Peer-to-Peer Networking and Applications manuscript No.
(will be inserted by the editor)

Performance evaluation of P2P-TV diffusion algorithms under realistic settings

Paolo Veglia · Dario Rossi

the date of receipt and acceptance should be inserted later

Abstract Internet video and peer-to-peer television (P2P-TV) are attracting more and more users: chances are that P2P-TV is going to be the next Internet killer application. In recent years, valuable effort has been devoted to the problems of chunk-scheduling and overlay management in P2P-TV systems. However, many interesting P2P-TV proposals have been evaluated either in rather idealistic environments, or in the wild Internet. Our work sits in between these two antipodean approaches: our aim is to compare existing systems in a controlled way, but taking special care in realistic conditions for their evaluation at the same time.

We carry on a simulation analysis that considers several factors, modeling the L7 overlay (e.g., chunk scheduling, topology management, overlay topology, etc.), the L3 network (e.g., end-to-end latency models, fixed vs dynamic conditions, etc.), and the interaction of both layers (e.g., measurement errors, loss of signaling messages, etc.). To depict a comprenshive system view, results are expressed in terms of both user-centric and network-centric metrics.

In a nuthshell, our main finding is that P2P-TV systems are generally robust against measurement errors (e.g., propagation delay or capacity estimation), but are on the contrary deeply affected by signaling errors (e.g., loss or outdated system view), which are often overlooked without justification.

Keywords P2P-TV · Simulations · QoE

1 Introduction

Internet users habits are changing, and consequently the shape of Internet traffic is changing as well. The primate of Peer-to-Peer (P2P) file sharing, which for a long time was accounted to as constituting the bulk of Internet traffic, is now being challenged. In its Visual Networking Index [1], Cisco estimates that although P2P traffic is growing in volume, it is however declining as a percentage of overall IP traffic. Video is indicated as the primary source of this traffic shift: the amount of video traffic is currently rising faster than any other type of service, and is estimated that all form of video (TV, Video on Demand, Internet, and P2P) will account for over 90% percent of Internet traffic [1] and about 60% of Mobile traffic [2] in the next few years.

At the same time, P2P is alive and in good shape. Among most popular applications Spotify, that allows P2P streaming of musical content, generates more traffic than all Sweden [3]. Given the above trend in video growth, it is reasonable to ask, e.g., what would happen in case Spotify started offering music videoclips, or in case YouTube content was shared on a P2P platform, or when Pheon [4] the new P2P-TV projet led by BitTorrent, will be released. Besides, we point out that Adobe flash player –the de facto standard for front-end and encoding of Internet video on the Web– recently started offering P2P capabilities through the Real Time Media Flow Protocol (RTMFP) [5] and that also W3C, the standard body for the Web, is considering embedding P2P technology in future HTML standards.

Thus, exactly as users habits are changing, i.e., users want to enjoy video content without having to download it first, the P2P technology is evolving to cope with user requirement: for these reasons, applications offering TV and video over P2P technology are often indicated as the next Internet killer application. At the same time, while a number of successful and popular P2P-TV applications exists nevertheless the evolution of P2P streaming technology is not over yet. In the last years, a number of different proposals have targeted mesh-based P2P streaming [6–21]. With few exceptions [22, 23] such proposals have typically been stud-

Paolo Veglia · Dario Rossi
Telecom ParisTech, Paris, France.
E-mail: firstname.lastname@enst.fr

ied in isolation, possibly focusing on very specific aspects of the system (notably, chunk scheduling policies), in possibly highly ideal settings (e.g., overlay-only studies, homogeneous settings, synchronous timelines, perfect neighborhood knowledge, etc.). As such, a thorough comparison of the different proposals under a common and realistic framework is missing so far. A first aim of this work is thus not to propose any new system, but rather to compare existing ones. A second aim is instead to understand how the performance of these system declines under more realistic scenarios.

Building on our previous work [24], we implement some of these algorithms [17–20] in a custom event driven simulator, and evaluate their performance considering important (but often overlooked) factors, which we model with increasing levels of realism. A first issue is that "network aware" P2P-TV systems typically makes chunk scheduling and topology management decision based on some measured properties of other peers in the swarm: yet, as gathering precise and reliable measurements is notoriously difficult in the Internet, it is important to understand the implication of *measurement errors* in the system performance. A second issue is that scheduling algorithms are generally evaluated assuming a perfect, though unrealistic, knowledge of the system state (e.g., neighbors buffer maps): as such, it is important to evaluate the impact of *state inconsistency* (e.g., due to lost or outdated control information) as well.

Our main findings can be summarized as follows. On the positive side, we find that system performance are rather robust to measurement errors, as performance degrades gracefully even for very large capacity and latency measurement errors. Conversely, we find that state inconsistencies significantly degrade the achievable performance even for very low signaling error rates: as such, signaling should not be neglected in future studies aiming at a realistic assessment of the quality provided by P2P-TV services.

2 Related work

P2P-TV is a relatively long studied subject, which has been the focus of many interesting work that we overview in this section. P2P-TV studies started from seminal work on single [6–9] and multiple [10–12] trees architectures, where video GOPs (possibly encoded using multiple descriptors) are pushed from the source down the trees. To overcome the inherent limitation of tree architectures, inspired by BitTorrent, the design of latest generation P2P-TV has moved toward chunk-based diffusions architectures featuring partly meshed architectures [13–21], which we focus on in this work.

In mesh-based architectures, video chunks are either *pulled* or *pushed* on the overlay and are thus no longer received in a particular order. In a pull system, receiver peers initiate the exchange, asking some peers for the content they need: in this case, an additional round-trip delay is required for the handshake phase, as otherwise a receiver may be asking for unexisting content (i.e., causing a chunk miss). In push systems instead, the sender peer decides whom to send content to: in this case, peers need to mutually exchange content availability, as otherwise a sender may be transmitting content already available at destination (i.e., causing a chunk collision). Interestingly, under homogeneous network settings and assuming perfect knowledge of the system state, a given class of push-based scheduling algorithms (namely the *latest useful chunk* policy) has been proven [17] to achieve rate-optimality (and delay-optimality up to an additive constant term).

In [17] the peer selection strategy is however the simplest one (namely, a *random* policy), and may result in suboptimal choices in the multi-domain heterogeneous Internet environment. Hence, building over [17], many schedulers have then been proposed that also incorporate "awareness" to the network properties (such as bandwidth [19], latency [18], and their ratio [20]). At the same time, [17–21] have been studied in isolation, possibly adopting a highly idealized view of the system and of the network models. Though simplistic, this viewpoint is nevertheless necessary to gather solid theoretic foundations for specific algorithms design choices. In this work, we focus on such class of schedulers, which we analyze in a common framework under more realistic conditions. With this respect, closest work to our is [22] that, by means of simulation, however limitedly compare two systems (namely *SplitStream* [10] and *PRIME* [15]).

We point out that full blown systems [13–16] have also been evaluated by means of middle scale deployments of real prototypes. With the exception of [23] (that compares *Chainsaw* [25] and *SplitStream* [10] and is the closest work in spirit to ours), and of [26] (that analyzes *PPlive*, *SopCast*, and *TVAnts*), real system have however been studied in isolation. Moreover, what makes the comparison difficult is that experimental conditions are hardly reproducible. Also, although performance results are in this case realistic, as systems are very tightly designed, it is often not possible to isolate and understand the impact of different factors in the overall system performance.

While a great deal of literature focus on the study of individual P2P-TV applications, only few work tackle the *comparison* of such system: namely, an analytical approach is followed in [17], while [22] and [23] respectively follow a simulative and experimental methodology. As in [17,22,23], this work compares the performance of a several network aware systems, that we represent by different peer selection and chunk scheduling policies. Our aim is to perform a sensitivity analysis of the impact of different settings on the performance of medium to large scale system. With this goal in mind, we resort to a *simulation-based* approach, that offers both (i) complete control over the investigation environment (with a greater level of details with respect to what analytical techniques allow) and (ii) the ability to investigate larger swarm sizes (with respect to those achievable with an experimental methodology). In the following, we report results for swarms comprising more than 50,000 peers, a scale that is hardly achievable in real-world experiments. Notice indeed that, as pointed out in [27], PlanetLab experiments are usually limited to few lightly loaded and reliable nodes (direclty quoting [27], authors limit " the overlay size to 160 peers running on machines that report at least 5% idle CPU time"). Conversely, if recent experimental work [28] achieved swarms of 10,000 peers on dedicated experimental facilities such as Grid'5000 (running 100 peers per machine), the testbed environment however looses the realism of the Internet (and may bring other elements out of experimental control, as pointed out by the same authors in [29]).

Otherwise stated, this work aims at performing a thorough and realistic, but controlled and reproducible comparison of relevant systems proposed in the literature [17–20]. In order to provide a fair comparison, we consider several algorithms that perform well in ideal settings, and implement them in a common simulation framework. We then challenge these algorithms by plugging different models, representative of a realistic Internet environment, so to assess their performance into the wild. This paper builds on our previous work [24], extending it by (i) studying the impact of different underlay network models, (ii) showing the benefits of a continuous topology management task, (iii) introducing errors in the latency and capacity measurement processes and (iv) thoroughly justifying our scenario setting with dedicated experiments.

3 Framework Description

This section overviews the framework we devised to compare P2P-TV systems, which is available as open-source software at [30]. The custom chunk-level event-based simulator takes into account several components, which are visually presented in Fig. 1. From an high-level point of view, the framework consists of two layers: namely, the underlaying physical L3 network and the logical L7 overlay, which are coupled by different models of their possible interactions.

From the L3 point of view, at the edge of the architecture we have end hosts, which are physically interconnected to the L3 network by access links, that acts as bottleneck, and that are modeled as a capacity–delay pair. Hosts are attached to edge routers, which constitute the entry point of P2P-TV traffic in the network, which we model with increasing levels of details. From the L7 viewpoint, hosts run P2P-TV applications, which we express in terms of the algorithms (e.g., chunk scheduling, peer selection, topology management) they implement, and of the overlay graph resulting by those algorithms. Finally, we model L7/L3 interaction by taking into account that, in the real world, different sources of error can slip in at any point of the process (e.g., loss of signaling packets, bias on measurement of L3 properties performed by L7 overlay peers, etc.).

Table 1 Breakdown of hosts into classes

Class	Ratio	BW$_D$	BW$_U$	t_{TX}
I	10%	∞	5.0 Mbps	20 ms
II	40%	∞	1.0 Mbps	100 ms
III	40%	∞	0.5 Mbps	200 ms
IV	10%	∞	0 Mbps	∞

In the remainder of this section, we further detail each component, motivating the realism and soundness of our choices. At the same time, we point out that the framework is extremely flexible, and can easily accommodate other models for the different components as well: as such, where relevant, we list other interesting models that could be investigated by further research but that are out of the scope of this work.

3.1 L3 Components

With L3 components we indicate objects in the physical world, such as (i) hosts and (ii) routers, that are interconnected by a (iii) network.

3.1.1 Host

Hosts are machines running P2P applications instances, and are characterized by a physical interface to the L3 network. Hosts are divided in different classes according to their upload bandwidth BW$_U$, while we consider the download bandwidth BW$_D$ to be infinite. This is a reasonable assumption in case of asymmetric access, provided that we further assume that the bottleneck is placed at the edge of the network (which represents the common case today and is generally assumed by other research on P2P-TV [19, 20] and P2P-filesharing [31]).

In our simulations we consider up to $N_H \leq 512000$ hosts divided into four classes, where the average BW$_U(i)$ for the i-th class is allocated as described in Tab. 1, consistently with [19, 31] (we further motivate this choice in the Appendix). The first column of Tab. 1 reports the class breakdown: the bulk of peer population is constituted by mid-speed peers, with a non marginal presence of very-high and very-low speed peers. In class i, the uplink capacity of each peer p is set to $\nu \cdot \mathrm{BW}_U(i)$ where ν is a random variable uniformly distributed in $[0.9, 1.1]$ (i.e., the actual uplink of each peer deviates at most 10% from the average for that class). For reference purpose, last column reports the transmission time t_{TX} of a 12.5 KBytes chunk (corresponding to about 10 full-payload packets, considering application layer header), where we consider that all the uplink bandwidth is

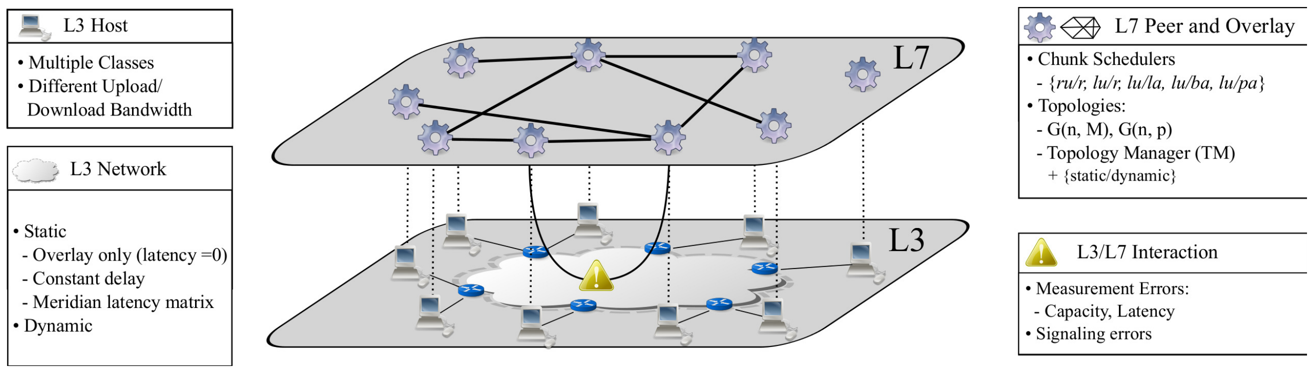


Fig. 1 Sketch of the evaluation framework: overview of L3 and L7 components under study, including L3/L7 interaction

devoted to the chunk transmission. Since we are interested in P2P-TV steady state performance, and since host population is generally stationary during the show (contrarily to what happens, e.g., in P2P filesharing systems), we neglect churn in this study (a longer discussion motivating this choice is deferred to Appendix A.1).

3.1.2 Router

Each host is single-homed, i.e., attached to a single access router, which models the first IP router in the aggregation network (e.g., the BRAS for ADSL access). In our simulations, we consider a number of routers equal to $N_R = 100$ and we use a simple host-to-router mapping policy: each host is bound randomly to a router, so that in average $N_H/N_R =$ 20 hosts are attached per router.

As depicted in Fig. 1, routers are placed at the edge of the network and act as access points forming a logical full mesh at L3. Each router keeps statistics about packets passing through its interfaces and discriminates traffic between *remote* (i.e., the traffic that it injects further down toward the core) and *local* (i.e., the traffic that is reflected toward other access links insisting on the same router).

Notice that, in this way, routers directly yield a very simple measure of traffic locality as $P\% = local/(local + remote)$, which is independent from the actual network topology, from the Autonomous System AS level topology, from the router-to-AS mapping policy, etc. We point out that the *absolute* value of this measure is heavily affected by several factors (e.g., AS topology, host-to-router mapping), which disqualify this index to be used for realistic assessment of locality awareness. At the same time, this rough indication however allows to *relatively* compare the locality awareness of P2P-TV systems, which is our main aim more that to evaluate the amount of inter-AS traffic (for which we refer the reader to [18, 32]).

3.1.3 Network

The L3 network models the interconnection of routers: in this work, we consider different models of network, with additional complexity and levels of details.

If we consider the access link to be the bottleneck, likely no queuing happens within the network core: as such, the network simply models the delay of the end-to-end path. In this case, the network topology is well represented by a *static* end-to-end latency matrix, where the latency essentially represents the propagation delay along links of the end-to-end path. We consider different models of static networks, from an ideal overlay model (where the end-to-end delay is given solely by the chunk transmission duration over the uplink bottleneck) to more realistic models such as Meridian [33] (where end-to-end delays are derived from real measurement performed among a large number of Internet hosts).

We also consider the case where congestion may still happen in the network by employing *dynamic* end-to-end latency matrices, where the latency between any two peers may thus differ from chunk to chunk, due to variation in the queuing delay and to background traffic. We point out that the case where the amount of P2P-TV traffic is (i) minority or (ii) prevalent shall be considered separately. In the former case, which is typical today and that we consider in this work, congestion is due to the background traffic: we model this effect by simply varying the latency between two consecutive chunks at random. In the latter case network links should be modeled as well, so that Traffic Engineering mechanisms (e.g., load balancing, periodic optimization of routing weights IGP-WO, etc.) could be applied to handle the edge-to-edge traffic matrix induced by P2P-TV traffic. As for the time being the case (i) is the most common, we consider the case in which P2P-TV traffic is prevalent to be out of scope[1].

[1] Still, we point out that we are currently investigating the case (ii), using however an orthogonal technique: real P2P applications deployed over a network *emulator* featuring Traffic Engineering capabilities [34]

3.2 L7 Components

With L7 components, we indicate higher level components, such as (i) peers, which are instances of L7 P2P-TV applications running on L3 hosts. In more detail, we model peers by defining the algorithms they implement: specifically, each peer has to (ii) manage the overlay topology and (iii) schedule the transmission of chunks on the overlay links.

3.2.1 Peer

Each peer establishes and maintains several logical connections to other peers in the overlay: we denote with $N(p)$ the set of peers in the neighborhood of p. As in mesh-push systems chunks are not received in playout order, peers need to have a buffer-map B that describes the chunks received and stored into the peer memory. Given a peer p, we indicate with $B(p)$ its buffer-map, and denote by $c \in B(p)$ the fact that peer p has received chunk c. The size of the buffer map $B(p)$ determines P2P-TV performance as in the following tradeoff: large buffer maps reduce the chunk loss probability, but increase the time lag with respect to the source chunk generation time; conversely, small buffer maps reduce the playout delay with respect to the source at the price of an increased chunk loss probability (as chunks that arrive later than the playout delay are no longer useful and thus can be considered as lost).

In order to gather performance of the system in *steady state*, in this paper we do not consider churn (i.e., peers arrival or departure). While this choice may seem strange at first sight, especially given our attention to the realism of the scenario, we nevertheless show its soundness in Appendix A.1. Shortly, results from a measurement campaign in real ISP network show that, while churn in filesharing applications is dominated by user habits, the churn in livestreaming applications is dominated by the content palimpsest: in other words, users connect to watch specific content at a specific time, and stay connected during the whole program.

3.2.2 Overlay Topology

Logical links established by peers form an overlay topology. To enhance their performance, peers may perform Topology Management (TM): i.e., they rearrange their overlay neighborhood in order to exploit population heterogeneity, so to globally optimize the topology based on local decisions.

In this work, we focus on topology management by considering it as either (i) a black box tool that induces a particular type of overlay graph or (ii) a specific algorithm that continuously adjusts the topology. In more details, for (i) we consider different Erdos-Renyi $G(n, M)$ and $G(n, p)$ topologies that are created at $t = 0$ and are never changed later on, and that thus define a fixed logical neighborhood for all peers at time $t = 0$. For (ii) we additionally consider a topology management process that continuously run and adapts the initial topologies, based on the measured peer properties (e.g., latency for geolocalization or capacity for performance). Indeed, since higher capacity peers can serve more neighbors, placing them near the source allows spreading new chunks faster and to a greater number of nodes. This should turn out in a per-chunk diffusion trees (i.e., the instantaneous trees followed by each chunk, which differ from chunk to chunk) with higher fan-out and reduced depth.

We point out this to be a reasonable approach: indeed, considering a $G(n, M)$ or $G(n, p)$ topology at time $t = 0$, roughly models a system in which peers joining the system receive a small number of bootstrap peers (e.g., by means of a BitTorrent-like tracker) that constitute their initial neighborhood, which may be then continuously adjusted by the TM (e.g., by means of a BitTorrent-like peer exchange PEX function[2]). Similarly to BitTorrent, TM process is continuously run with a timescale on the order of a few seconds, so that the TM process happens a few tens of times per simulation run. We point out that according to [26], applications in the Internet may exhibit behavior closer to case (i) such as TVAnts and Joost, or to (ii) such as PPLive and SopCast, which makes both cases relevant. We also point out that despite other graphs could be considered for (i), such as Barabasi-Albert [35] scale-free and Watts-Strogatz [36] small-world, this would not however add further realism to our simulation campaign – as we will see, the topology dynamics are far more important than the initial conditions at time $t = 0$.

3.2.3 Chunk Scheduler

The ultimate goal of any P2P-TV system is to give each peer a continuous stream of data: as such, peers must avoid having gaps in the buffer-map positions that are closer to the playout deadline. The video exchange process is handled by a chunk scheduler, which acts whenever a peer can use the host upload bandwidth. In push systems, any peer p runs a scheduler that has to choose: (i) a chunk from its buffer map $B(p)$ and (ii) a destination peer among its neighbors $N(p)$.

Scheduling algorithms can be divided in two classes depending on the order in which the chunk/peer selection is made: in this work, we focus on algorithms that first chooses the chunk to send and then the destination peer. We consider the chunk scheduling algorithms proposed in [17–20] which we summarize in Tab. 2. Loosely following [17], we denote each algorithm as *c/p* where *c* and *p* stand for *chunk* and *peer* selection algorithm respectively.

The simplest scheduler is the work-conserving ru/r, that selects a *random* chunk $c \in B(p)$ which is sent to a random *useful* peer $p' \in N(p)$, i.e., a peer that misses that chunk $c \notin$

[2] http://www.rasterbar.com/products/libtorrent/extension_protocol.html

Table 2 Chunk scheduler policies

Scheduler	Description		
ru/r [17]	Random useful chunk	/	Random peer
lu/r [17]	Latest useful chunk	/	Random peer
lu/la [18]	Latest useful chunk	/	Latency-aware peer
lu/ba [19]	Latest useful chunk	/	Bandwidth-aware peer
lu/pa [20]	Latest useful chunk	/	Power-aware peer

$B(p')$. We then consider a series of schedulers that select the *latest* chunk in their buffer-map, which then they send to a useful peer selected according to either a lu/r random strategy [17] or a *network-aware* criterion $lu/\{la, ba, pa\}$. As far as network-aware strategies are concerned, we consider a latency-aware lu/la strategy adapting [18] from file sharing to P2P-TV applications, a bandwidth-aware lu/ba strategy [19], and a power-aware lu/pa strategy [20] (i.e., where power is the ratio of bandwidth to latency B/L). Selection is performed by measuring the property of each peer, that are then ranked according to the property value (e.g., low latency, high bandwidth or power) and selected *probabilistically* (i.e., not in strict order), with a probability that decreases with increasing ranking.

Intuitively, lu/r aims at keeping the playout delay from the source as low as possible by diffusing the most recent chunk at their disposal, i.e., the latest in their buffermap $B(p)$. We consider the simple ru/r for reference purposes, and lu/r as it is proven to be optimal in ideal homogeneous settings [17]. Network-aware $lu/\{la, ba, pa\}$ schedulers [18–20] are instead expected to enhance performance beyond lu/r, especially in case of heterogeneous realistic scenarios: in more details, lu/la aims at locally confining the traffic by proximity peer selection, lu/ba aims at reducing the chunk diffusion time by preferring peers with higher upload capacities and lu/pa aims at combining both benefits.

3.3 L3/L7 Interaction

Finally, the efficiency of scheduling decisions is possibly perturbed by errors affecting (i) the precision of network property measurements or (ii) the fate of control information exchanged by peers, that have largely been neglected in the reference work we consider [17–20]

On the one hand, (i) network-aware schedulers base their chunk-scheduling decision on properties concerning neighbors and possibly the underlying network conditions (e.g. path RTT, available bandwidth, peer upload capacity BW_U, etc.). Such properties can either be retrieved through an "oracle" entity (such as an IETF ALTO [37] compliant server in an ISPs), or directly measured by peers themselves. Direct measurement can be rather imprecise for several reasons (e.g. cross traffic, OS scheduling, NICs interrupt coalescing, unexpected interaction between simultaneous measurement probes, etc.), which can in turn lead to unfaithful neighborhood representation and wrong scheduling decisions. In order to assess the impact of measurement errors without being bound to specific measurement techniques, we resort to a high-level model where the measurement process is controlled by a single parameter α describing the magnitudo of the error.

On the other hand, (ii) control information can be not timely disseminated, or event lost, at L3. Indeed, in case of gossiping algorithms using UDP, such information would not be retransmitted, distorting thus the vision that each peer has of the system state. Inconsistency can also be due to slow dissemination of control information (e.g., a system may wish to limit the amount of signaling traffic injected at L3 by reducing the refresh rate of control information exchange). Considering mesh-push P2P-TV systems, both types of errors translate into out-of-date knowledge concerning neighbors' buffer maps: in this case, a peer may decide to schedule the transmission of a chunk even if the destination has already received that chunk, resulting in an unnecessary chunk transmission (i.e., a chunk collision). In order to assess the impact of signaling without being bound to specific algorithms (nor to their settings), we resort to a high-level abstraction, and model errors due to packet loss or out-of-date system knowledge as error on the buffer-maps.

4 Simulation Results: Impact of L7 and L3

In this section, we study the impact of L7 and L3 factors on the system performance: we first analyze the impact of chunk scheduler and topology manager, and then evaluate the impact of L3 topologies on the system performance.

Simulations have been performed according to the following general settings. For each parameter under investigation, simulations are averaged over 6 repetitions: specifically, we consider 3 different instances of 2 different overlay graphs[3], i.e., one $G(n, M)$ and one $G(n, p)$. Unless otherwise stated, we use the Meridian dataset [33, 38] as a default realistic model of L3 network latencies with $\overline{M} = 35$ ms.

By default we consider overlays consisting of $N_H = 2000$ peers, of which we simulate a lifetime of 150 seconds, during which 1500 chunks of video stream are disseminated in the overlay. We consider a single source node that streams video at an average rate of 1 Mbps, and consider 100 kbit fixed-size chunks (i.e., 10 new chunks are generated in each second). Statistics are collected starting from 500th chunk in order to avoid the initial transient. We consider that buffer maps store 50 chunks, which correspond to a playout delay of 5 seconds.

[3] For lack of space, we do not breakdown results according to the overlay topology at time $t = 0$, but we can assert that its impact to be modest.

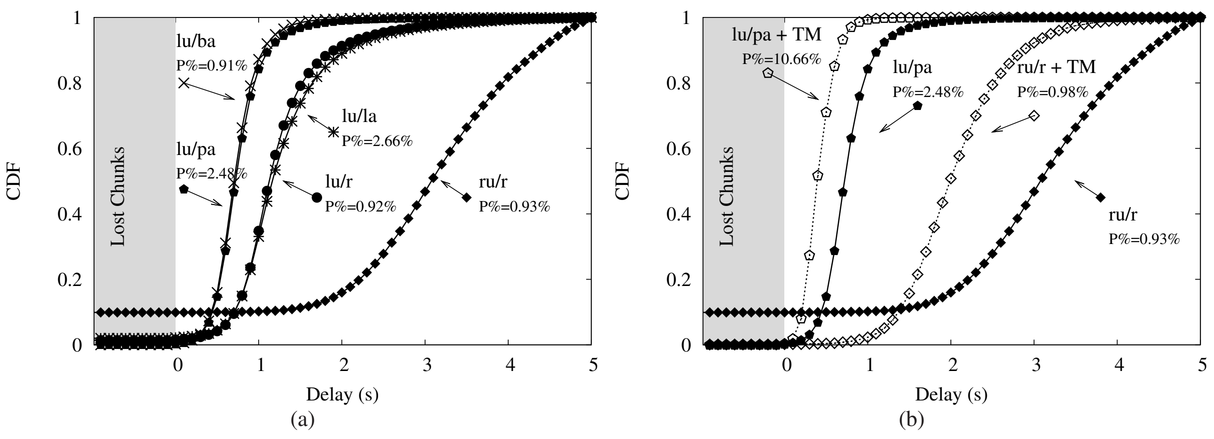


Fig. 2 Cumulative distribution function of chunk delay: (a) Performance of different schedulers (ru/r and $lu/\{r, la, ba, pa\}$) on a Meridian network. (b) Effects of topology management for best (lu/pa) and worst (ru/r) schedulers, with (TM) and without topology management features. Labels along the curves in (a) and (b) express the traffic proximity percentage $P\%$.

4.1 L7 Overlay

4.1.1 Schedulers

Curves in Fig. 2(a) show the cumulative distribution function (CDF) of chunk delays perceived by each peer (i.e., the temporal interval elapsed from the generation of the chunk at the source and the arrival to a given peer). Each curve represents a different scheduler, and we indicate lost chunks (i.e., chunks that arrived later than the playout deadline) as chunk with negative delay (i.e., falling into the gray shaded zone): this is especially visible for the simplest scheduler ru/r, where the fraction of lost chunks exceeds 10%. The picture further reports the traffic-locality $P\%$ percentage along each curve. Recall that $P\%$ represents the fraction of chunks that do not traverse the core network (i.e., the destination host is attached to the same router of the sender host), and is thus a rough indication of network friendliness.

With the exception of ru/r, other schedulers limit the fraction of lost chunk (which is very close to 0%), but instead differ by chunk delay and locality $P\%$ measures. Considering lu/r and lu/la, both strategies select the latest chunk and send it to peers which do not own it: lu/r selects a destination peer at random, while lu/la proportionally prefers closer neighbors. Clearly, locality improves when latency-aware lu/la peer selection is performed with respect to lu/r. At the same time, notice that lu/r and lu/la are very close in terms of delay, despite lu/la preference for low latency neighbors. This can be explained with the fact that the propagation delay has a less prominent impact with respect to transmission delay, especially considering that chunks possibly travel multiple hops on low-capacity access links.

Consider indeed that the average propagation delay between any two peers is $\overline{M} = 35$ ms, whereas from Tab. 1 we have that the average chunk upload time ranges from 20 ms for class-I peers to 200 ms for class-III peers. This entails that, at each hop, the transmission time likely plays the most important role in determining the chunk delay performance: thus, merely choosing a peer which is closer in terms of the propagation delay does not allow to improve the overall system chunk delay performance.

Finally, the lu/ba and lu/pa schedulers achieve the best delay performance. Consider that both lu/ba and lu/pa assign scores according to the destination upload bandwidth, with the power-aware lu/pa scheme taking into account the propagation latency as well. Results confirm that uploading chunks to high-capacity peers, which can in turn diffuse them fast, is beneficial to the whole system [19]. Moreover, we further gather confirmation of the fact that explicitly taking into account node latency improves locality $P\%$ but does not further ameliorate delay performance.

Overall, preference toward high-bandwidth peers is necessary to reduce the delay incurred by chunks; instead, preference toward low-latency peers is not helpful in reducing the chunk delay, but may ameliorate the network friendliness confining the traffic at the access.

4.1.2 Topology management (TM)

We now investigate the impact of topology management (TM) on the system performance. To gather performance bounds for a large class of schedulers, we consider lu/pa as *upper-bound* (since it exhibits the best results in terms of both delays and locality) and the simple ru/r as *lower-bound* (notice that in case peers do not have any accurate signaling, bandwidth and latency information, lu/pa would degenerate into ru/r).

In their overlay maintenance process, peers have the chance to tune their neighborhood, both in terms of its *size* (i.e., change their out-degree) and *composition* (i.e., preferring

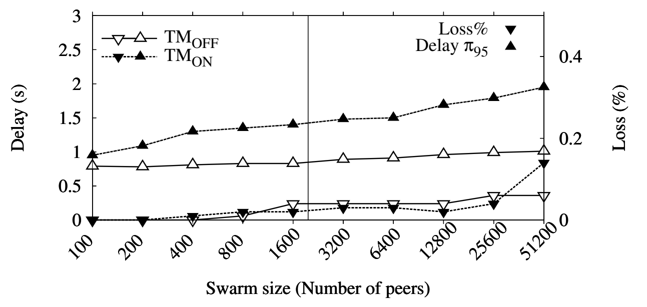


Fig. 3 Impact of swarm size on P2P-TV performance, measured in terms of the chunk loss rate (*Loss%*) and 95th delay percentile π_{95}. The picture limitedly reports performance of a network aware lu/pa scheduler, over a Meridian L3 network, with and without topology management TM.

high-bandwidth peer as in BitTorrent tit-for-tat, or trying new peer as in BitTorrent unchocking). For the sake of simplicity, we consider topology management as a feature that can be turned on or off, and select thus a single algorithm: specifically, we use the approach described in [39], which we refer the reader to, for a detailed description. Briefly, according to [39] peers continuously vary their neighbor size, selecting peers according to a "desirability" function that depends on the neighbor upload bandwidth: as we previously observed in the case of chunk scheduling, bandwidth-awareness is extremely beneficial (specifically, more beneficial than latency or power-awareness) to the overall system performance, hence our selection. Notice that the TM algorithm is run in the background by each peer, so that the overlay topology is continuously rearranged. Specifically, we select TM parameters that were shown to yield to good performance in [39], so that the topology is updated every 5 seconds – which is about twice faster than the BitTorrent unchoking policy, and of the same order of magnitude of the P2P-TV playout deadline. Since each simulation lasts for 150 seconds, TM has the chance to rearrange the topology 30 times per run.

In Fig. 2(b), improvements induced by the topology manager are clearly noticeable for both schedulers. Taking into account ru/r, for instance, we notice a significant amelioration in terms of both losses and delays. According to [39], high-bandwidth peers have an higher fan-out, and tend to select high-bandwidth nodes in their neighborhood: in this way, chunks generated by the source will be first sent to nodes that are capable of spreading them faster, thus reducing the overall delay and, by consequence, loss rates. In other words, performance enhancements are due to the fact that high-capacity nodes "move" up toward the source in the instantaneous chunk diffusion tree. In Fig. 2(b), the same scheduler ru/r can lower the mean delay by 0.75 seconds with an improvement of 25% and reduce losses to 0.25% by performing topology management.

Yet, improvements can be achieved in case of lu/pa as well: e.g., the 99th percentile of the delay reduces by 50% reaching 1.1 second. Moreover, notice that several beneficial effects combine altogether: indeed, despite being based on bandwidth-awareness only, TM topology management process consistently increases the fraction of the traffic confined in the access network as well ($P\% = 10.7\%$). Indeed, consider that the improved neighborhood is composed mostly by high-bandwidth nodes: the power-aware scheduler is then able to select among closer nodes, that are also higher capacity than in the previous case. Overall, this yields higher odds to choose high-capacity nodes that are also connected to the same router, and as high-capacity peers can offer a larger amount of data in the same time window, traffic locality increases as well.

Summarizing, active topology management is beneficial, as it increases the chances to find higher capacity peers, thus lowering the chunk delay and hence reducing losses.

4.1.3 Swarm size

Finally, we perform a simulation campaign to assess the impact of the swarm size on the achievable performance. We limitedly consider the lu/pa scheduler, that yields to the best results in terms of playout delay and traffic proximity.

Fig. 3 shows system performance as a function of the swarm size, in logscale, varying from 100 to 51,200 peers. Specifically, the picture reports the 95-th percentile of the delay distribution π_{95} (left y-axis) and the loss rate (right x-axis), for both topology management TM settings (i.e., on/off).

It can be seen that both delay and losses increase *sub-linearly* with the swarm size – notice indeed that the linear delay and loss slopes are achieved for a *logarithmic* x-scale. Notice also that topology management is highly effective in limiting the chunk delay and losses even for large swarms.

Notice further that $N_H = 2000$ is roughly at the center-fold of the explored swarm size, where roughly average performance is achieved. Thus, in the remainder of the paper we consider the more manageable swarm size of $N_H = 2000$ hosts: in turn, this allows us to run an important number of simulations, so to perform a thorough sensitivity analysis of several system parameters. Notice that, since this swarm size corresponds to average delay and loss, it should also be easy to interpolate the gathered results for other settings, such as less (more) popular TV channels, that simply map to smaller (bigger) swarm sizes.

4.2 L3 Network

We now assess the impact of the following models for the underlaying L3 network, each of which assigns latencies between access routers in a specific way:

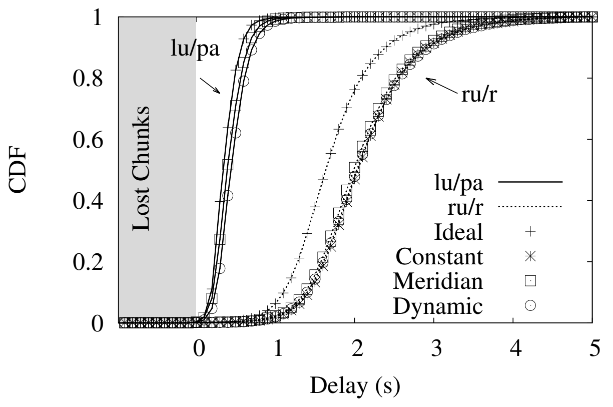


Fig. 4 Cumulative distribution function of chunk delay: Impact of different models of L3 networks, for best (lu/pa) and worst (ru/r) scheduler.

- **Ideal**: This represents an L3 network without latencies, or in other words, propagation delay is 0, so that in practice only the logical L7 overlay topology is taken into account.
- **Meridian**: Latencies provided by the Meridian project [33, 38], where realistic latencies are gathered by means of end-to-end Internet measurements (the mean latency of the Meridian dataset equals to $\overline{M} = 35$ ms as most of peers in the data-set are from US).
- **Constant**: Latencies are constant and equal among all end-to-end paths, and the latency value is equal to the mean value $\overline{M}$ of Meridian latencies.
- **Dynamic**: Latencies between any two pairs of routers are distributed according to a Negative Exponential distribution, whose mean is fitted using the mean value $\overline{M}$ of Meridian latencies; to simulate the effect of cross traffic, yielding to different levels of congestion on a chunk duration timescale, new values of latency are extracted for each new chunk propagation.

The network models we consider range from a simplistic *Ideal* model to the realistic *Meridian* one, where one would expect network-aware algorithms to stand from the lot. The *Constant* network model is a simple, still unrealistic model, that is however fitted on real data: notice that we expect latency-aware algorithms to be ineffective in this case. Finally, we include the *Dynamic* network model as a worst case for latency-aware algorithms such as lu/pa, since the decisions are taken on the basis of measurements that however continuously change (so that each chunk between any two peers will *always* experience different latencies).

In Fig. 4, we show the CDF of delays for both ru/r and lu/pa schedulers using different topologies, when the topology management feature is enabled (since similar considerations hold, we avoid reporting the case where topology management is disabled). From Fig. 4 we notice that the performance of each scheduler remains clearly separated, and further that the L3 network model only minimally affects the chunk delay performance (i.e., given any scheduler, curves are tightly clustered across all models).

In case of ru/r scheduling, ideal L3 network exhibits remarkably lower delays than the other network models, which all have a very similar impact on the delay performance. In case of lu/pa scheduling, we instead remark that performance figures are very tight irrespectively of the network model, which surprisingly holds even for the dynamic network scenario. Again, this is due to the predominant impact of transmission delay over propagation delay, entailing that even a rather simple network model (e.g., non-null delay fitted over real measurement), yields consistent performance estimates.

Notice that this might change in case the transmission and propagation delay are comparable, which can happen when e.g., the uplink bandwidth increases, or the chunk size shrinks. Still, as small chunk implies higher overhead (i.e., due to increased signaling rate, buffer map size, etc.) we can expect the relationship between transmission and propagation delay to hold for a while. Notice also that, although lu/pa performance are not affected by the network model given the current peer population breakdown, in future scenarios where most of nodes have high-capacity, we instead forecast an increased importance of latency-awareness over bandwidth-awareness. In this case, we may also expect the dynamic network case to constitute a stiffer scenario (see Appendix for further discussion).

Summarizing, the impact of L3 network models is almost negligible: in case access network is the bottleneck, the chunk transmission time largely dominates the end-to-end delay (at least for reasonable chunk sizes and the current access rates).

5 Simulation Results: L3/L7 Interaction

In this section, we investigate the impact of L7/L3 interaction on the system performance: first we study latency and capacity estimation errors, afterwards we analyze the effect of signaling errors.

5.1 Measurements Errors

P2P-TV systems need to implement measurement techniques in order to successfully implement both topology management policies and network-aware scheduling algorithms such as $lu/\{la, ba, pa\}$. In a real deployment, both latency-related (e.g., one-way delay or RTT-latency) and capacity-related (e.g., bottleneck capacity, available bandwidth) measurements will be affected by some degree of errors, that are either intrinsic to the measurement techniques, or depend on tem-

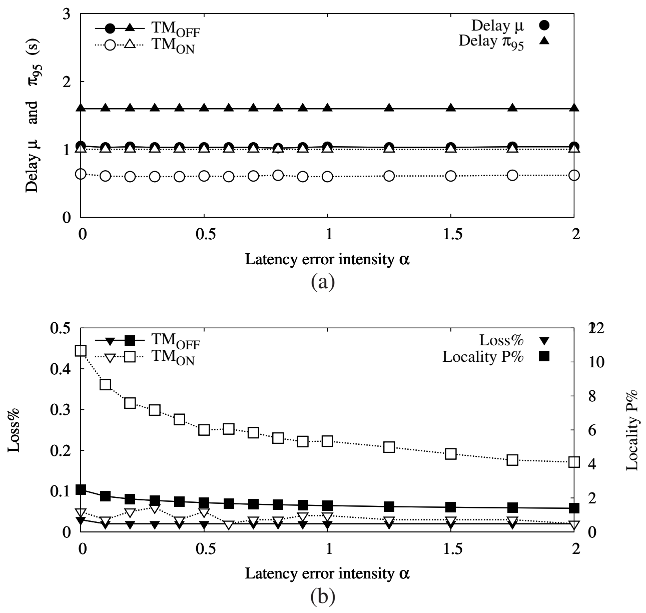


Fig. 5 L3/L7 Interaction: Impact of latency measurement error. (a) reports chunk delay mean μ and 95-th percentile π_{95}, (b) reports $Loss\%$ and locality $P\%$. Both topology management TM settings are used.

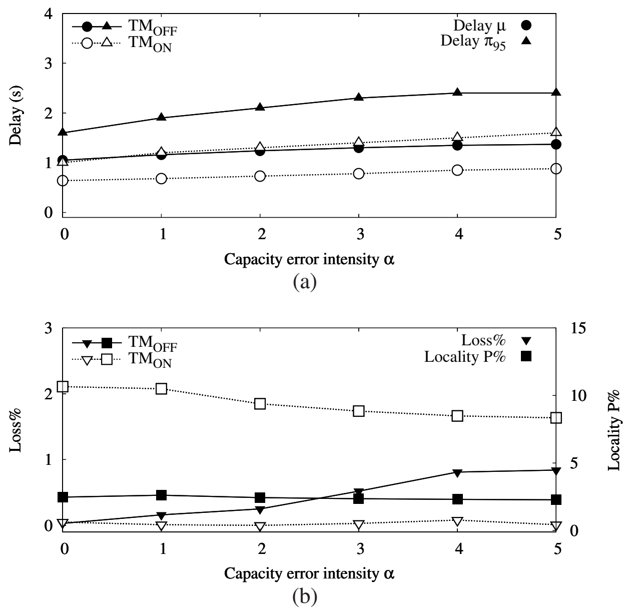


Fig. 6 L3/L7 Interaction: Impact of capacity measurement error. (a) reports chunk delay mean μ and 95-th percentile π_{95}, (b) reports $Loss\%$ and locality $P\%$. Both topology management TM settings are used.

porary network conditions: our aim is thus to evaluate the robustness of P2P-TV systems to such errors.

5.1.1 Latency measurement errors

Consider latency first, which is generally simple to estimate, and focus on the minimum RTT estimation, which is especially simple since it does not need clock-synchronizaton. In case RTT measurements can passively exploit the continuous transmission of data/acknowledgment pairs, this yields to many samples and thus to robust estimates. However, there are cases where RTT measurements are needed *prior* that any data transmission happened: in this case, active measurement are needed which yields to fewer samples and thus to possibly bias the RTT estimation. Specifically, we consider that RTT can only be *over-estimated* (e.g., since the acknowledgment packet may be delayed due to cross-traffic, self-induced congestion at the access, sustained CPU load in the host machine running the P2P application, etc.), so that a nearby peer can be mistaken as a faraway one. Formally, denote by $\varDelta(p,p')$ the actual round trip time latency between p and p', and consider that peer p will measure a $\tilde{\varDelta}(p,p') = (1+\alpha)\varDelta(p,p')$, where α represent the error intensity and is modeled by a random variable, that follows a negative exponential distribution with mean $\overline{M}\alpha$.

System performance as a function of increasing error intensity α are reported in Fig. 5. Specifically, x-axis reports the percentage of overestimation error between the measured and the actual latency value, varying from $\alpha = 0$ to $\alpha = 2$ (measured $\tilde{\varDelta}$ is, on average, the double of the actual $\varDelta$). Pictures report both user-centric (i.e., mean μ and 95th percentile π_{95} of the delay, Loss %) and network-centric metrics (i.e., percentage of local traffic $P\%$).

Overall, it can be seen that precision of latency estimation has a small impact on P2P-TV performance, limitedly affecting the overlay ability to localize the traffic. While this may be surprising at first, recall that due to ADSL-like capacities, the relative duration of chunk transmission time is larger than the propagation delay. Therefore, since delay toward the (erroneously) selected peer still complies to the Meridian latency matrix, this mismatched peer selection cannot translate into higher playout delays (as chunk transmission duration prevails), nor losses (as the playout buffer equals 5 seconds, and is thus about two orders of magnitude longer than the average delay of the Meridian matrix).

Conversely, latency measurement errors yield to select peers with mismatched delay: as the scheduler knowledge of its neighborhood becomes erroneous, proximity-aware choices are no longer correct, and the traffic locality index significantly decays. Hence, latency measurement errors can only impact the P% metric, while the delay (μ, π_{95}) and loss statistics are practically unaffected. Notice also that a qualitatively similar behavior holds both in absence or presence of topology management (though in the latter case the absolute amount of locality P% loss is more important).

5.1.2 Capacity measurement errors

Let us now consider capacity measurement. In this case, we expect measurement errors to have a possibly larger impact on the system performance, since the transmission delay component (which depends on the uplink capacity) plays a major role in determining the chunk delay. At the same time, capacity measurement are notoriously difficult, as several techniques typically yield rather different measurement [40]. Also, unlike the previous case, capacity can be either under-estimated or over-estimated (depending on the considered technique, due to cross traffic, etc.). In the case of P2P, this gets further complicated as concurrent measurements have mutual influence [41], which further adds to the error. Finally, in case of P2P-TV, the uplink capacity measure of the *receiver peer* is generally needed, which is not straightforward since the measuring peer cannot rely on the chunk transmission process. For the above reasons, we can expect capacity measurement errors to be larger in magnitude with respect to latency errors. Formally, we denote by $C(p', p)$ the actual bottleneck capacity in the path from p' to p; we consider that peer p will erroneously estimate $C(p', p)$ as $\tilde{C} \sim \mathrm{N}(C(p', p), \alpha C)$: in other words, peers capacity estimate is a normally distributed variable, with mean equal to the actual bottleneck capacity $C(p', p)$ and a variance equal to αC.

In Fig. 6 we explore errors ranging in $\alpha \in [0, 5]$: despite we consider such a large range, we observe that performance is rather robust: indeed, only a marginal increase of latency (and of losses, in case topology management is disabled) can be observed. This can be explained with the fact that, provided that measurement errors still allow to clearly separate the peers in classes [4], performance of bandwidth-aware algorithms remains consistent. Given the challenges in capacity and bandwidth measurement, this intrinsic robustness is a very advantageous feature, as a rough binary discrimination capability in high-capacity vs low-capacity peers may be enough for network-aware algorithms.

Overall, capacity estimation precision has a small impact on P2P-TV performance: indeed, provided that peer classes are sufficiently separated, it is always possible to differentiate among classes even in presence of measurement errors.

5.2 Signaling errors

We now investigate the effect of signaling errors on the system performance. Recall that, in order to send chunks that are *useful* for the receivers, each peer must have a precise knowledge of its neighbor buffer-maps. This can be accomplished by periodically exchanging buffer-map status in control packets, or by piggybacking buffer maps in data packets. In order for this knowledge to be as up-to-date as possible, any peer p should inform all of its neighbors as soon as it receives a new chunk c. Still, even in this "perfect signaling system", due to the unavoidable latency of the signaling process (both propagation and transmission delay), it is still possible that p schedules the transmission of a chunk c to a peer p' that has just received it (but not sent its buffer map $B(p')$ out yet), thus generating a collision. Furthermore, loss of signaling messages can happen in the L3 network, further degrading the quality of peers knowledge. Clearly, frequent buffer-map exchange has a high cost in terms of overhead: as several new chunks are generated at each second, the signaling process would thus need to be continuous in order for peers to have an up-to-date view of their neighborhood. However, lowering the signaling rate to reduce the messaging overhead also increases the chance for collisions to occur.

5.2.1 Impact on L7

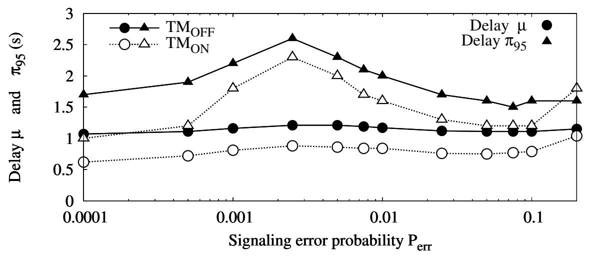
Up to now, we have evaluated network-aware P2P-TV systems performance by assuming that peers have a perfect instantaneous knowledge of the buffer maps of their neighbors – a rather unrealistic assumption. We therefore model the impact of low signaling rates (or signaling messages losses at L3) as a quality degradation of system state knowledge in the distributed P2P system. In more detail, we model imprecision of system state knowledge as "usefulness" errors: in other words, with a given probability P_{err} a peer p can take a scheduling decision of chunk c toward p' which he believes to be useful (i.e., $c \notin B(p')$) despite it is not (i.e., $c \in B(p')$), which generates a collision.

Conversely, we do not consider the opposite kind of errors (i.e., p believes $c \in B(p')$ despite actually $c \notin B(p')$), as this would indeed model a somewhat unlikely case of *misconfigured* peers sending erroneous updates (i.e., advertising chunk $c \in B(p')$ to be available while it is not).

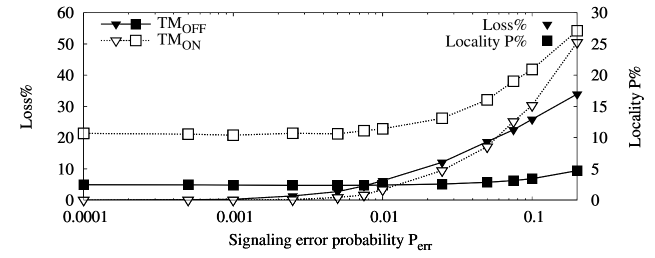
Fig. 7 shows the mean μ and 95th percentile π_{95} delay, along with chunk loss statistics. Notice that while the mean delay is roughly unaffected by signaling error probability P_{err}, a counter-intuitive phenomenon characterizes the π_{95} measure. Indeed, the 95th delay percentile increases until $\mathrm{P}_{err} = 1/400$, and afterward starts decreasing: this behavior is strongly correlated to the chunk loss rate, which starts rising roughly at $\mathrm{P}_{err} = 1/400$. What happens is that for increasing P_{err}, peers indeed receive chunks with higher delay, which in turns raises the probability that chunks arrive beyond the playout delay (i.e., delay larger than 5s), and are thus marked as lost: as lost chunks are not accounted in the delay curve, the peak is thus an artifact due to the playout deadline.

Traffic locality exhibits a non-straightforward behavior as well: indeed, it can be seen from Fig. 8 (left y-axis) that

[4] Measurements are correct with a very coarse granularity, from Tab. 1 we have that $\frac{BW_U^i}{BW_U^{i+1}} > 2,\ \forall i$ which means that we may correctly separate peers into classes even when the measurement precision is rather poor.



(a)



(b)

Fig. 7 Delay and chunk losses as a function of signaling errors. reports chunk delay mean μ and 95-th percentile π_{95}, (b) reports $Loss\%$ and locality $P\%$. Both topology management TM settings are used.

locality increases as buffer-map errors increase as well, which is especially visible in case of topology management. This can be explained by considering that the lu/pa scheduler preferentially selects nearby high-capacity peers. When the error probability is low, these peers will be fed first, but then, as peers rarely fail in estimating the usefulness of their decisions, other lower-capacity higher-latency peers get successfully served during the remaining upload slots. Conversely, when error probability is high, the scheduler will keep on sending chunks to close high-capacity neighbors, despite they likely already have received that chunk from other peers (notice that we forbid peers sending the same chunk to the same peer multiple times).

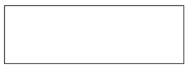
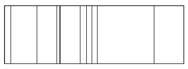
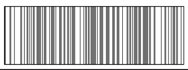
We acknowledge that it is hard to relatively compare the magnitude of the error intensity across the different metrics. Nevertheless, it can be safely pointed out that while system performance remains practically unaffected over the whole range of *measurement* errors, even small *signaling* error rates have a much more visible impact.

Overall, system performance are extremely sensitive to errors due to stale signaling: effects are noticeable on the tail of the delay distribution for error rates as low as $P_{err} = 1/1000$ *and loss probability become excessive for error rates as low as* $P_{err} = 1/100$.

5.2.2 Impact on Quality of Experience (QoE)

We then dig the user Quality of Experience (QoE) by evaluating an objective video quality metric, namely the Peak Signal to Noise Ratio (PSNR). We consider the standard Soccer

Table 3 PSNR values for the Source, a class-III and a class-IV peer

Class	PSNR	Delay $\mu(\pi)$	Lost chunks	Loss pattern
Source	39.4	–	–	
II	16.1	1.18s (4.3s)	45 (2.2%)	
IV	12.1	1.28s (3.7s)	487 (24.3%)	

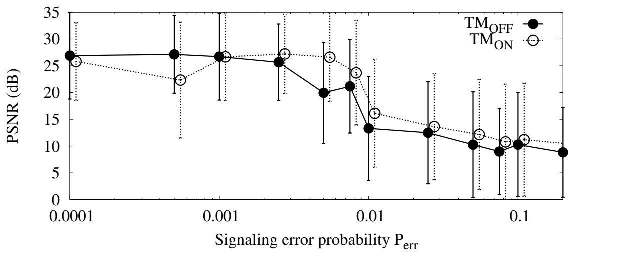


Fig. 8 PSNR evaluation as a function of signaling errors. Bars represent the standard deviation of PSNR values gathered over the 10-sample population. PSNR of the original video sequence at the source is 35.6 dB.

sequence (H624 format, CIF resolution, 300 frames @30Hz, looped for the whole simulation duration), and record for each peer the list of lost chunks. We then make use of Evalvid [42] to evaluate video quality, by feeding the tool with the video sequences where we take into account the chunk loss pattern for each peer. To give the reader an intuition of PSNR value, and how it roughly corresponds to other system performance, we report in Tab. 3 a few examples. Each row refers to either the video source, a peer of class-II, or a peer of class-IV, and reports different performance metrics such as delay (mean and 95th percentile) and loss statistics,.with a graphical representation of the loss pattern, where each vertical bar corresponds to a loss. As PSNR evaluation is very time consuming and due to the size of our system, we resort to stratified sampling: specifically, we rank peers according to the amount of losses and select a 10-peers sample (corresponding to different loss amounts) out of the total $N_H = 2000$ peer population. Right y-axis of Fig. 8 reports the PSNR averaged over the 10-peers sample (bars report the standard deviation over the sample), which due to stratification is however representative of the whole population. It can be seen that PSNR drops as soon as a losses occur in the system: notice further that, since a PSNR<24 dB is generally considered as an indicator of extremely bad video quality, this suggest that buffer-map errors should be kept below $P_{err} < 1/100$.

Overall, as in our evaluation buffer maps hold 50 chunks and a new chunk is generated every 100ms, this suggests

Table 4 Per-class performance breakdown

	Ideal $P_{err} = 0\%$			Harsh $P_{err} = 5\%$			$\mathcal{D}$		
	μ	π_{95}	L%	μ	π_{95}	L%	μ	π_{95}	L%
I	0.4	0.6	10^{-3}	0.8	1.3	0.2	*1.9*	*2.1*	*92.2*
II	0.6	1.0	10^{-2}	0.8	1.1	4.4	*1.2*	*1.1*	*263.9*
III	0.7	1.1	10^{-2}	0.8	1.2	8.5	*1.1*	*1.2*	*1421.3*
IV	0.9	1.3	0.2	1.1	1.4	33.9	*1.2*	*1.1*	*198.6*
$\mathcal{F}$	*2.2*	*2.2*	*68.2*	*1.3*	*1.1*	*146.9*			

that the signaling rate should be about as high as the chunk generation rate.

5.2.3 Impact on Peer Class

For the sake of completeness, we analyze how system performance vary across the different classes, comparing ideal (P_{err}=0%) and harsh (P_{err}=5%) settings, so to gather performance bounds. To quantify the impact of P_{err}, we define the *intra-class degradation factor* as:

$$\mathcal{D}(X) = X(P_{err} = 5\%|c)/X(P_{err} = 0\%|c) \qquad (1)$$

where X is any metric considered in the table and c the considered class: intuitively, $\mathcal{D}(X)$ is a compact indicator of the performance loss from ideal to realistic settings.

To quantify the fairness of the results among classes, we define a *inter-class fairness factor* as:

$$\mathcal{F}(X) = \max_c X(c|P_{err})/\min_c X(c|P_{err}) \qquad (2)$$

intuitively, $\mathcal{F}$=1 corresponds to comparable performance across classes, while the larger the value of $\mathcal{F}$>1, the larger the unfairness.

Results are reported in Tab. 4. Notice that, already on ideal settings, class-IV peers experience a delay twice than that of class-I peers, and about 68 times more losses (loss $L\%$ unfairness exceeds a factor of 146 in harsh settings). Concerning the degradation due to signaling error, we see that class-I experiences a larger delay degradation than other classes but a limited loss increase: in other words, average delay increases but not enough to exceed the playout deadline (and to cause losses). The opposite happens instead for class-II and III, whose delay . The extent of class-IV degradation is instead smaller, however the absolute amount of losses exceed 33%, which likely makes video quality unbearable.

We conclude that, even in absence of signaling errors, performance breakdown is unfair with respect to peer belonging to different classes. Under harsh signaling errors, delay become more fair among classes, with however extremely large loss rates for peers belonging to the poorer classes.

6 Conclusions

In this work, we compare different state-of-art "network-aware" P2P-TV systems, i.e., systems whose main algorithms (such as chunk selection and topology management) are based on informed decisions concerning the status of the network. We define a flexible framework, able to accommodate further aspects beyond to the one we focus on in this work, and perform a thorough simulation campaign: our purpose is to understand what are the main factors that affect P2P-TV performance, and to what extent performance degrades under realistic settings.

Our main findings can be summarized as follows. First, we find the impact of the L3 underlay network model to be modest, with a small performance gap between simple (e.g., constant and fixed delay) and realistic models (e.g., meridian latency or dynamic latencies). This owes to the fact that the propagation delay has a smaller impact with respect to transmission delay, especially considering the relative low-capacity of current scenario. At the same time, we can expect that as the access capacity increases, the impact of propagation latency may need to be reconsidered.

Second, we find that system performance is rather robust to errors in the measurement of peer properties. More precisely, provided that peers capacity is clearly separated, the ability to roughly discriminate high-capacity from low-capacity peer is sufficient to guarantee a good level of performance. Similarly, errors in the latency estimation only affect the traffic locality, but system performance are otherwise unaltered.

Finally, we find the impact of signaling errors to be, by far, the most important factor able to significantly degrade the quality of P2P-TV services and is able to severely impact overlay performance already from very low intensities. We acknowledge that it is hard to relatively compare the magnitude of the error intensity for heterogeneous metrics such as latency, bandwidth and signaling. Nevertheless, it can be safely pointed out that while system performance remains practically unaffected over the whole range of *measurement* errors, even small *signaling* error rates can have a dramatic impact.

This suggests that, in order to gather reliable estimate of the achievable P2P-TV system performance, special attention must be payed to the signaling logic – which holds irrespectively of the adopted analytical, simulative or experimental methodology. As future work, we aim at studying the interplay between chunk size and buffer maps update rate and exploring the trade-off between the overhead caused by high signaling rate and the outdated knowledge of neighborhood buffer-maps due to low signaling rate.

Acknowledgements

This work was supported by the European Commission under the FP7 STREP Project Network-Aware P2P-TV Application over Wise Networks (NAPA-WINE).

References

1. "Cisco visual networking index: Forecast and methodology, 2009-2014." http://www.cisco.com/en/US/solutions/-collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-481360_ns827_Networking_Solutions_White_Paper.html, June 2010.
2. "Cisco visual networking index: Global mobile data traffic forecast update, 2009-2014." http://www.cisco.com/en/US/-solutions/collateral/ns341/ns525/ns537/ns705/ns827/white-_paper_c11-520862.html, February 2010.
3. "Spotify consumes more internet capacity than all of sweden." http://techcrunch.com/2010/03/16/live-blog-spotify-ceo-daniel-eks-keynote-interview/.
4. "Project pheon homepage." http://www.utorrent.com/labs/pheon.
5. "Adobe real time media flow protocol (rtmfp)." http://labs.adobe.com/technologies/stratus/.
6. G. Fox and S. Pallickara, "The Narada event brokering system: Overview and extensions," in *Parallel and Distributed Processing Techniques and Applications, PDPTA '02*, (Las Vegas, USA), June 2002.
7. D. Tran, K. Hua, and T. Do, "A peer-to-peer architecture for media streaming," *IEEE Journal on Selected Areas in Communications*, vol. 22, pp. 121 – 133, Jan. 2004.
8. J. Zhang, L. Liu, L. Ramaswamy, and C. Pu, "PeerCast: Churn-resilient end system multicast on heterogeneous overlay networks," *Journal of Network and Computer Applications*, vol. 31, no. 4, pp. 821–850, 2008.
9. R. Sherwood, S. Lee, and B. Bhattacharjee, "Cooperative peer groups in NICE," *Computer Networks*, vol. 50, no. 4, pp. 523–544, 2006.
10. M. Castro, P. Druschel, A. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, "Splitstream: high-bandwidth multicast in cooperative environments," in *Proceedings of the nineteenth ACM symposium on Operating systems principles*, (Bolton Landing, USA), 2003.
11. D. Manzato and N. da Fonseca, "Incentive mechanism for the CoopNet network," *Peer-to-Peer Networking and Applications*, vol. 1, pp. 29–44, 2008.
12. R. Rejaie and A. Ortega, "PALS: peer-to-peer adaptive layered streaming," in *Proceedings of the 13th international workshop on Network and operating systems support for digital audio and video, NOSSDAV '03*, (Monterey, USA), ACM, 2003.
13. B. Li, S. Xie, Y. Qu, G. Keung, C. Lin, J. Liu, and X. Zhang, "Inside the new coolstreaming: Principles, measurements and performance implications," in *Infocom 2008*, (Phoenix, USA), IEEE, April 2008.
14. F. Pianese, D. Perino, J. Keller, and E. Biersack, "PULSE: an adaptive, incentive-based, unstructured P2P live streaming system," *IEEE Transactions on Multimedia*, vol. 9, no. 8, pp. 1645–1660, 2007.
15. N. Magharei and R. Rejaie, "Prime: Peer-to-peer receiver-driven mesh-based streaming," *IEEE/ACM Transaction on Networking*, vol. 17, no. 4, 2009.
16. H. Chang, S. Jamin, and W. Wang, "Live streaming performance of the Zattoo network," in *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference*, ACM, 2009.
17. T. Bonald, L. Massoulié, F. Mathieu, D. Perino, and A. Twigg, "Epidemic live streaming: optimal performance trade-offs," in *ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, (Annapolis, USA), 2008.
18. R. Bindal, W. Chan, J. Medved, G. Suwala, T. Bates, and A. Zhang, "Improving traffic locality in bittorrent via biased neighbor selection," in *International Conference on Distributed Computing Systems, ICDCS*, (Lisboa, POR), 2006.
19. A. P. C. D. Silva, E. Leonardi, M. Mellia, and M. Meo, "A Bandwidth-Aware Scheduling Strategy for P2P-TV Systems," in *Peer to Peer computing, P2P'08*, (Aachen, GER), IEEE, 2008.
20. D. Ren, Y. Li, and S. Chan, "On Reducing Mesh Delay for Peer-to-Peer Live Streaming," in *Infocom 2008*, (Phoenix, USA), IEEE, April 2008.
21. B. Zhao, J. Lui, and D. Chiu, "Exploring the optimal chunk selection policy for data-driven P2P streaming systems," in *Peer to Peer computing, P2P'09*, (Seattle, USA), IEEE, 2009.
22. N. Magharei and R. Rejaie, "Mesh or multiple-tree: A comparative study of p2p live streaming services," in *Infocom 2007*, (Anchorage, Alaska), IEEE, May 2007.
23. J. Seibert, D. Zage, S. Fahmy, and C. Nita-Rotaru, "Experimental comparison of peer-to-peer streaming overlays: An application perspective," in *IEEE Conference on Local Computer Networks, LCN'08*, (Montreal, Canada), 2008.
24. D. Rossi and P. Veglia, "Assessing the impact of signaling on the qoe of push-based p2p-tv diffusion algorithms," in *Conference on New Technologies, Mobility and Security, NTMS'11*, (Paris, FRA), IEEE, 2011.
25. V. Pai, K. Kumar, K. T. andy Vinay Sambamurthy, and A. E. Mohr, "Chainsaw: Eliminating trees from overlay multicast," *Lecture Notes in Computer Science*, 2005.
26. D. Ciullo, M. A. Garcia, H. Akos, E. Leonardi, M. Mellia, D. Rossi, M. Telek, and P. Veglia, "Network awareness of p2p live streaming applications: A measurement study," *IEEE Transaction on Multimedia*, 2010.
27. F. Picconi and L. Massoulié, "Isp friend or foe? making p2p live streaming isp-aware," in *In Proc. of IEEE International Conference on Distributed Computing Systems (ICDCS'09)*, (Montreal, Quebec, Canada), IEEE, 2009.
28. S. L. Blond, A. Legout, and W. Dabbous, "Pushing bittorrent locality to the limit," *Elsevier Computer Networks*, vol. 55, no. 3, pp. 541 – 557, 2011.
29. A. Rao, A. Legout, and W. Dabbous, "Can Realistic BitTorrent Experiments Be Performed on Clusters?," in *IEEE International Conference on Peer-to-Peer Computing (P2P'10)*, (Delft, Netherlands), Aug. 2010.
30. http://www.napa-wine.eu/cgi-bin/twiki/view/Public/P2PTVSim.
31. M. Piatek, T. Isdal, T. Anderson, A. Krishnamurthy, and A. Venkataramani, "Do incentives build robustness in BitTorrent?," in *Symposium on Networked System Design & Implementation, NSDI'07*, (Cambridge, USA), USENIX, 2007.
32. M. Piatek, H. V. Madhyastha, J. P. John, A. Krishnamurthy, and T. Anderson, "Pitfalls for isp-friendly p2p design," in *ACM Workshop on Hot Topics in Networks (HotNets-VIII)*, (New York City, USA), 2009.
33. K. Gummadi, S. Saroiu, and S. Gribble, "King: Estimating latency between arbitrary Internet end hosts," in *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet Measurment*, (Marseille, FRA), 2002.
34. "Modelnet-te homepage." http://www.enst.fr/ drossi/index.php?n=Software.ModelN TE, 2010.
35. A. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, p. 509, 1999.
36. D. Watts and S. Strogatz, "Collective dynamics of "small-world" networks," *Nature*, vol. 393, no. 6684, 1998.
37. "Alto ietf working group." http://datatracker.ietf.org/wg/alto/charter/.
38. "Meridian project." http://www.cs.cornell.edu/People/egs/meridian/.

39. R. J. Lobb, A. P. Couto da Silva, E. Leonardi, M. Mellia, and M. Meo, "Adaptive overlay topology for mesh-based P2P-TV systems," in *International Workshop on Network and Operating Systems Support for Digital Audio and Video, NOSSDAV'09*, (Williamsburg, USA), ACM, 2009.
40. A. Shriram, M. Murray, Y. Hyun, N. Brownlee, A. Broido, M. Fomenkov, and K. Claffy, "Comparison of public end-to-end bandwidth estimation tools on high-speed links," *Passive and Active Measurement conference, PAM'05*, 2005.
41. D. Croce, M. Mellia, and E. Leonardi, "The quest for bandwidth estimation techniques for large-scale distributed systems," *ACM SIGMETRICS Performance Evaluation Review*, vol. 37, no. 3, pp. 20–25, 2009.
42. J. Klaue, B. Rathke, and A. Wolisz, "Evalvid a framework for video transmission and quality evaluation," in *Computer Performance*, vol. 2794 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, 2003.
43. E. Leonardi, M. Mellia, A. Horvath, L. Muscariello, S. Niccolini, and D. Rossi, "Building a cooperative p2p-tv application over a wise network: the approach of the european fp-7 strep napa-wine," *IEEE Communication Magazine*, vol. 64, April 2008.
44. A. Finamore, M. Mellia, M. Meo, and D. Rossi, "Kiss: Stochastic packet inspection classifier for udp traffic," *IEEE Transactions on Networking*, vol. 18, pp. 1505 – 1515, October 2010.
45. S. Valenti, D. R. amd M. Meo, M.Mellia, and P. Bermolen, "Abacus: Accurate behavioral classification of p2p-tv traffic," *Elsevier Computer Networking, to appear*, 2010.
46. A. Finamore, M. Mellia, M. Meo, D. Rossi, and S. Valenti, "Kiss to Abacus: a comparison of P2P-TV traffic classifiers," in *Traffic Measurement and Analyis (TMA'10), LNCS*, (Zurich, Switzerland), April 2010.
47. A. Finamore, M. Mellia, M. Meo, D. Rossi, and S. Valenti, "Peer-to-peer traffic classification: exploiting human communication dynamics," in *Globecom'10 Demo session*, (Miami, USA), IEEE, December 2010.
48. "Internet traffic classification demo webpage." http://perso.telecom-paristech.fr/ drossi/index.php?n=Software.ClassificationDemo.
49. "Pplive." http://www.pptv.com/en/.
50. "Sopcast." http://www.sopcast.org/.
51. "Tvants." http://tvants.en.softonic.com/.
52. F. Mathieu, "Heterogeneity in data-driven live streaming: Blessing or curse?," in *IEEE International Symposium on Parallel Distributed Processing (IPDPS)*, pp. 1–8, April 2010.
53. Bandwidth-test.net, "Bandwidth test statistics across different countries." http://www.bandwidth-test.net/stats/country/.

A Appendix

This section describes our careful selection of some crucial parameter of our simulations campaign, such as (i) the population size and stability, (ii) the sharing ratio and (iii) the upload bandwidth. We stress that our choice was to gather simulation scenarios that are as representive as possible of the Internet and real-life: hence, we used own measurement, or other relevant measurement performed by colleagues in the scientific community, to derive a realistic set of simulation parameters.

A.1 Population size and stability in real P2P-TV systems

To justify our assumption of absence of churn in the population, we show in Fig. 9 the stability of a real user base population in the operational network of a major European ISP that we continuously monitor in the context of the NAPAWINE project [43]. For the same project,

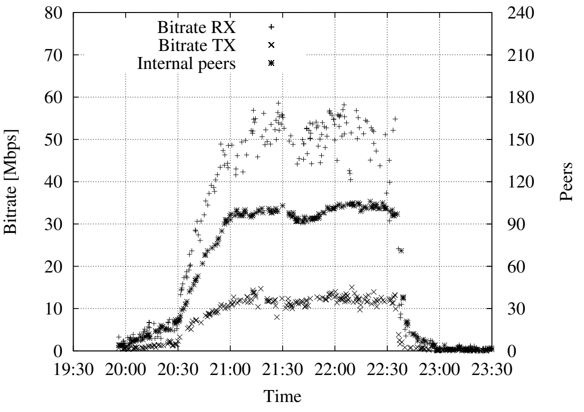


Fig. 9 Temporal evolution of the number of peers, received and sent traffic volume during a typical P2P-TV event (SopCast application), at a PoP of a major European ISP that we continuously monitor.

we have developed state of art P2P-TV classifiers, that are either based on the stochastic analysis of the packet payload (KISS [44]) or on the behavioral analysis of the connection pattern (Abacus [45]). The classifiers are comparably accurate [46,47] and an open source implementation is available at [48]. We run the classifiers on several probes in different major European ISP, so that we are able to recognize the traffic of popular applications such as PPlive [49], SopCast [50], TVAnts [51], as they are used by real user in operational networks.

While in general the usage of P2P-TV application is episodic, as it is driven by a specific program –rather typically, a sport event– *during* the event the population remains *extremely stable.* We support this statement with the help of Fig. 9, which reports the temporal evolution of the number of peers, depicted with a star point on the right y-axis, during a typical sport event streamed by the popular SopCast [50] application gathered during a Championship match in April 2009. Each point in the picture reports a measurement related to 5 seconds, and we sub-sample the observation points for the sake of readability. The picture also reports, on the left x-axis, the received (plus sign) and sent (cross sign) bitrate in Mbps. As it can be seen from the picture, peers arrive in a flash-crowd pattern starting from 20h30 (thus prior that the match begins), while during the whole 1h30-long soccer match the peer population keeps extremely stable to about 100 peers (i.e., the value that we actually simulated). Then, immediately after the end of the match, peers rapidly depart and the system empties. Also, noticeable from the picture, the traffic contributed by the peer behind the residential point of presence (PoP) is lower than the received traffic, which is due to the asymmetry typical of ADSL lines.

This pattern is very common in our measurements and justify our environment choice of *absence of churn*. First, by focusing on steady-state performance of different algorithms, we gather results that are not only more relevant (as they pertain to the whole duration of the show, rather that to a startup phase), but are also statistically more significant with respect to performance gathered during a transient phase (i.e., during arrival or departure). Second, it is likely that the algorithm exhibiting the best performance in steady-state, will also be the best candidate in the transient phase. Third, these results are also more relevant from the end-user perspective, as users are clearly interested in the QoS during the match, while they are likely less interested in the system performance prior that the match begins: indeed, notice how arrival time roughly uniformly distributes in the 30 minutes preceding the match, which is more likely tied to user "warm-up" of the channel (i.e., joining the P2P-TV system in advance to be sure seeing the first kick of the match) and personal habits rather than reflecting actual interactive usage of the channel. Finally, we point out that churn could

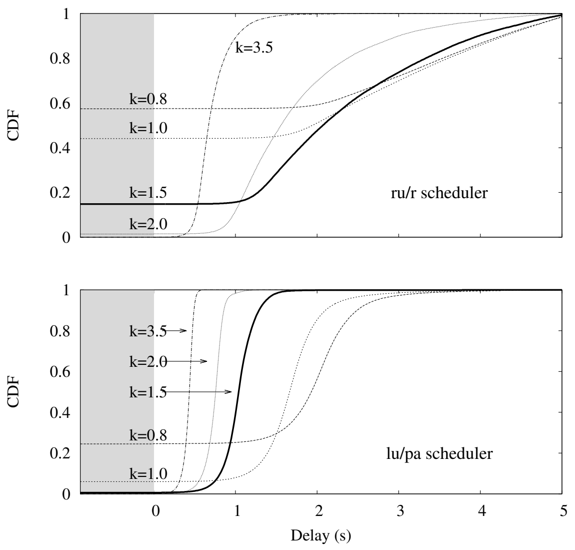


Fig. 10 Delay distribution of ru/r and lu/pa schedulers for different values of the sharing ratio k

not instead be disregarded in case the performance metrics of interest pertained to the transient phase (e.g., the duration of the warmup phase needed to reach a steady state) or to resilience aspects of the system (e.g., , measuring the system response in case everybody changed the channel during the break of the football match), which are however out of the scope of this paper.

A.2 Sharing ratio

Sharing ratio k is a fundamental and critical parameter in every P2P system since it describes its capacity to disseminate data to all nodes: k is defined as

$$k = \frac{\overline{BW_U}}{\lambda_{video}} = \frac{\sum_{i \in N_H} BW_{U\,i}}{N_H \cdot \lambda_{video}} \tag{3}$$

or the ratio between the total uplink capacity (i.e. the sum of uplink capacity of all nodes) and total inbound traffic; intuitively a value of $k > 1$ means that nodes uplink capacity is sufficient for the system to be self-sustainable as long as algorithms are good enough to exploit capacity. On the contrary, in a scenario where k is < 1, total up-link capacity is not high enough to guarantee that every peer receives the entire stream. Notice that, as the value of k provided by common ADSL peers in actual system is between 0.2 and 0.3 [16], the correct working of the system is guaranteed by the presence of "amplifier" nodes providing the missing capacity.

Notice that scheduling algorithm may not be able to succesfully exploit the available system capacity even for $k > 1$. Considering for simplicity the case of a single-class homogeneous population, top plot Fig. 10 show for instance the CDF of the chunk delay for the naïve *ru/r* scheduler, where it can be seen that for $k = 1$, almost half of chunks are lost and even when the capacity is twice $k = 2$ the needed sustainable rate, the system still experiences a non negligible amount of 1.4% losses.

In case more sophisticated schedulers are used, such as the power-aware lu/pa in the bottom plot of Fig. 10, we notice that the system is almost lossless for a sharing ratio of $k = 1.5$. Hence, in our simulation we designed the class population to match the factor $k = 1.5$, so that the system is self-sustainable, which allows to isolate the impact of other factors (e.g., L3 network, L7 schedulers, L3/L7 interactions) on an otherwise lossless system.

In addition, an interesting aspect emerges from this analysis: comparing the $k = 1.5$ homogeneous single-class system of Fig. 10 to the $k = 1.5$ heterogeneous multi-class system shown early in Fig. 2(a), we see that peer heterogeneity plays a non marginal role in improving global efficiency [52]. Intuitively, having peers with different capacity is beneficial because high capacity peers, which can handle a greater number of active connections, will occupy the high portion of the istantaneous chunk distribution tree, close to the source, possibly as a result of topology management. On the other hand, peers with poor connectivity can occupy far positions in each chunk distribution tree, since they can serve a lower number of neighbors (or even none).

A.3 Mean upload bandwidth

In Appendix A.2 we fixed a value for parameter k, binding video rate to mean upload bandwidth: hence, we need to investigate sound values for the peer upload capacity. To the best of our knowledge, there are no studies or tools, as for instance the Meridian project for latencies, which estimate end-host bandwidth distribution. Moreover, even if there were any, the gathered data would likely be strongly dependent on countries or service provider. For these reasons we perform a survey, to gather rough boundaries for the uplink capacity, as well as a sensitivity analysis, to gather reliable simulation results between these boundaries.

Again, for the sake of simplicity, we consider an homogeneous peer population, where each peer belongs to a single class with upload bandwidth BW_U. We perform a sensitivity analysis by carrying on several simulations varying the relative magnitudo of the propagation and trasmission time, by varying BW_U. Notice moreover that latencies, chunk-size and buffer-map size are kept constant. In more detail, we denote by γ the ratio between the propagation delay and the chunk upload time:

$$\gamma = \frac{\overline{M}}{t_{TX}} = \overline{M}\frac{BW_U}{C} \tag{4}$$

where $\overline{M}$ is the average Meridian end-to-end latency, t_{TX} is the chunk transmission time and C the average chunk size. Intuitively γ indicates which component of the delay has the largest influences on the total chunk transfer time. For instance, when $\gamma < 1$ the transmission delay is greater than the propagation delay, so that the total chunk delay reduces in case of bandwidth-awareness; conversely, when $\gamma > 1$, propagation delay is the largest component of total chunk transmission time, which would thus reduce in case of latency-awareness.

To gather valid boundaries for γ we surveyed the average uplink capacity of different European countries [53], to gather upper (CZ) and lower (IT) bounds of $\overline{BW_U}$ (and, hence, of γ). Fig. 11 depicts several performance metrics (i.e., chunk losses, mean and 95th percentile of the total chunk delay) as a function of γvarying in the range resulting from the survey. As γ grows, we notice a smooth decrease of the delay curves, which is an expected consequence of the transmission time reduction due to higher upload bandwidth. Clearly, this decrease is not an artifact occasioned by greater losses (as gathered early in section 5.2 but a real gain), which is confirmed by the consistently low loss rate in the same interval.

The vertical thick line in between IT and CZ references represent the working point that we selected for our simulation, which can be thought to represents an average European country. Two considerations hold: first, notice that, as loss rate remains steady over the whole interval, we can expect the results shown earlier in this paper, to hold for a number of different European countries. Second, we gather that, as γ

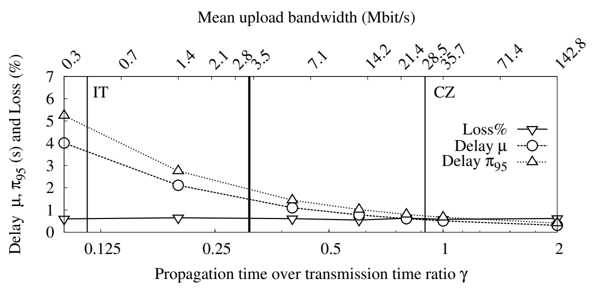


Fig. 11 Sensitivity analysis of delay and losses as a function of the propagation-delay over transmission-delay ratio γ. Top x-axis represent the average per-country bandwidth reported in [53], with landmarks for the lower (Italy, IT) and upper (Czech republich, FR) bounds; with a thick line, we report the $\overline{BW_U}$ value we selected in this paper.

varies in the selected range, the delay can vary by almost one order of magnitude: hence, our simulation results should be considered as representative of an average country, and we can expect delay results to (roughly linearly) vary depending on the actual value of $\overline{BW_U}$ in the country of interest.