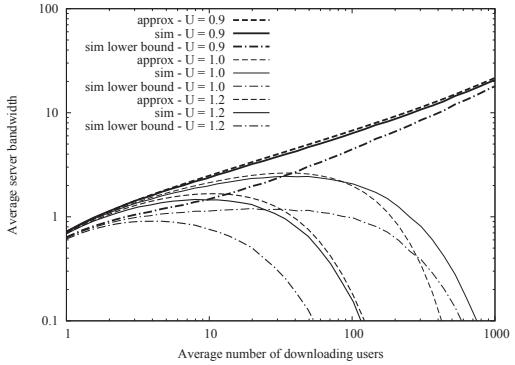

Fig. 2: Comparison of average server bandwidth in the case $d = d_v$, as function of the number of downloading users $\overline{N}_d$, for different values of $\overline{U}$, in the presence of $\overline{N}_{seed} = 0.1 \cdot \overline{N}_d$ seeds.

4.1 Impact of the number of watching users and seeds

Figure 1 reports, on a log-log scale, the average server bandwidth $\overline{S}$ as function of the average number of users $\overline{N}$, in the case $d = d_v$, $\overline{T} = T_v$. We consider three different values of average upload bandwidth $\overline{U} = 0.9, 1.0, 1.2$, corresponding to systems operating in deficit, balanced, and surplus mode, respectively (here $\gamma = 1/\overline{U}$).

Besides noticing the accuracy of the approximate analysis, it is interesting to see that the average server bandwidth saturates for $\overline{U} = 1.2$ (surplus mode) to a value about 3.5 times larger than the corresponding lower bound, which tends to $d_v = 1$. As expected, the average server bandwidth diverges under the deficit and balanced modes. Moreover, in the deficit mode, the sequential system requires asymptotically the same bandwidth as the completely non-sequential system (*i.e.*, the lower bound).

In Figure 2 we compare the results obtained in the same system considered above, but assuming that users remain active after the end of the watching time for an exponentially distributed amount of time of mean equal to 10% of the watching time, generating an average number of seeds $\overline{N}_{seed} = 0.1 \cdot \overline{N}_d$. Now the systems with $\overline{U} = 1.0$ and $\overline{U} = 1.2$ operate in surplus mode, whereas the system with $\overline{U} = 0.9$ operates very close to the balanced mode (here $\gamma = 1/(1.1 \cdot \overline{U})$). We observe that, in the presence of seeds, the average server bandwidth requested by systems operating in surplus mode reaches a maximum, after which it goes to zero as the number of users increases. Results such as those reported in Figures 1 and 2 can be useful in system dimensioning, as they allow to estimate, in the surplus mode, the worst-case server bandwidth which is needed when the number of downloading users $\overline{N}_d$ is not known.

4.2 Impact of the target download rate

Even when users tend to leave the system at the end of the watching time, it is still possible to benefit from the positive effect created by the seeds, who absorb part of the fluctuations in the bandwidth requested by downloading peers, shielding the servers. The trick to 'artificially' create some seeds is to make the users download the video at rate $d > d_v$, so that they become seeds for other peers before the end of the watching time. Intuitively, however, d should not be set too large to offset the gain achievable by the seeds. Figure 3 illustrates the performance of this strategy in the case of $\overline{N} = 100$ users, $\overline{T} = T_v$, showing the average server bandwidth as function of d. We observe that for all the considered values of $\overline{U}$, the average server bandwidth achieves a minimum for a value of d slightly larger than d_v. The impact is particularly striking in the surplus mode ($\overline{U} > 1$), in which setting $d > d_v$ brings the server bandwidth close to zero.

4.3 Impact of non-sequential delivery

To understand the performance gains achievable by adding (in a limited way) BitTorrent-like chunk delivery schemes to the video distribution, we consider two scenarios operating in

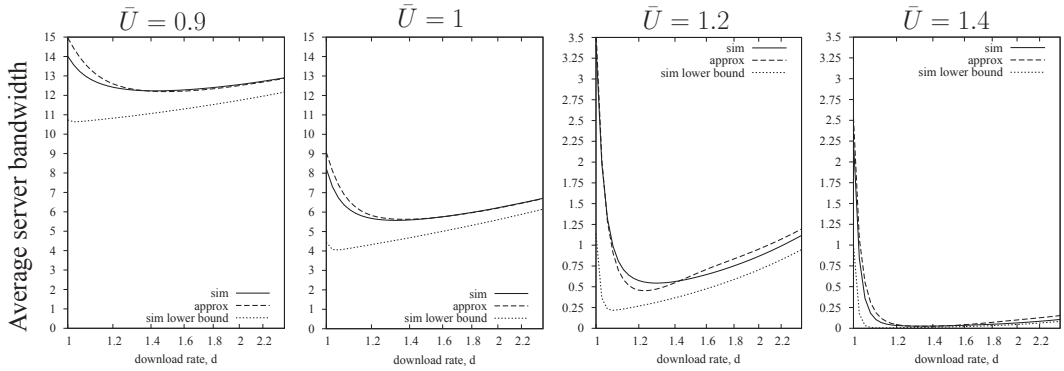

Fig. 3: Average server bandwidth as function of the target download rate d, for different values of $\overline{U}$, with $\overline{N} = 100$, $\overline{T} = T_v$.

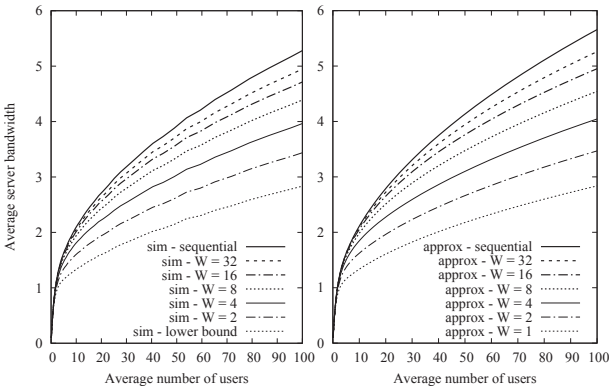

Fig. 4: Average server bandwidth as function of the number of users, with $\overline{T} = T_v$, $d = d_v$, $\gamma = 1$, for different values of the number of segments W. Comparison between simulation (left plot) and approximate analysis (right plot).

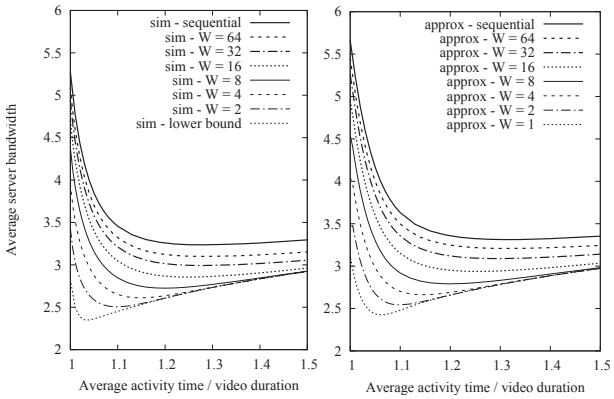

Fig. 5: Average server bandwidth as function of the ratio $\overline{T}/T_v$ between average activity time and video duration, keeping fixed $\overline{N} = 100$, $d = d_v$, $\gamma = 1$, for different values of the number of segments W. Comparison between simulation (left plot) and approximate analysis (right plot).

balanced mode ($\gamma = 1$), in the case of uniformly distributed user upload bandwidth. In the first one (see Figure 4) we vary the number of users, assuming $\overline{T} = T_v$, $d = d_v$. Recall that the case $W = 1$ in our analysis provides an approximation of the lower bound.

As we increase the number of segments, we obtain intermediate curves between the completely non-sequential scheme and the pure sequential scheme, which corresponds to $W \to \infty$. Besides noticing the accuracy of the approximation, we observe that swarming schemes provide non-negligible gains over pure sequential schemes only when the average number of users per segment is not too small (say larger than a few units). Recall, however, that larger segments (and thus larger number of users in them) imply larger startup delays. For example, the case $W = 32$ corresponds to a sliding window of about 4 minutes for a typical movie (2 hours long). As we can see on Figure 4, the benefit of a non-sequential scheme with $W = 32$ is negligible for the considered values of the number of users (below one hundred).

In the second scenario (see Figure 5) we fix the total number of users $\overline{N} = 100$, and increase the total activity time of the users $\overline{T}$ (the average upload bandwidth $\overline{U}$ is reduced accordingly to keep $\gamma = 1$). We observe that as we increase the activity time (and thus the number of seeds), the difference in performance between swarming schemes and sequential delivery becomes less significant.

5 Non-stationary systems

In this section we show how our analytical framework can be applied to study the performance of time-varying systems in which the arrival rate of requests for a given content changes significantly over time. In particular, we will see that the behavior of a non-stationary system can dramatically differ from the one of a stationary system in which the arrival rate of requests is constant, due to a misalignment problem between the temporal evolution of the number of downloaders and the temporal evolution of the number of seeds.

5.1 The effect of daily traffic variations

We consider a reference scenario in which the arrival rate of requests for a given video follows a daily pattern which is modeled for simplicity by a sine function of period equal to 24 hours, between a minimum of $\lambda = 0.1$ and a maximum of $\lambda = 1$, represented by the thick solid line in the top plot of Fig. 6.

We first analyze a *software-based* system in which users contribute their upload bandwidth during the watching time of the video, plus a random additional time in which the application is kept running. We assume that the video duration is $T_v = 2$ h, and the additional activity time after the end of the video is exponentially distributed with mean 1 h. We normalize

$d_v = d = 1$ and assume the upload bandwidth of users to be exponentially distributed with mean $\overline{U} = 0.7$. The per-user load is $\gamma_p = 2/(0.7 \cdot 3) \approx 0.95$. The top plot of Fig. 6 reports the temporal evolution of both the number of downloaders and the number of seeds. Since peers become seeds only after the end of the watching time, the dynamics of downloaders and seeds are misaligned, with a temporal shift about $T_v = 2$ h.

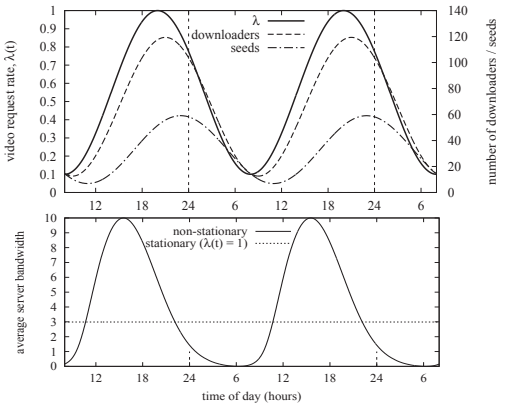

Fig. 6: Temporal evolution of video request rate (top plot, left y axes), number of downloaders/seeds (top plot, right y axes), and average server bandwidth $\overline{S}(t)$ (bottom plot), in the *software-based* system.

The effect of this misalignment on the required average server bandwidth is depicted on the bottom plot of Fig. 6, by the solid line labeled 'non-stationary', which exhibits a peak preceding the point at which the video request rate is maximum. Fig. 6 reports also a curve labeled 'stationary', representing the server bandwidth that would be necessary if the content request rate were constant and equal to $\lambda = 1$ (the maximum request rate). We observe that the performance of the non-stationary system is worse than that of the stationary system, both in terms of peak server bandwidth and average server load. This occurs even if the content request rate is always larger in the stationary system.

If we increase the activity time after the end of the video, while keeping the same per-user system load γ_p (either by reducing the upload bandwidth of the users, or equivalently by increasing the download rate of the video, *i.e.*, its resolution/quality), the negative effect of the misalignment problem become worse. As an extreme case, we consider a P2P-VoD system relying on *set-top-boxes* which are always active and serving the last watched video. To mimic the behavior of set-top-boxes with our model of peer dynamics, we assume that the activity time after watching a movie is much longer than before (in the order of a day), representing a set-top-box which remains always on before the user downloads the next video. In particular, we consider an additional activity time of 22 h, which added to the watching time of a movie leads to $\overline{T} = 1$ day. To obtain the same per-user load of the *software-based* system, we set $\overline{U} = 0.7 \cdot 3/24$.

Fig. 7 reports analytical results for this scenario, analogous to those in Figure 6. We have also reported on the bottom plot of Fig. 7 a sample path obtained from simulation, to confirm the analytical prediction. In this case the instantaneous system load $\gamma(t)$ is severely unbalanced across the day. During peak

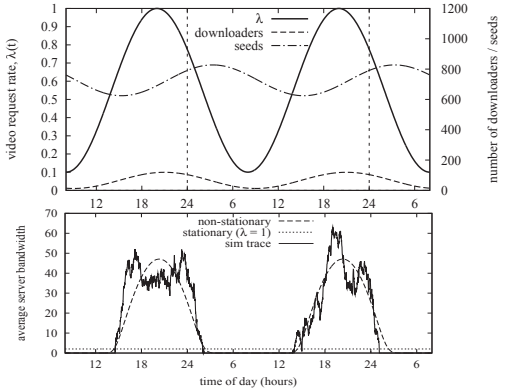

Fig. 7: Temporal evolution of video request rate (top plot, left y axes) number of downloaders/seeds (top plot, right y axes), and average server bandwidth $\overline{S}(t)$ (bottom plot), in the *set-top-box* system.

hours, the bandwidth requested at servers grows very large, while for the rest of the day it is negligible. The problem is that the upload capacity of the seeds, which are very numerous and almost stable along the day (see top plot of Figure 7) is totally wasted for a large fraction of the day. Notice that we are not saying that *set-top-boxes* are not useful: increasing the activity time of peers (up to the point of having always-on user devices) is very beneficial to the system performance, since the per-user load γ_p is reduced. However, one must be careful that the instantaneous system load $\gamma(t)$ can vary significantly around γ_p, and peak traffic demand cannot be absorbed well by large populations of seeds (*set-top-boxes*) each devoting a small amount of upload bandwidth to the video distribution. Indeed, to minimize the bandwidth deficit at peak times, the ratio $\overline{U}/d$ should not become too small.

5.2 The effect of newly introduced videos

We now consider a system in which the non-stationarity of the video request process is due to new contents regularly introduced in the catalogue, whose popularity changes over time. We model such a system using a *shot-noise* process [14]. We assume that new videos are made available in the system at (constant) rate β. The request rate of a given video i inserted at time t_i follows an non-homogeneous Poisson process with time-varying intensity

$$\lambda_i(t) = \Lambda \phi(t - t_i)$$

where $\phi(t)$ is a shaping function, defined over $t \geq 0$, modeling how the popularity of the video evolves over time. We require $\phi(t)$ to be an integrable function, and, without loss of generality, we assume that $\int_0^\infty \phi(t)\,\mathrm{d}t = 1$. By so doing, Λ equals the average number of times that the video is requested during its lifetime. In general, both Λ and $\phi(t)$ could be random, *i.e.*, each file, upon arrival, could be assigned a popularity shape $\phi(t)$ extracted from a family of functions with randomized parameters, and a value of Λ taken from a given distribution possibly associated to the chosen shape $\phi(t)$. We denote by $F_{\Lambda,\phi}$ the joint cdf of Λ and ϕ.

Our target is to evaluate the average overall server bandwidth Z resulting from the distribution of all videos available

in the catalog. Despite the complexity of the system, our approximate model can be exploited to quickly estimate Z under several parameter settings, without running expensive simulations. We briefly summarize the computational procedure for clarity: using (1) and (2) we can analytically compute, for any $t > t_i$, the average number of peers downloading a given video i, and the average number of users acting as seeds for it. Equation (10) provides the mean bandwidth $\overline{S}(t, \Lambda, \phi)$ requested from the servers at any time $t > t_i$. Standard numerical integration techniques[3] allow to compute the average amount of data $D(\Lambda, \phi) = \int_{t_i}^{\infty} \overline{S}(t - t_i, \Lambda, \phi)\, \mathrm{d}t$ supplied by the servers for a file characterized by popularity parameters $\{\Lambda, \phi\}$, from which we can compute

$$Z = \beta \int_{\Lambda,\phi} D(\Lambda, \phi)\, \mathrm{d}F_{\Lambda,\phi} \tag{13}$$

Notice that the average number of users in the system is $\beta\Lambda\overline{T}$.

We start considering a scenario in which both Λ and ϕ are deterministic. In particular, we consider files with exponentially decreasing popularity: $\phi(t) = \mu e^{-\mu t}$. Since Z is trivially linear in the arrival rate of new contents (13), we arbitrarily set $\beta = 1$. Throughout our experiments we also fix the per-user system load to $\gamma_p = 0.5$, as the effects that we are going to show occur also when users can upload a much larger amount of data than what they request during their activity period. We consider the pure sequential delivery scheme, and normalize $d_v = d = 1$. We further normalize $\overline{T}_d = 1$, and assume, whenever $\overline{T} > \overline{T}_d$, that the additional activity time after downloading the movie is exponentially distributed. Under the above settings the average upload bandwidth of the users, which is assumed to be exponentially distributed, is directly related to $\overline{T}$ according to $\overline{U} = 2/\overline{T}$. At last, we define $\chi = 1/(\mu\overline{T}_d)$, which can be thought as the average lifetime of a video (in terms of popularity) normalized by its watching duration. We now investigate the joint impact of the only free parameters: Λ, χ, and $\overline{T}$.

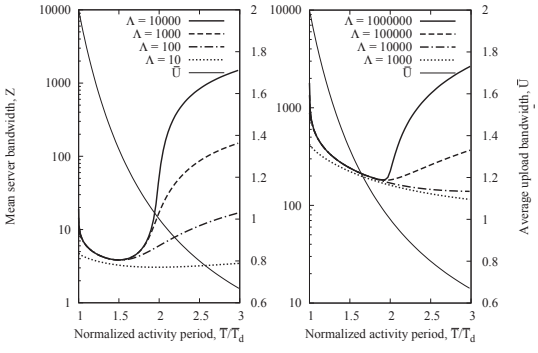

Fig. 8: Average server bandwidth Z as function of normalized user activity period $\overline{T}/\overline{T}_d$, for $\chi = 1$ (left plot) and $\chi = 100$ (right plot), and different values of Λ. The associated average upload bandwidth $\overline{U} = 2/\overline{T}$ is reported on the right y axes.

Fig. 8 reports, for different Λ's, the average server bandwidth Z as function of user activity period $\overline{T}$, for $\chi = 1$ (left plot) and $\chi = 100$ (right plot). We observe that the server bandwidth Z remains low and mainly independent of Λ (*i.e.*, the system can scale up to arbitrarily large number of users), for values of $\overline{T}$ slightly larger than 1 and smaller than 2. Notice that when $\overline{T} < 2$ the average user upload bandwidth (right y axes) is larger than the video download rate ($\overline{U} > 1$). When this condition is not met (in general, when $\overline{U} \leq d_v$), the system does not sustain itself, despite the fact that users can potentially upload twice the amount of traffic they download ($\gamma_p = 0.5$). This is again due to the misalignment problem between downloaders and seeds: when $\overline{T}$ increases, more seeds becomes available, but too late with respect to the time their upload capacity can be exploited. We emphasize that this is in sharp contrast to what we have seen under stationary conditions, where the effect of decreasing $\overline{U}$ can be completely compensated by increasing $\overline{T}$, so that the system performance is not compromised (see Figure 5). We further notice that this asymptotic behavior (for increasing Λ) occurs for any value of χ (*i.e.*, popularity shape), and that larger values of χ (*i.e.*, files whose popularity decays more slowly) lead to larger values of Z in the regime where the system is able to scale.

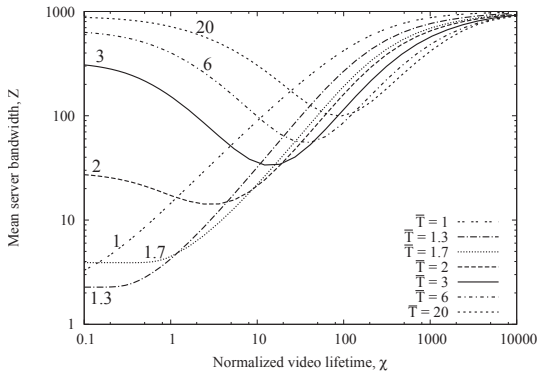

Fig. 9: Average server bandwidth Z as function of the normalized video lifetime χ, for different values of user activity period $\overline{T}$, in the case of $\Lambda = 1000$.

The impact of the video popularity shape is better understood looking at Fig. 9, which reports, in the case of fixed $\Lambda = 1000$, the server bandwidth Z as function of the normalized video lifetime χ, for different values of $\overline{T}$. First, note that Z is bounded above by $\beta\Lambda\overline{T}_d d_v$, since in the worst case the system provides rate d_v to each downloading user. In our parameters setting, the above upper bound on Z coincides numerically with $\Lambda = 1000$. We observe that Z approaches the upper bound for large values of χ (and any $\overline{T}$), for which requests for the same file are so diluted over time that it becomes more and more unlikely to have peers concurrently downloading the same file (and thus helping other peers). When $\overline{T}$ increases, the upper bound is approached also for small values of χ, this time because downloads of the same file are so synchronized that the upload bandwidth of users (which decreases with $\overline{T}$) can be exploited only during a short interval equal to $\overline{T}_d$ after the file is inserted into the catalog, and the resulting contribution of peers tends to become negligible. Small values of χ can be the result of videos posted on web pages providing suggestions to the users which are frequently updated: our results suggest that this practice can be harmful as it can compromise an effective peer-assisted distribution

3. We adopted the simple trapezoid rule.

(when $\overline{U} \leq d_v$). For $\overline{T} > 2$, Z achieves a minimum for a given value of χ. We observe that, for large values of χ (files with slowly decaying popularity), long user activity times are actually beneficial to the system, although in this regime the system is not able to scale to large number of users, as we have seen.

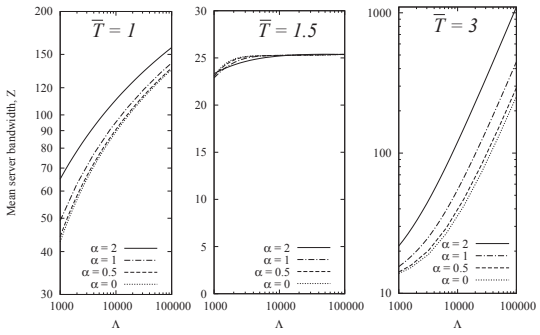

Fig. 10: Average server bandwidth Z as function of Λ, for different values of the Zipf's exponent α, and fixed $\chi = 10$. The user activity period is either $\overline{T} = 1$ (left plot), or $\overline{T} = 1.5$ (middle plot) or $\overline{T} = 3$ (right plot).

At last, we consider a scenario in which Λ is a random variable, while maintaining the assumption that the popularity decay exponent μ is the same for all files. More specifically, we assume that files belong to 10 different classes, whose request rates follows a Zipf's distribution of exponent α, *i.e.*, files of class i ($1 \leq i \leq 10$) are requested at rate $\Lambda_i = \Lambda K / i^{\alpha}$, where K is a normalizing constant such that $\sum_{i=1}^{10} \Lambda_i = \Lambda$. Note that $\alpha = 0$ corresponds to the previous scenario in which all files have the same popularity profile. Results are shown in Fig. 10, in which we report the mean server bandwidth Z as function of Λ, for different values of the Zipf's exponent α, and fixed $\chi = 10$. In the left plot we consider $\overline{T} = 1$, i.e., users abandon the system immediately after downloading the video, *i.e.*, no seeds are available. We notice that the server bandwidth Z scales logarithmically with Λ, which can be explained by the lower limit d in (5). In the middle plot we consider $\overline{T} = 1.5$, which belongs to the range in which the system performance is mainly insensitive to Λ (see Fig. 8), and thus also to the Zipf's exponent α. In the right plot, we consider $\overline{T} = 3$, for which the system is not able to scale to large number of users. Actually, in this case Z scales linearly with Λ.

In conclusion we can say that in non-stationary scenarios the system performance critically depends on the relationship between the average peer upload bandwidth and the download rate: when $\overline{U} \leq d_v$ the bandwidth deficit cannot be effectively compensated by just increasing the seed availability (i.e, by increasing $\overline{T}$). Smart prefetching policies can in principle reduce the burden on the servers. However, prefetching policies can not be easily implemented in non-stationary (*e.g.*, flash-crowd) scenarios where contents are not available in advance, and their popularity cannot be easily predicted at the early stages.

6 Conclusions

We have proposed a computationally-efficient methodology to analytically estimate the server bandwidth requested in non-stationary peer-assisted VoD systems. Our approach is highly flexible, and can account for several important effects such as peer upload bandwidth heterogeneity, churning, non-sequential chunk delivery schemes.

We have shown that our analysis provides efficient, accurate predictions of all observed phenomena, providing a useful tool for the design of peer-assisted VoD systems.

References

[1] D. Ciullo, V. Martina, M. Garetto, E. Leonardi, and G. L. Torrisi, "Performance Analysis of Non-stationary Peer-assisted VoD Systems," in *INFOCOM Mini-Conference*, 2012.

[2] "Cisco Visual Networking Index: Forecast and Methodology, 2012–2017," white paper published on Cisco website, 2012.

[3] C. Huang, J. Li, and K. W. Ross, "Can Internet Video-on-Demand Be Profitable?" in *ACM SIGCOMM*, 2007.

[4] Y. Huang, T. Z. J. Fu, D. ming Chiu, J. C. S. Lui, and C. Huang, "Challenges, Design and Analysis of a Large-scale P2P VoD System," in *ACM SIGCOMM*, 2008.

[5] "PPLive, http://www.gridcast.cn/. GridCast, http://www.gridcast.cn/. PPStream, http://www.ppstream.com/. TVU, http://www.tvunetworks.com/. SopCast, http://www.sopcast.com/. Kankan, http://www.kankan.com/."

[6] M. Cha, P. Rodriguez, S. Moon, and J. Crowcroft, "On next-generation telco-managed P2P TV architectures," in *IPTPS*, 2008.

[7] D. Ciullo, V. Martina, M. Garetto, E. Leonardi, and G. L. Torrisi, "Stochastic Analysis of Self-Sustainability in Peer-Assisted VoD Systems," in *INFOCOM*, 2012.

[8] ——, "Peer-assisted VoD Systems: an Efficient Modeling Framework," in *Supplemental material*, 2013.

[9] H. Abrahamsson and M. Nordmark, "Program popularity and viewer behaviour in a large tv-on-demand system," in *ACM IMC*, 2012.

[10] M. Ahmed, S. Traverso, M. Garetto, P. Giaccone, E. Leonardi, and S. Niccolini, "Temporal locality in today's content caching: Why it matters and how to model it," *ACM Computer Communication Review*, vol. 43, no. 5, 2013.

[11] W. Wu and J. Lui, "Exploring the Optimal Replication Strategy in P2P-VoD Systems: Characterization and Evaluation," in *INFOCOM*, 2011.

[12] S. M. Ross, "Stochastic processes. 1996," 2001.

[13] W. Whitt, "Approximations for the gi/g/m queue," *Production and Operations Management*, vol. 2, no. 2, pp. 114–161, 1993.

[14] D. Daley and D. Vere-Jones, *An introduction to the theory of point processes*, Springer-Verlag, Ed., 1998.

[15] R. Kumar, Y. Liu, and K. Ross, "Stochastic Fluid Theory for P2P Streaming Systems," in *INFOCOM*, 2007.

[16] D. Wu, Y. Liu, and K. W. Ross, "Queuing Network Models for Multi-Channel P2P Live Streaming Systems," in *INFOCOM*, 2009.

[17] X. Yang and G. de Veciana, "Service Capacity of Peer to Peer Networks," in *INFOCOM*, 2004.

[18] D. Qiu and R. Srikant, "Modeling and Performance Analysis of BitTorrent-Like Peer-to-Peer Networks," in *ACM SIGCOMM*, 2004.

[19] N. Parvez, C. Williamson, A. Mahanti, and N. Carlsson, "Analysis of BitTorrent-like Protocols for On-Demand Stored Media Streaming," in *ACM SIGMETRICS*, 2008.

[20] B. Fan, D. Andersen, M. Kaminsky, and K. Papagiannaki, "Balancing Throughput, Robustness, and In-Order Delivery in P2P VoD," in *ACM CoNEXT*, 2010.

[21] B. Tan and L. Massoulie, "Optimal Content Placement for Peer-to-Peer Video-on-Demand Systems," in *INFOCOM*, 2011.

[22] W. Wu, J. Lui, and R. Ma, "On incentivizing upload capacity in P2P-VoD systems: Design, analysis and evaluation," *Computer Networks*, vol. 57, no. 7, pp. 1674 – 1688, 2013.

[23] C. Zhao, J. Zhao, X. Lin, and C. Wu, "Capacity of P2P On-Demand Streaming with Simple, Robust and Decentralized Control," in *INFOCOM*, 2013.

[24] Y. Zhou, T. Z. Fu, and D. M. Chiu, "Server-assisted adaptive video replication for P2P VoD," *Signal Processing: Image Communication*, vol. 27, no. 5, pp. 484 – 495, 2012.

[25] Y. He, Z. Xiong, Y. Zhang, X. Tan, and Z. Li, "Modeling and analysis of multi-channel P2P VoD systems," *Journal of Network and Computer Applications*, vol. 35, no. 5, pp. 1568 – 1578, 2012.

[26] Z. W. Zhao, S. Samarth, and W. T. Ooi, "Modeling the effect of user interactions on mesh-based P2P VoD streaming systems," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 9, no. 2, 2013.

Delia Ciullo received the Master degree in Telecommunications Engineering and the Ph.D. degree in Electronics and Communications Engineering, both from Politecnico di Torino in 2007 and 2011, respectively. Between 2012 and 2013 she was a post-doc fellow in the MAESTRO team at INRIA Sophia Antipolis with an ERCIM fellowship. She is currently a post-doc researcher at EURECOM Sophia Antipolis.

Valentina Martina received the Master degree in Mathematical modeling in Engineering and the Ph.D. degree in Electronics and Communication Engineering, both from Politecnico di Torino in 2007 and 2011, respectively. In 2010, she has been a visiting student at the Technicolor Paris Research Lab. She is currently a post-doc researcher at Politecnico di Torino.

Michele Garetto (M'04) received the Dr.Ing. degree in Telecommunication Engineering and the Ph.D. degree in Electronic and Telecommunication Engineering, both from Politecnico di Torino, Italy, in 2000 and 2004, respectively. He is currently assistant professor at the University of Torino, Italy.

Emilio Leonardi (M'99, SM'09) is an Associate Professor at the Dipartimento di Elettronica of Politecnico di Torino. He received a Dr.Ing degree in Electronics Engineering in 1991 and a Ph.D. in Telecommunications Engineering in 1995 both from Politecnico di Torino. His research interests are in the field of performance evaluation of wireless networks, P2P systems, packet switching.

Giovanni Luca Torrisi graduated in Mathematics in 1994 at the University of Rome "La Sapienza" and obtained a Ph.D. in Mathematics at the University of Milan. Since December 2001 he has been a researcher at the CNR. His research interests are in the field of Probability Theory and Applied Probability.

Supplemental material of paper:

Peer-assisted VoD Systems: an Efficient Modeling Framework

Delia Ciullo, Valentina Martina, Michele Garetto, Emilio Leonardi, Giovanni Luca Torrisi

7 Analysis extensions

7.1 VCR functionalities

VoD systems can optionally provide to the users limited VCR functionalities. For example, Netflix allows watching users to pause or restart viewing at will. In this section we show how our model can be extended to take into account the most common VCR functionalities provided by VoD services, i.e., the possibility to place the video in pause mode, and the the possibility to replay portions of the video which have already been downloaded. The above two operations offer to the users a significant degree of flexibility, while being easily implementable in practice. Furthermore, they actually improve the effectiveness of peer-assisted video distribution, as we will see.

Observe that a VoD system can react to users making pause/replay actions in different ways. We will consider the following two extreme strategies: 1) the video download is interrupted as soon as the user enters the pause/replay state, and it is resumed only when the user starts watching the left portion of the video; 2) the download is never interrupted, and thus proceeds up to its natural end.

The first strategy makes sense while having in mind a short-term objective. Indeed, by interrupting the download of users in pause/replay, the system achieves an instantaneous reduction of global user bandwidth request. The second strategy, instead, which keeps on the video download, can be justified by having in mind a long-term objective. Indeed, users who complete sooner the download (and are likely to watch the entire video) will act as seeds for an extended period of time.

In conclusion, the first strategy responds to the need of reducing the instantaneous load, whereas the second aims at increasing the number of future seeds. While it is clear that pause/replay actions improve in general the system performance (since users remain in the system for a longer time, contributing their upload bandwidth) it is not easy, however, to guess which strategy performs better, since this depends on many system parameters, including the user behavior. An analytical model like ours can actually be very useful to predict the system performance in different scenarios.

Both strategies described above can easily be incorporated in our model. In particular, the second strategy does not require any substantial modification to the model which does not consider VCR functionalities. Indeed, users entering the pause/replay state, but keeping on the download, have no impact on (1) and the approximation developed in Section 3.2. The only difference is in the activity time T of users (and its distribution $G_T(t)$), which will be prolonged by pause/replay events, increasing the number of seeds (and reducing the system load).

The iterative procedure described in Section 3.2, instead, must be slightly modified to analyse the first strategy, in which video download is suspended while users are in pause/replay state. In this case, Proposition 1 is generalized to,

Proposition 2: Quantity $S_d(t,k)$ satisfies the following recursive equation:

$$S_d(t,k) = \begin{cases} d\,\mathbb{I}_{\{1 \notin \mathrm{P}\}} & k=1 \\ d\,\mathbb{I}_{\{k \notin \mathrm{P}\}} + \max\{0, S_d(t,k-1) - U_k\} & k>1 \end{cases}$$

where $\mathbb{I}_{\{k \notin \mathrm{P}\}}$ is the indicator function of the event $\{$user k is not pausing/replaying$\}$. Note that $\mathbb{I}_{\{k \notin \mathrm{P}\}}$ models the fact that nodes which are currently in pause are not downloading the video (so their contribution to the increase of the bandwidth request is null) while they are still contributing their upload bandwidth to the content distribution.

Then, denoting by $p_k = \mathbb{E}[\mathbb{I}_{\{k \notin \mathrm{P}\}}]$ the probability that the k-th downloading user is not pausing, we can easily extend the Gaussian approximation procedure described in Section 3.2 by simply replacing (8) and (9) with:

$$\begin{aligned} \overline{S}_d(t,k) &\simeq p_k d + \mathbb{E}[y'] \\ \sigma^2_{S_d}(t,k) &\simeq (p_k)(1-p_k)d^2 + \mathbb{E}[y'^2] - \mathbb{E}[y']^2 \end{aligned}$$

under the initial conditions: $\overline{S}_d(t,1) = dp_1$ and $\sigma^2_{S_d}(t,1) = (p_1)(1-p_1)d^2$.

7.2 Peers with limited memory capabilities

At last, we consider the situation in which the amount of memory available at a peer to store the video currently being watched is not large enough to cache the entire video file. This lack of memory clearly penalizes the effectiveness of peer-assisted video distribution, as users can redistribute only a limited portion of downloaded data.

To mitigate the impact of memory constraints, an effective solution that we propose and analyse here is to adopt a 'striping' approach: the original video stream is divided into M sub-streams, called stripes, whose size (and number) can be adapted to the storage capacity of peers. Assuming, for simplicity, that all peers have the same memory constraint (the more general case can be similarly handled), we will focus on the case in which all stripes have the same size, equal to the storage capacity of peers. Peers downloading a video need to retrieve the M stripes in parallel. Then, each peer is requested to cache and redistribute a single, randomly selected stripe belonging to the requested video.

Notice that each stripe can be regarded as a content of the same temporal duration of the original video, with an associated play-out rate equal to d_v/M. Our model can be easily extended to analyze this case as well. Indeed (1) can be generalized to compute the bandwidth requested from the servers by each individual stripe, as follows:

Proposition 3: Quantity $S_d(t,k)$, representing now the bandwidth requested by downloaders of one particular stripe, satisfies the following recursive equation:

$$s_d(t,k) = \begin{cases} \frac{d}{M} & k=1 \\ \frac{d}{M} + \max\{0, S_d(t,k-1) - U_k \mathbb{I}_{\{k \in S\}}\} & k>1 \end{cases}$$

where $\mathbb{I}_{\{k \in S\}}$ is an indicating function specifying whether the k-th user is redistributing the current stripe. By construction $\mathbb{E}[\mathbb{I}_{\{k \in S\}}] = 1/M$.

The Gaussian approximation procedure described in Section 3.2 can be easily adapted to analyze this case, using: $\mu = \overline{S}_d(t,k-1) - \overline{U}/M$ and variance $\sigma^2 = \sigma^2_{S_d}(t,k-1) + (\sigma^2_U + \overline{U}^2)/M - \overline{U}^2/M^2$. At last observe that, as consequence of the fact that every peer redistributes only one randomly selected stripe, we need to modify the mean of $S_{\text{seed}}(t)$ (the contribution of seeds associated to the considered stripe) to $\overline{U}\,\overline{N}_{\text{seed}}(t)/M$, whereas the variance of $S_{\text{seed}}(t)$ becomes $\overline{N}_{\text{seed}}(t)(\sigma^2_U + \overline{U}^2)/M$.

8 Performance under stationary conditions

8.1 Impact of upload bandwidth heterogeneity

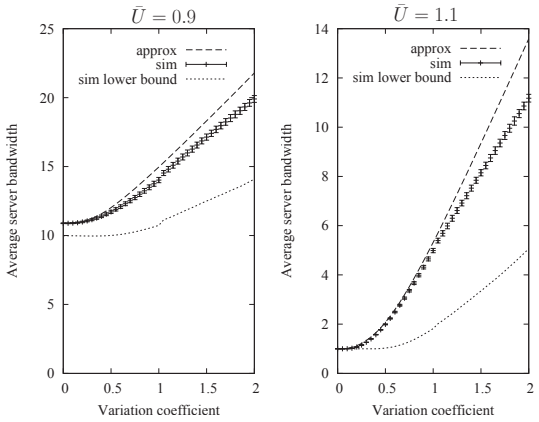

Fig. 11: Comparison of average server bandwidth with $\overline{N} = 100$, $d = d_v$, $\overline{T} = T_v$, as function of the variation coefficient of peer upload bandwidth, for $\overline{U} = 0.9$ (left plot) and for $\overline{U} = 1.1$ (right plot).

Figure 11 compares the average server bandwidth $\overline{S}$ as function of the variation coefficient of the peer upload bandwidth (keeping fixed the mean), considering $\overline{N} = 100$, $d = d_v$, $\overline{T} = T_v$. The average upload bandwidth is equal to either $\overline{U} = 0.9$ (deficit mode) or $\overline{U} = 1.1$ (surplus mode). Simulation results are supplemented by 95%-level confidence intervals. The upload bandwidth distribution used in the simulations depends on the variation coefficient: for values larger than one, we adopt a second-order hyper-exponential distribution with balanced means; for values smaller than one, we employ an exponential distribution added to a constant (this explains the small glitch at variation coefficient equal to 1).

We observe that the average server bandwidth increases significantly as the variability of upload bandwidth increases, while our approximation tends to provide a conservative prediction.

8.2 Impact of VCR functionalities

In Figure 12 we evaluate the impact of users pausing the video playback or replaying portions of video already watched. As explained in Sec. 7.1 these VCR functionalities can be easily modeled introducing the probability p_k that the k-th downloading

user is watching fresh new content (i.e., the portion of video currently being donwloaded). For simplicity, we assume that p_k is a given constant, equal for all k (more in general, we would need a sub-model to compute p_k on the basis of additional assumptions about the user behavior). The left plot of Figure 12 refers to the system that keeps on the download while the user is in pause/replay state (system 1), whereas the right plot refers to the system in which the download is suspended in pause/replay state (system 2).

Results in Figure 12 refer to the same scenario considered in Figure 5. We observe non-negligible differences between the two systems only when users act as seeds for zero or small time after finishing watching the video[4]. In this case, system 1 performs better than system 2, due to the very beneficial effect induced by just a small number of seeds, which are artificially created in system 1 by keeping on the download. We remark, however, that the gain achieved by system 1 over system 2 also depends on the probability that users watch the entire video, actually consuming all downloaded data. Premature abandons (not present in the scenario considered in Figure 12), if likely to occur, can make system 2 preferable to system 1.

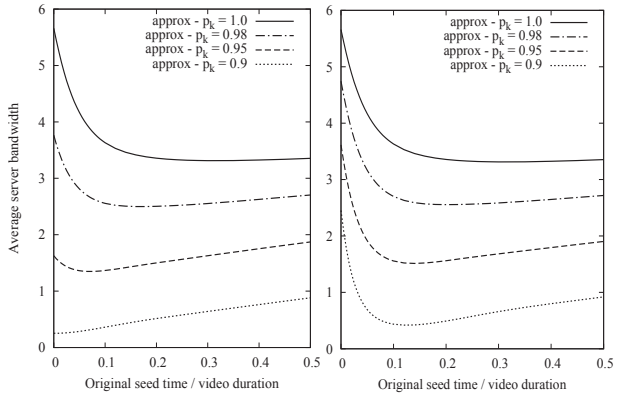

Fig. 12: Average server bandwidth as function of the (original) ratio between the seed time and the video duration, for different values of probability p_k. Comparison between system 1 (left) and system 2 (right).

8.3 Impact of limited memory at peers

At last, we investigate the impact of memory constraints at peers, as analysed in Section 7.2. We consider again the same scenario as in Figures 5 and 12. Recall that $1/M$ represents the fraction of the entire video that can be cached at a peer.

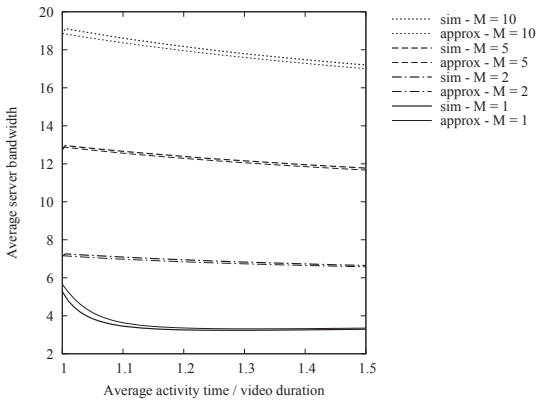

Fig. 13: Average server bandwidth as function of the ratio between average activity time and video duration, for different values of the number of stripes M. Comparison between simulation and approximate analysis.

As expected, Figure 13 confirms that limited memory availability at peers negatively affects the system performance, in rather strong way. The increase in the server bandwidth, however, is not linear with respect to M.

8.4 Summary of results under stationary conditions

By applying our performance evaluation methodology under various parameters setting, we have discovered several interesting properties of P2P-VoD systems:

- in the surplus mode, the server bandwidth achieves a maximum as we increase the number of users, and then decreases to

4. The seed time is incremented in system 1 by pause/replay functionalities: the seed time reported in Figure 12 refers to the original seed time (without pause/replay effects), due to user behavior after the video playback is finished.

zero provided that $\overline{T} > \overline{T}_d$ (*i.e.*, the average activity time is larger than the average download time) (Fig. 2);
- under the pessimistic assumption that users leave the system after watching the video, the server bandwidth can be minimized by a proper selection of the target download rate, under any system load (Fig. 3);
- the server bandwidth increases with the variation coefficient of the peer upload bandwidth distribution (Fig. 11);
- the gain achievable by non-sequential schemes over the simple sequential scheme depends critically on the size of the sliding window and the number of downloading users, and vanishes as the number of seeds increases (Fig. 4 and 5);
- VCR-like functionalities allowing users to stop/rewind the video playback improve the system performance (Fig. 12);
- memory contraints can severely penalize the effectiveness of peer-assisted video distribution (Fig. 13).

9 Summary of results under non-stationary conditions

We found out the following interesting properties:

- under non-stationary traffic conditions peer-assisted VoD systems are affected by a misalignment problem between downloaders and seeds. The per-user load γ_p is not enough to characterize the system performance, which comes to critically depend on the amount of bandwidth contributed by peers while they download the video: when $\overline{U} > d_v$ the bandwidth demanded from the servers is essentially independent from the system size (number of watching users); if $\overline{U} < d_v$, instead, the bandwidth demanded from the server scales linearly with the system size, regardless of the availability of users as seeds and the per-user system load γ_p (Fig. 8);
- in the regime where the system does not scale with the number of users, the system performance is further negatively affected by an increase in the content heterogeneity (in terms of popularity) (Fig. 10).

10 Related Work

A stochastic fluid approach to analyze peer-assisted video distribution has been proposed in [15] in the context of live streaming, in which (heterogeneous) peers download and playback content synchronously. They derive simple conditions for the existence of a fluid distribution scheme that achieves universal streaming. in a system with a given available server rate, and two classes of users (with high/low upload capacity). In [16] authors extend the analysis in [15] developing queueing network models of multi-channel P2P live streaming systems, capturing peer churn, bandwidth heterogeneity, and Zipf-like channel popularity, and showing the benefit of the View-Upload Decoupling strategy. Here we apply the stochastic fluid approach to VoD systems, whose dynamics are quite different from live streaming, since users can watch the video asynchronously.

A mathematical formulation of the server bandwidth needed under sequential delivery appeared in [4], in which authors resort to a Monte Carlo approach to get basic insights into the system behavior (like surplus and deficit modes). The sequential delivery scheme has been considered also in [11], where authors explore by simulation the effectiveness of different replication strategies to minimize the server load in the slightly surplus mode, as well as distributed replacement algorithms to achieve it. Differently from [11], we provide an analytical approximation that can account for both sequential and non-sequential delivery schemes under any system load, considering also the impact of seeds.

Stochastic fluid models for BitTorrent-like file-sharing system, accounting for the dynamics of downloaders and seeds, have been proposed for both transient and steady-state regimes [17], [18], but they are not directly applicable to streaming systems. In [19], authors adapt the fluid model in [18] to VoD systems, investigating the impact of different piece selection policies (rarest-first and in-order) on download latency and startup delay, in the case of homogeneous peers. In contrast to [19], we focus on the characterization of VoD systems with strict service guarantees and heterogeneous user upload bandwidths. In [20], a per-chunk capacity model is developed to show the tradeoff that exists between system throughput, sequentiality of downloaded data and robustness to heterogeneous network conditions. Optimal content placement strategies to maximize the upload capacity of (homogeneous) set-top-boxes (and thus minimize the servers workload) in VoD systems have been recently investigated in [21] under many-user asymptotic.

More recently, [22] characterizes the content providers' uploading cost as a function of the peers' contribution, and proposes an incentive mechanism that rewards peers based on their dedicated upload bandwidth. However, they evaluate the performances of their scheme only through simulation. Somehow complementary to our work is the study in [23], where control algorithms that achieve the optimal streaming capacity as the number of peers increases, are proposed. [24] considers a distributed and adaptive video replication strategy with some server feedback. Similarly to our paper, [24] deals with peer churn and non-stationary popularity of movies, but results are validated through simulation only. [25] proposes two analytical models that capture several aspects of peer behavior, such as participating in the system, sojourning in a channel, downloading and uploading the content, wandering around channels and leaving the system. However the effects of upload bandwidth heterogeneity, non sequential delivery and non-stationary traffic conditions are not taken into account. In [26] an analytical model to both qualitatively and quantitatively study the effect on server cost of seeks and pauses on mesh-based P2P VoD streaming systems is developed.

To the best of our knowledge, we are the first to analytically investigate the performance of peer-assisted VoD systems under non-stationary traffic conditions.

11 Discussion on model assumptions

In this section we critically revisit the main assumptions of our model discussing their impact on the analysis of a VoD system.

The first strong assumption consists in assuming a fixed video playback rate. This assumption actually does not hold in practice since most video encoding schemes produce variable bitrate streams. However, rapid bitrate fluctuations are usually averaged out by the playout buffer of the decoder, so that assuming a fixed download rate d larger than or equal to the average playback rate can be an acceptable assumption, while being also a reasonable design choice to simplify the system. Nevertheless, it would be possible to incorporate in the model a random download rate, representing fluctuations due to variable video bit-rate and cross-traffic effects. Indeed, since different users in a VoD system are retrieving at time t different and independent segments of the video content, we could well assume instantaneous play-back rates of users to be described by i.i.d. random variables with known distribution. The effect of variable play-back rates could then be easily incorporated in our modeling framework without changing its mathematical structure. Observe, indeed, that the structure of $S_d(t,k)$ in (5) remains unchanged when d is replaced with d_k, where d_k is a random variable, while the gaussian approximation in 3.2 can be easily extended to handle this case, given the first two moments of d_k.

The second important assumption of our work consists in modeling the users' upload bandwidth as i.i.d. random variables U_i with assigned distribution. We do not consider this assumption particularly restrictive, since it permits to represent pretty well bandwidth heterogeneity of users' access links, as well as random fluctuations in the available upload bandwidth due to cross traffic and other forms of bandwidth restriction. Notice that such fluctuations could be also correlated over time at each user, without affecting our analysis, which is essentially based on an instantaneous analysis of the system. The assumption that upload bandwidths are uncorrelated among users is also reasonable, since in a VoD system users are geographically spread, and thus they are likely to experience independent bandwidth fluctuations.

At last, in our work we have ignored implementation issues such as: i) the effects of protocol overheads and signalling bandwidth (necessary to reconfigure the cooperation among users); ii) possible constraints on the number of peers from which a user can simultaneously download data; iii) the effect of congestion inside the network. All these issues can potentially affect the performance of a realistic system, but we have not incorporated them for the sake of simplicity and analytical tractability.

Appendix A
Proof of Proposition 1

The case $k = 1$ is obvious. The recursive expression for $k \geq 2$ can be easily explained if we look at the users in reverse order with respect to the arrival time into the system, *i.e.*, user k arrives before user $k-1$. Suppose that we know the server bandwidth $S_d(t,k-1)$ needed in the presence of $k-1$ users. Then user k can reduce this rate by its upload bandwidth U_k, possibly bringing the server rate to zero. Instead, user k cannot be helped by any other peers, hence it requires fresh new content from the server at rate d.

Appendix B
Extension to non-sequential delivery

More formally, let $P_0(t,v) = e^{-\overline{N}_v(t)}$ be the probability that there are no users downloading segment v at time t.

Let $\overline{S}_d(t,v)$ and $\overline{S}_d^{(2)}(t,v)$ be the first and second moment of $S_d(t,v)$, respectively. To start our iterative computation, we set $\overline{S}_d(t,0) = 0$ and $\overline{S}_d^{(2)}(t,0) = 0$. Now, suppose that we know $\overline{S}_d(t,v-1)$ and $\overline{S}_d^{(2)}(t,v-1)$ for a given $v \geq 1$. Conditioning on the event $N_v(t) > 0$, we approximate $S_d(t,v-1) - \tilde{U}$ by a normal random variable y of mean

$$\mu = \overline{S}_d(t,v-1) - \frac{\overline{N}_v(t)}{1-P_0(t,v)}(\overline{U}-d) - d$$

and variance

$$\sigma^2 = \overline{S}_d^{(2)}(t,v-1) - (\overline{S}_d(t,v-1))^2 + \frac{\overline{N}_v(t)}{1-P_0(t,v)}\sigma_U^2 + \left[\frac{\overline{N}_v(t)+(\overline{N}_v(t))^2}{1-P_0(t,v)} - \left(\frac{\overline{N}_v(t)}{1-P_0(t,v)}\right)^2\right](\overline{U}-d)^2$$

and apply (6) and (7) to compute the first moment $\mathbb{E}[y']$ and second moment $\mathbb{E}[y'^2]$ of $y' = \max\{0,y\}$. Then we obtain the first two moments of $S_d(t,v)$ as:

$$\overline{S}_d(t,v) = P_0(t,v)\overline{S}_d(t,v-1) + (1-P_0(t,v))(d+\mathbb{E}[y'])$$
$$\overline{S}_d^{(2)}(t,v) = P_0(t,v)\overline{S}_d^{(2)}(t,v-1) + (1-P_0(t,v))(\mathbb{E}[y'^2] + d^2 + 2\mathbb{E}[y']d)$$

Iterating the above computation we can compute the first two moments of $S_d(t,v)$ for any v.