QoE in Pull Based P2P-TV Systems: Overlay Topology Design Tradeoff

QoE in Pull Based P2P-TV Systems: Overlay Topology Design Tradeoffs

R. Fortuna[1], E. Leonardi[2], M. Mellia[2], M. Meo[2], S. Traverso[2]
[1]Politecnico di Bari, Italy – email: r.fortuna@poliba.it
[2]Politecnico di Torino, Italy – email: {firstname.lastname}@polito.it

***Abstract*—This paper presents a systematic performance analysis of pull P2P video streaming systems for live applications, providing guidelines for the design of the overlay topology and the chunk scheduling algorithm. The contribution of the paper is threefold: 1) we propose a realistic simulative model of the system that represents the effects of access bandwidth heterogeneity, latencies, peculiar characteristics of the video, while still guaranteeing good scalability properties; 2) we propose a new latency/bandwidth-aware overlay topology design strategy that improves application layer performance while reducing the underlying transport network stress; 3) we investigate the impact of chunk scheduling algorithms that explicitly exploit properties of encoded video. Results show that our proposal jointly improves the actual Quality of Experience of users and reduces the cost the transport network has to support.**

I. Introduction

Mesh based P2P streaming systems (P2P-TV in short) are between the most promising solutions for the broadcast of real time video contents over the Internet. Indeed, they offer to content providers and broadcasters the opportunity of reaching a potentially unlimited audience without expensive infrastructural investments.

Similarly to file sharing P2P systems, in mesh based P2P-TV systems the video content encoded at the source is sliced in pieces called *chunks*, which are distributed over a meshed overlay topology exploiting a fully distributed epidemic approach. But differently from file sharing P2P systems, chunks must be received by the peers within a *deadline*, so that delivery delay is one of key aspect of these systems. The other major difference with respect to file sharing relies on the fact that the video content is produced in *real time, and at a limited bitrate*. This makes P2P-TV system design deeply different from file sharing applications design.

Two key architectural ingredients of P2P-TV systems are: the *chunk scheduling algorithm* according to which peers exchange chunks, and the logic adopted to create and maintain the *overlay topology*. For the *chunk scheduling*, several algorithms have been proposed and analyzed in an idealized scenario [1], [2], [3], [4], [5] (see Sec II for a discussion of related works). Most notably, a common assumption of these works is that each peer has a perfect knowledge of the chunks other peers possess, or, in other terms, impact of latency between peers is completely ignored. Similarly, only few papers specifically focus on the impact of peer access bandwidth heterogeneity, a crucial aspect since peers homogeneity is hardly met in practice today. Considering the *overlay topology* design, performance of mesh based systems has been typically analyzed assuming the overlay topology to be either a fully connected mesh or an homogeneous random graph [6], [7], [8], [9]. Important questions to be still addressesd are: what is then the impact of latencies and peer access bandwidth heterogeneity on the performance of a P2P-TV system? Is it possible to design smart overlay topologies that explicitly consider them? And also, how can we reduce then the underlying transport network load by designing a smart system? On this regard, we emphasize that building new P2P applications that reduce the underlay transport network resource consumption is an important issue. While this problem has received a lot of attention considering P2P file sharing applications (see [10] and references therein), the only work that focuses on P2P-TV applications that we are aware is [11]. Therefore, it is still a matter of debate how to design P2P-TV applications that minimize the impact on underlying network, while still guaranteeing good performance for the users.

At last, we would like to emphasize that all the previously mentioned papers considered a rather naive model of an encoded video stream, according to which fixed size chunks are generated by the source at constant rate. However, actual video streams carry highly organized information, part of which is more important than other, and with high variability in the generated bitrate. Chunk loss probability and delivery delay (which are performance indexes typically adopted by the networking community) provide therefore only a partial view of the actual performance of a P2P-TV system, the user Quality of Experience (QoE) being the paramount index. In the multimedia and signal processing communities, indeed, the evaluation of the QoE is considered mandatory, see [12], [13] for notable examples.

In this paper, we evaluate for the first time, to the best on our knowledge, the performance of P2P streaming systems in realistic scenarios in which peer *bandwidth heterogeneity*, *network latencies* and *properties of encoded video* streams are accounted for. The main contributions of the paper are the following:

- We propose a realistic simulative model of the system that represents the effects of bandwidth heterogeneity, latencies, peculiar characteristics of the video, while still guaranteeing good scalability properties.
- We show how information about peers and network properties can be effectively exploited by the application to improve

performance and user QoE while significantly reducing the resource demand for the underlying transport network. In particular, we propose a simple latency/bandwidth-aware overlay topology design strategy that is based on information about peer upload bandwidth and end-to-end path latencies.

• We investigate the impact on QoE of video aware scheduling algorithms, successfully and largely employed in other contexts. The intuition, indeed, suggests that the user QoE would benefit by explicitly exploiting properties of encoded video. However, results are somehow unexpected.

The proposed algorithms are extensively tested by simulations, considering different network scenarios and actual video sequences. Results show consistent improvement of actual users' QoE. Moreover, the traffic load on the network to support the application is also reduced, an important and timely achievement. All shown solutions are being implemented in the new P2P-TV application under development within the EU-FP7 NAPA-WINE STREP project [14].

II. Related work

Two key architectural ingredients of P2P-TV systems are: the *chunk scheduling algorithm* according to which peers exchange chunks, and the logic adopted to create and maintain the *overlay topology*. For the *chunk scheduling*, several algorithms have been proposed and analyzed in an idealized scenario in which each peer is supposed to instantaneously have a perfect view of the internal state of other peers and, in particular, of the chunks they need [1], [2], [3], [4], [5]. This assumption, however appears rather far from reality in light of the fact that in practical settings the typical end-to-end latencies between peers are usually comparable or even larger than time constants associated to chunks dynamics. The impact of latency on system performance has been analyzed by means of a simple model corroborated by real measurements in PlanetLab in [15]. The authors propose a system that mitigates the effect of latency introduced by the need for exchanging state information. Furthermore, while many of the previous works have considered homogeneous scenarios in which all peers are indistinguishable, few papers specifically focus on the impact of peers bandwidth heterogeneity and how it can be exploited to improve system performance [4], [16], [7]. This aspect seems crucial, peers homogeneity being hardly met in practice where narrow bandwidth residential users coexist with larger bandwidth business/residential users.

The *overlay topology* design has been investigated less deeply and the system performance has been typically analyzed assuming the overlay topology to be either a fully connected mesh or an homogeneous random graph. In [6] the problem of building an efficient overlay topology, taking into account both latency and bandwidth, has been formulated as an optimization problem; in this case, however, peers do not exchange chunks, but continuous streams of information are considered. In [7] a theoretical investigation of optimal topologies is formulated, considering latency and peer bandwidth heterogeneity; scaling laws are thus discussed. In work [8], a distributed and adaptive algorithm for the optimization of the overlay topology in heterogenous environments has been proposed, but network latencies are still ignored. In [9], it is proposed a push-pull mechanism, the tree structure on which the push mechanism is implemented combines two ideas: good topological properties are guaranteed by means of prefixes based on peers identifiers (similarly to what is done in other structured P2P systems) and latency awareness is used to select a specific peer between those with the same prefix. Similar in spirit, but in a pure pull-based scenario, we propose in this paper an overlay topology design strategy that, taking into account latency and peer heterogeneity, aims at creating an overlay with good properties and low chunk delivery delays.

Many solutions implementing *network awareness* in P2P file-sharing systems have been proposed [10], [17], but at best of our knowledge, none of them have tried to apply it to peer-to-peer streaming systems. Our work shows that in P2P-TV systems localizing traffic actually improves network friendliness like shown in [11], but focusing also on users' side, we show that not only quality of experience of users is not reduced due to localization, but it is even improved.

In [16], [8] measurements about peers *quality of experience* are reported in terms of chunk delivery delays and loss probability. A better metric is represented by PSNR measure; indeed since real video sequences consist in many frames, each with different importance and variable size, we choose PSNR as index to represent the perceived quality by system users like authors of [12], [13] did, but we analyzed many sequences with various features (see Table III) in order to prove that proposed algorithm is effective independently on the type of video the source is going to broadcast.

III. System Description

We consider a chunk-based system in which a single source encodes and seeds the content. Let $\mathcal{N}$ be the set of peers, with cardinality N. Since the application satisfies near real-time constraints, every generated chunk must be received within a deadline from the moment it is emitted by the source; let this deadline be the *playout delay*, D_{max}. After the playout delay expires, the chunk is not traded anymore.

Chunks are exchanged among peers that are neighbors to each other. The overlay topology determines the structure of the peer neighborhoods and can be represented as an undirected graph $G(V, E)$, where $(p, q) \in E$ if and only if $p \in \mathcal{N}$ and $q \in \mathcal{N}$ are *neighbors* to each other. The overlay topology evolves dynamically for effect of the peer churning and the possible dynamic algorithms driving its maintenance and optimization [8]. Its dynamics are usually much slower than chunk distribution dynamics (minutes versus tens/hundreds of ms). From an abstract point of view, the overlay can be obtained by assigning a set of neighbors to every peer p. In practice, this set can be either assigned by a central authority (a tracker as in BitTorrent), or be obtained through some distributed gossiping algorithm that allows peers to discover other peers. In this paper, we consider peers *upload bandwidth* and *latencies* between peers as parameters that are used for the overlay topology design. Both parameters are easy

$\mathcal{N}$	Set of peers; $\|\mathcal{N}\| = N$
$\mathcal{K}_p$	Set of p selected neighbors; $\|\mathcal{K}_p\| = K_p$
$\mathcal{C}_p$	Set of p total neighbors; $\|\mathcal{C}_p\| = C_p$
$\mathcal{N}_p$	Set of neighbors p sends offers to; $\|\mathcal{N}_p\| = N_p$
B_p	Peer p upload capacity
l_{pq}	Latency between peer p and q
D_{max}	Playout delay
L	Chunk size
r_s	Average video rate
ρ	Network load
α	Probability of selecting a peer at random
γ	Bandwidth preference index
T	Latency preference index
δ	Topology scaling factor - relates K_p to B_p
β	Signaling scaling factor - relates N_p to B_p
λ_0	Average offer rate
M	Number of chunks a receiver can select per offer

Fig. 1. Notation.

to be measured, e.g., using some collaborative approach as recently proposed in [18]. Central authorities that support the overlay creation and maintenance as the one proposed by the IETF [17] are instrumental and compatible with this scenario. Alternatively, in a fully distributed way, peers can actively measure latency by means of probe packets, and then evolve their neighborhood as in [8].

Similarly to previous studies, in this paper, we consider scenarios in which peers upload bandwidth constitutes the bottleneck to system performance. This is justified by the fact that the majority of residential peers, which are expected to constitute the dominant component of the audience, is connected to the network through ADSL lines, which provide much higher download rates than upload rates.

IV. Design Choices

In this section, we discuss our main design choices for the overlay topology creation and scheduling algorithm. To help the reader, Fig. 1 summarizes the notation used in the following.

A. Overlay topology design

In presence of non negligible end-to-end latencies, the overlay topology design should take into account three different aspects. First, peers with short end-to-end latencies should be connected with each other, so that the delay introduced by scheduling control feedback is marginal. Second, and opposite to before, random choices are preferable since deterministic choices based on the end-to-end latency lead to topologies with high clustering degree and poor diameter properties, i.e., the overlay resembles a planar graph whose diameter scales as the square root of the number of nodes. Third, as already shown in [4], high bandwidth peers should be highly connected to each other and should be pushed close to the source, so that they can effectively contribute to the chunk distribution.

Borrowing concepts from the random graph theory, the overlay topology design strategy that we propose exploits peer latency information joint with information about peer upload bandwidth to improve the system performance while maintaining good topological properties (i.e., a diameter that scales logarithmically with the number of nodes as in a regular tree).

The strategy we envision for the formation of the neighbor sets is the following. Denote by B_p the upload bandwidth of peer p and by l_{pq} the end-to-end latency between peers p and q. Let γ, T and α be appropriate values whose meaning and setting we discuss later on.

Given a peer p, all peers q such that $B_q \geq \gamma B_p$ and $l_{pq} < T$ are marked as p **desired peers**. *Every peer p is associated to a set of neighbors $\mathcal{K}_p$ of cardinality K_p composed of αK_p peers randomly selected and $(1-\alpha)K_p$ peers randomly selected within the set of p desired peers. The final neighbor set of p, denoted by $\mathcal{C}_p$, is completed by adding all the peers q such that $p \in \mathcal{K}_q$.*

Basically, the idea is to make each peer connect to a *fraction of "good" peers*, with high bandwidth and small latency, so as to improve performance and *a fraction of random peers*, so as to reduce clustering effects that degrade the overlay topological properties. The thresholds γ and T define the concept of "good" peers for p; their setting is not critical, since we just need to partition peers into two rough categories. For the same motivation, the eventual limited availability of B_q and of l_{pq} can be overcome by considering only the subset of peers that peer p is aware of.[1] We set T to half the mean end-to-end latencies, $T = E_q[l_{pq}]/2$, and $\gamma = 1/2$. The parameter α is instead the core of the strategy. Its impact and tuning are discussed in the next section; just note that $\alpha = 1.0$ corresponds to a pure random choice, i.e., to homogeneous random graphs typically considered in the previous literature.

The value of K_p (number of connected peers) should be related to p upload bandwidth. Indeed, only high bandwidth peers need large neighborhoods to effectively exploit their bandwidth. On the contrary, the maintenance of large neighborhoods for low bandwidth peers has the only effect to increase the bandwidth waste due to signaling. We consider then two policies:

- *Fixed K_p*: the value K_p is the same for all peers, this policy is for comparison purposes.
- *Variable K_p*: set

$$K_p = \max\left(3, \lceil \delta B_p / r_s \rceil\right), \tag{1}$$

  where r_s is the average video rate. In this policy, K_p is proportional to the peer contribution factor B_p/r_s.

In the Variable K_p policy, the minimum value of 3 guarantees connectivity to very low bandwidth peers, while $\delta \geq 1$ is a topology scaling factor, that allows to grow a peer neighborhood to better exploit its upload bandwidth. In the following, we always use $\delta = 10$; its choice is, however, not critical.

[1] In this paper, we focus on the strategies for the overlay creation and on their impact on the system performance, while we leave the actual implementation as future work.

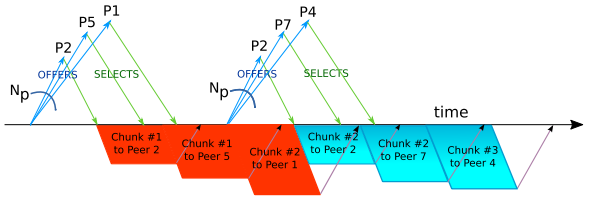

Fig. 2. Schematic representation of the peer chunk trading mechanism.

Finally, notice that the actual neighborhood of peer p is the union of the set $\mathcal{K}_p$ and of the set of peers q that contacted p; i.e., those peers q that chose p as a neighbor in their set $\mathcal{K}_q$.

B. Chunk Scheduling

We focus on a *pull* mechanism similar to the one used in other mesh based P2P-TV systems, [15], [19], [20]. A chunk is delivered from peer p to q after a negotiation phase during which p and q exchange information about the chunks they possess. The negotiation phase is sketched in Fig. 2; in the figure, signaling exchanges are represented above the time line, and data transmissions are below it. We assume that the negotiation is initiated on the transmission side. Peer p periodically selects a subset of its neighbors $\mathcal{N}_p \subset \mathcal{C}_p$ (with $|\mathcal{N}_p| = N_p$) to which offer chunks. p sends to all peers in $\mathcal{N}_p$ an *offer* message containing its buffer map (i.e., the list of chunks that it possesses and whose age is smaller than the playout delay D_{max}). Every peer in $\mathcal{N}_p$ replies using a *select* message with the indication of a set of at most M desired chunks. As soon as p receives a select message, it schedules the transmission of all requested chunks, maintaining a *transmission queue* of chunks. Finally, ACK messages are sent back by receivers to signal the end of the reception of each received chunk.

Several design choices can have impact on the performance of the scheduling algorithm, in particular: 1) the frequency at which p offers chunks to its neighbors, and the values of the parameters M and N_p; 2) the criterion to select peers belonging to $\mathcal{N}_p$; 3) the strategy according to which peers in $\mathcal{N}_p$ select chunks to download. In what follows we discuss these choices.

1) Parameter setting: To limit the chunk delivery delay, the number of chunks that are enqueued for transmission must be kept as small as possible. This suggests two things: i) set $M = 1$ to avoid to enqueue multiple chunks directed to the same peer, and ii) make a new offer only when the number of pending chunks is virtually null. However, to avoid idle time until select messages are received, a new offer phase is anticipated by an amount of time equal to the average round-trip time, as shown in Fig.2.

Consider that $1/\lambda_o$ is the average time between two offers, and $B_p/(L\lambda_o)$ with L the average chunk size, is the number of chunks that can be transmitted between two consecutive offers. To effectively use the peer bandwidth, it must be:

$$N_p \geq \frac{B_p}{L\lambda_o} \tag{2}$$

i.e., the value of offers N_p needed to efficiently exploit peer p bandwidth is proportional to its upload bandwidth B_p. We choose

$$N_p = \min\left(K_p, \lceil \beta B_p / r_s \rceil\right) \tag{3}$$

being β the *signaling scaling factor* which takes into account the fact that some peers in $\mathcal{N}_p$ can be not interested in downloading any chunk from p, because they already possess all offered chunks. Impact of β is discussed later.

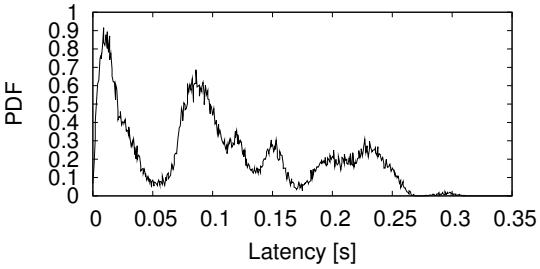

Fig. 3. Latency frequencies.

2) Peer Selection: The choice of the peers chunks should be offered to can follow different strategies. One simple strategy is to select peers in $\mathcal{N}_p$ uniformly at random [3] within the set of p neighbors, $\mathcal{C}_p$; another possibility is to associate to every peer in $\mathcal{C}_p$ a weight that depends on some physical attribute such as its upload bandwidth [4] or the latency between peers and thus selecting peers according to a weighted random choice. Since a smart peer selection policy has been already embedded in the system by means of the overlay topology design (according to which low latency and high bandwidth peers are preferred), we use a simple uniformly random peer choice at the scheduler. This choice is also justified by the intuition that the impact of explicit signaling and latency can cause undesirable drawbacks when biased peer selection mechanisms are in place. For example, if low bandwidth peers are served systematically after other peers, they might suffer from an offer rate which is too small to guarantee them to retrieve all the chunks.

3) Chunk Selection: For what concerns chunk selection at the receiver, two main strategies have been proposed and analyzed in idealized scenarios with negligible latencies [1], [3]: random chunk, and latest chunk (or its variants [5]). In this paper, we use the random chunk selection strategy for two reasons. First, our preliminary investigation has shown that, in scenarios with realistic latencies, random chunk selection policy is more robust to parameter settings and performs in general better than latest chunk selection policy. This is confirmed also by most of the implemented solutions, which rely on random selection [15], [21]. Second, as already discussed, we propose to exploit peer attributes during the overlay construction and to keep the algorithms involved in the chunk diffusion process as simple as possible.

V. Performance Evaluation Model

In this section we first describe the simulation scenario, and then present a thorough performance evaluation. All results have been obtained using simulation. Indeed, the complexity of the system makes it almost impossible to consider analytical

TABLE I
PERCENTAGE OF PEERS PER CLASS FOR DIFFERENT SCENARIOS.

Class	1	2	3	4
$H = 0.01$	1	76.7	2.3	20
$H = 0.05$	5	58.5	16.5	20
$H = 0.10$	10	35.8	34.2	20
$H = 0.15$	15	13.2	51.8	20

modeling methodologies, while experimental evaluation makes it hard to have full control on the system parameters. In addition, experimental test-bed cannot easily scale with the number of peers. All results are obtained with *P2PTV-sim*, an open source event driven simulator developed within the *Napa-Wine* [14] project [2] Results presented here amount to approximately 50 days of CPU time.

A. Network Scenario and Assumptions

Peers are partitioned in four classes according to their upload bandwidth:

- Class 1: B1 = 5Mb/s $\pm$ 10%
- Class 2: B2 = 1.6Mb/s $\pm$ 10%
- Class 3: B3 = 0.64Mb/s $\pm$ 10%
- Class 4: B4 = 0Mb/s, i.e., free riders.

We consider four different bandwidth scenarios. Let the parameter H denote the fraction of large bandwidth peers (Class 1 peers). The higher H is, the larger the overall bandwidth heterogeneity will be. The fraction of free-riders (Class 4 peers) is constant and equal to 20%. Fraction of Class 2 and Class 3 peers is then derived by imposing that the system-wide average upload bandwidth is $E[B_p] = 1.3$Mb/s in all cases. Table I reports for every scenario the distribution of peers.

The transport network is supposed to be transparent: it introduces a delay equal to the latency l_{pq} to all the datagrams traversing path from p to q. Latencies l_{pq}, assumed to be measured at application layer, are proportional to the geodetical distance between peers. Peers are distributed over the Earth surface according to a synthetic model that emulates the distribution of the Internet user population (as given by [22]). In particular, peers are scattered over seven domains representing continental/sub-continental geographical regions: Asia (Far East), Europe, Africa, Middle East, Oceania, North America and South America.

Each domain is modeled with disks placed in the center of mass of the relative geographical region (through latitude and longitude coordinates). Their radiuses are differently calibrated to match the extension of the corresponding geographical region. The resulting empirical pdf of latencies is reported in Fig. 3, which shows the clustering effect of peers within the same region, and belonging to different regions. The mean latency is $E[l_{pq}] = 96$ms.

For the results presented in the following we consider scenarios comprising $N = 2000$ peers, if not otherwise indicated. All results are averaged over at least eight independent simulation runs.

[2] *P2PTV-sim* is available at http://www.napa-wine.eu/cgi-bin/twiki/view/Public/P2PTVSim.

TABLE II
PARAMETERS OF THE VIDEO SEQUENCES.

	Length	Spatial Res.	Frame/sec
Pink	40s	352×240	25
Paris	33.3s	352×288	30
Foreman	40s	352×288	25

B. Video Parameters

In most of in the P2P-TV literature so far, a very simplistic synthetic model for the video stream has been adopted according to which the source generates fixed length chunks at a fixed rate. However, compressed video streams are known to exhibit variable and bursty encoding rate, and this can deeply impair the system. Similarly, frames have different importance in the sequence. For example, an "intra" frames carry very valuable information (and are therefore bigger), while "inter" frames carry only differential information (and are much smaller). Furthermore, the paramount performance index that has been considered for video streaming application should be the actual Quality of Experience, which can only be evaluated including the video coding and decoding processes in the simulation. In other words the loss of a frame can have a very variable impact on the QoE, depending on the type of the frame itself. In this paper we therefore explicitly model the streaming of actual video streams over the P2P-TV system, and evaluate the system performance using a direct measurement of the QoE users get.

Three well-known video sequences are considered as benchmarks: *Foreman* and *Paris* and *Pink of the Aerosmith*. Table II reports the encoding details for the three sequences. We selected the H.264/AVC protocol for encoding video sequences. A hierarchical type-B frames prediction scheme has been used, obtaining 4 different kinds of frames that, in order of importance, are: IDR, P, B and b. GOP structure has been set to IDR x 8 {P,B,b,b}, which can however be violated if the encoder detects a sudden scene change that forces the insertion of a IDR frame. Note that the video consists of 1000 frames in all cases, which however corresponds to less than 40s of visualization. This supports the assumption that peer churning can be neglected since it typically involves much larger time scales. The nominal video rate of the encoder r_s is a free parameter that we vary to enforce different values of the system load defined as $\rho = r_s/E[B_p]$.

C. Chunk injection - mapping frames into chunks

Given the highly structured video stream organization, a natural question is how to chop the video data into chunks. An intuitive approach is the simple *one to one* mapping, according to which every chunk contains exactly one video frame. The source node then generates a new chunk at regular time, i.e., every new frame. This mapping scheme minimizes the packetization delay at the source, thus allowing a stricter real-time streaming. Furthermore, the rounding at frame boundaries minimizes the impact of chunk losses, avoiding that a corruption of the video stream propagates to the following frames due to partial delivery of information, e.g, missing headers. Finally,

TABLE III
CHARACTERISTICS OF THE ENCODED VIDEO SEQUENCES.

	Pink			Paris			Foreman		
Frame type	Num	Avg. size [kb]	Std. dev	Num	Avg. size [kb]	Std. dev	Num	Avg. size [kb]	Std. dev
IDR+I	40	79.2	6.43	32	294.4	32	32	207.8	47
P	285	57.0	17.10	250	63.6	10	256	82.7	16
B	225	38.2	15.3	250	27.3	6	245	38.4	10
b	450	27.6	21.7	468	16.2	5	467	22.2	7

this simple mapping allows to easily exploit the information about the frame type at the scheduler level.

A limit of this scheme is that it is not possible to control the chunk size. As a result, the chunk distribution process can be significantly affected by the highly variable chunk sizes. As an example, Table III reports the frame size statistics for the three video benchmarks when $r_s = 1Mb/s$. More sophisticated mapping schemes in which several frames of the same type can be carried within the same chunk, as well as approaches according to which "large" frames are split into several chunks are left for future investigation.

D. Assessing video quality at the receiver

To assess QoE at the receiver, P2P-TV performance should not be evaluated by considering networking indices, such as the chunk delivery delay and chunk loss probability traditionally taken into account by previous work. The effect of loss pattern on video quality integrity must be taken into account since the same average chunk loss probability might induce very different effects on the quality of the reconstructed video.

To assess video quality at the receiver we use the classical Peak Signal-to-Noise Ratio (PSNR) metric [12], [13]. PSNR is a widely adopted objective video quality index that provides the mean square error between the original video and the received one. Note that the PSNR scale is logarithmic in dB, so that a difference of 2dB corresponds to a very large improvement of the QoE. For example, from Fig. 12 doubling the encoder rate from $r_s = 780$kb/s to $r_s = 1410$kb/s improves the PSNR by 2.5dB only (from 42.91dB to 45.45dB).

For each peer, the computation of the PSNR is done on a frame by frame basis, comparing the original image to the one that is reconstructed at the receiver. The obtained values are then averaged over all frames, obtaining the "PSNR per peer". Finally, the "overall average PSNR" is obtained by averaging over all peers the PSNR per peer. We notice that the computation of the PSNR cannot be done in correspondence of a missing frame. In this latter case, we assume that the receiver uses the last correctly decoded frame as reference to compute the PSNR.

To compare the performance of the location/bandwidth-aware and the unaware policies, during each simulation the original uncompressed video sequence is encoded using the selected video rate r_s. Frames are then mapped into chunks, which are traded by the P2P-TV system, being the transmission time of a chunk computed based on actual chunk length. Given the sequence made up of received chunks at the peer, the received video is rebuilt and the PSNR is evaluated comparing it with the original and uncompressed video. Notice that the final PSNR includes therefore both the effect of the encoder and of the possible chunk loss occurred during the P2P-TV distribution.

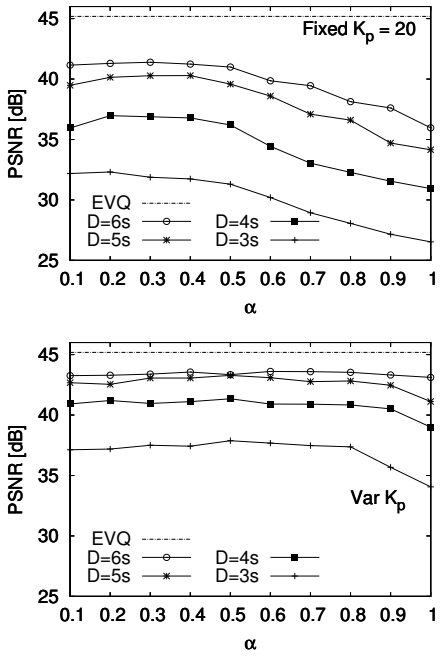

Fig. 4. Average PSNR versus α for different values of the playout delay with $\rho = 0.9$ and $H = 0.10$.

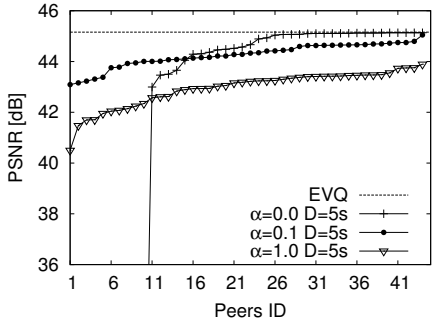

Fig. 5. PSNR for different peers with $\rho = 0.9$ and $H = 0.10$.

VI. RESULTS

First, we wish to assess the effectiveness of the proposed latency/bandwidth-aware overlay topology design. Fig. 4 reports the average perceived PSNR versus the overlay construction parameter α, for different choices of the playout buffer. The simulations refer to a network scenario with $H = 0.10$. The *Pink* video sequence encoded at rate $r_s = 1168$kb/s, so that the average offered load is $\rho = 0.9$.

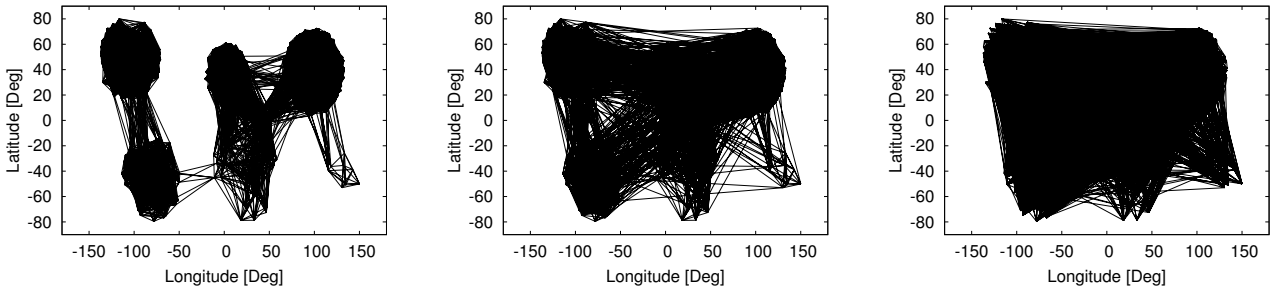

Fig. 6. Overlay topology graph for $\alpha = 0.0$, $\alpha = 0.1$ and $\alpha = 1.0$.

The top plot refers to the case in which the number of neighbors is fixed and equal to $K_p = 20$ for every peer in the network; that translates into actual neighborhood sizes that are on average equal to 40 (remind that besides choosing K_p neighbors, p can be chosen by the other peers as a neighbor). The bottom plot, instead, refers to the case in which K_p is set based on the bandwidth as in (1) with $\delta = 10$. In this case, the average neighborhood size results equal to 28.3.

In the following, we report also Encoded Video Quality (EVQ) at the source as reference. The effects introduced by the distribution system on the perceived quality can be grasped by comparing the received PSNR with the PSNR at the source.

A. Impact of α

The parameter α (that represents the randomness in choosing neighbors) has an impact on the video perceived by users. The performance of the system is poor for $\alpha = 0.0$, i.e., when all the peers in $\mathcal{K}_p$ are chosen among the p desired peers (this case is not reported in the plots because it causes a too large y-axis range, see Fig. 5 for a detailed comparison). Indeed, as expected, the topological properties of the overlay are poor when all the neighbors are selected based on a notion of distance, leading to possibly even disconnected topologies. However, this effect disappears as some randomness is introduced in the overlay design, i.e., for $\alpha \in [0.1, 0.2]$. Furthermore, values of α closer to 1 worsen the performance for effects of the larger latencies among neighbor nodes, especially in the case of fix K_p and/or when a tight play-out buffer is enforced. Recall that the PSNR metric is very sensitive; thus, also few dB of improvement can represent a significant QoE gain for the users. Finally, comparing the two plots it is possible to infer that the choice of adapting K_p the peer upload bandwidth leads to significant QoE improvement (in this section we set β equal to 2 and leave its analysis to the following section), for every α and $D_{\max}$. This confirms that exploiting upload capacity of high-bandwidth peers plays a key role as already shown in [4].

To give more insights, Fig. 5 provides a direct comparison of the peer PSNR for 44 randomly selected peers (11 peers per each bandwidth class). The figure reports results for the cases $\alpha = 0.0$, $\alpha = 0.1$ and $\alpha = 1.0$ with $D_{max} = 5$s. A proper choice of α shows that the QoE improves uniformly for all considered peers. It is worth to underline that $\alpha = 0.0$ shows the effect of a almost disconnected topology: some peers can improve their QoE, while other can hardly enjoy any video, driving the overall average PSNR to very small values.

To give the intuition of the effects of α on the overlay, a graphical representation of the topologies for three different values of $\alpha = 0, 0.1, 1$ is reported in Fig. 6. Notice that by choosing small values of α we effectively make neighborhoods more and more local, while $\alpha = 1.0$ gives a completely random graph. Again, the extreme choice of $\alpha = 0.0$, leads to overlay topologies with very few "intercontinental" logical links which result in bad topological properties, up to even disconnected graphs. Chunk propagation time increases a lot then, and QoE is severely impaired.

Turning our attention on the underlying transport network cost, Fig. 7 reports the *network stress*, i.e., the average distance covered by chunks expressed it terms of the corresponding latency. This metric represents the amount of transmission resources demanded to the underlying transport network to sustain the chunk diffusion process. In other terms, this can be taken as a metric of cost paid by the transport network for carrying traffic related to the P2P-TV service.

The network stress is monotonic increasing and almost linear with respect to α, while little impact is seen for different values of D_{max} and K_p (not reported). By decreasing α, i.e., better localizing neighborhoods, the stress for the underlying network is reduced. There is a factor 4.5 between the highest clusterization ($\alpha = 0.0$) and the random neighborhoods case ($\alpha = 1.0$).

A different but likewise important effect induced by α can be grasped by looking at Fig. 8. By reducing α, we obtain overlay topologies in which the peers are more clusterized based on their upload capacity. This leads to a more efficient exploitation of the upload bandwidth of high bandwidth peers, which is beneficial for the whole system performance. However, when $\alpha = 0.0$ low and high bandwidth peers tend to be separately clustered. Low bandwidth peers are ten severely penalized, since they tend to not receive the video-stream, while high bandwidth peers load decreases at the same time.

For completeness, in Figs. 9 and 10 we respectively explore the effect of α in different bandwidth scenarios and for different video sequences. Curves of Figs. 9 and 10 refer to the case of variable K_p with play-out delay $D_{\max} = 5$ s. In all cases performance consistently improves by reducing α.

Given these results, a choice of small values of α is highly supported. In the following, we choose $\alpha = 0.1$ as a good

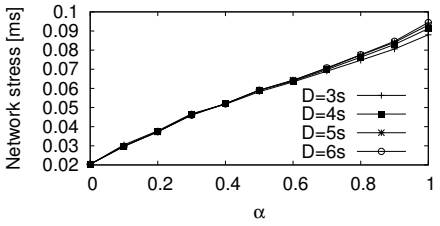

Fig. 7. Network stress versus α, for different topologies and values of the playout delay, $\rho = 0.9$.

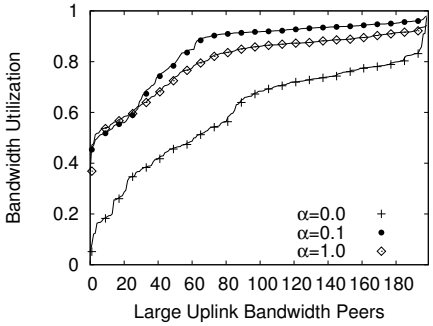

Fig. 8. High class peer bandwidth utilization for value of $\alpha = 0.1, 0.1, 1.0$.

tradeoff between overlay robustness and performance benefits.

At last Fig. 11 reports the overall PSNR versus D_{max}, considering a scenario with $N = 2000$ and $N = 7000$ peers. Results show the beneficial impact of reducing α, which is consistent for several playout delay. As expected, to achieve similar PSNR performance, the playout delay needs to be increases when the number of peers in the system increases. This is due to the larger overlay topology diameter (recall that the number of hops chunks have to be transmitted grows logarithmically with N), which translates into higher delay. Therefore, the benefit of a smart overlay design is higher for large N too.

B. Impact of r_s and β

We now turn our attention to the schedulers parameters. We consider only the location/bandwidth-aware policies with variable degree and $\alpha = 0.1$, since these choices achieve globally the best performance.

Fig. 12 reports the received average PSNR versus the network load ρ (on the bottom x-axis), i.e., for the video source rate r_s on the top x-axis). Different values of β are shown, to calibrate in (3) the number N_p of neighbors an offer message is sent to. First, notice that the PSNR increases initially for increasing video rate, reflecting the higher quality of the encoded stream (coding gain). However, for $r_s \geq 1.03$Mb/s (i.e., $\rho \geq 0.8$), dramatic performance degradations are suffered: the increased system load causes larger chunk loss rates, translating the coding gain into worse QoE. Increasing the playout delay has little effect on this (not reported here due to lack of space).

Considering the impact of β on the system performance, we notice that too small values of β limit the high bandwidth peers ability to exploit their upload capacity, causing undesirable chunk losses independent of the playout delay which is set

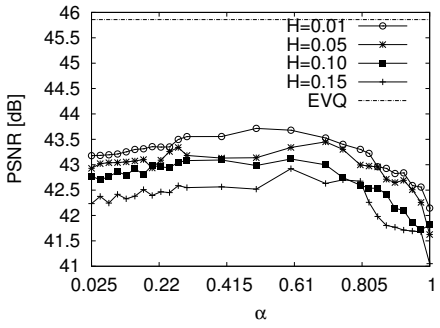

Fig. 9. Average PSNR versus α for different values of H with $D_{max} = 5$s.

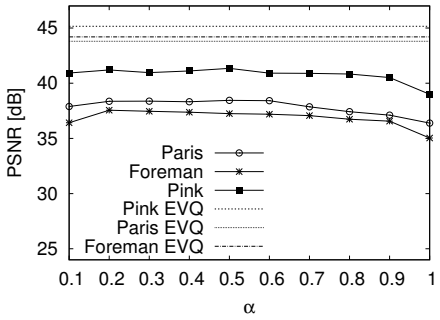

Fig. 10. Average PSNR versus α for different video sequences with $\rho = 0.9$ and $H = 0.10$.

$D_{max} = 5$s. Too large values of β make on the contrary chunk delivery delay increase due queuing at the transmitter peer, causing losses due to inadequate playout delay. In our setting, the best choice of β is 2, corresponding to $N_p = 10$ for high bandwidth peers; note (by contrast with r_s) that the sensitiveness to parameter β is rather limited.

To conclude this section Fig. 13 reports the temporal behavior of the PSNR (which we recall is computed frame by frame) for a random peer. A network load $\rho = 0.6$ and $\rho = 1.0$ are represented on the top and bottom plot respectively. Observe that for $\rho = 0.6$ limited fluctuations of the PSNR are observed. These fluctuations are intrinsically generated by the encoding process and not to chunk loss. For $\rho = 1.0$, instead, fluctuations of the PSNR become really huge, for effect of chunk loss. The loss of a precious chunk, indeed, cause a dramatic reduction of the video quality for a period of time which can be large, i.e., up to a GOP in the case of intra frames. The large video streaming PSNR cannot compensate the loss of frames, so that the average QoE dramatically drops.

VII. Video-Aware Schedulers

Fig. 13 shows clearly that different frame types have different importance in a video stream, and a frame loss event may cause very different levels of degradation of the reconstructed video quality. For instance, a missing intra frame (IDR or I) impairs the video decoding until the next intra frame is received, i.e., for a GOP, which corresponds to more than 1s of video. On the contrary, a missing inter (b) frame impairs only the decoding of one single frame. Therefore, by modifying the scheduler in such a way that it assigns some form of priority to chunks transporting most precious frames,

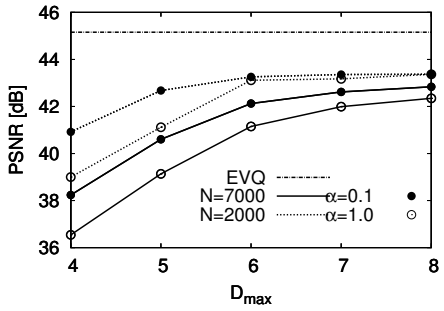

Fig. 11. Average PSNR for different values of the playout delay and number of peers with $\rho = 0.9$ and $H = 0.10$.

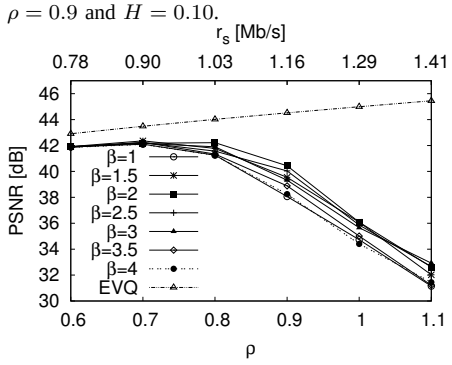

Fig. 12. Average PSNR versus ρ the for different values of β with $D_{max} = 5$s, $\alpha = 0.1$. The corresponding video rate r_s is reported on the top x-axis.

the overall system performance might improve, especially in presence of significant losses.

In this section, we investigate this issue and discuss how this simple basic idea can be exploited at the scheduler level to define a class of video aware scheduling policies. To the best of our knowledge, only the authors of [12], [13] have proposed a video-aware scheme for P2P-TV systems. Their work however is targeted to tree and multi-tree based P2P streaming systems, which exhibit significant differences from our mesh/pull based systems. According to their scheme, priority is assigned to frames based on the amount of video degradation they might induce if lost. This degradation, in its turn, is evaluated for a given frame in terms of the number of subsequent frames that would be affected by the lost of that frame.

We define a class of video-aware scheduling policies according to which the receiver selects the chunk to download according to a preference. A weight q^ω is assigned to every chunk encapsulating a frame; then, the probability of selecting a chunk is proportional to the corresponding chunk weight. We consider the same q as suggested in [12], [13]: q is equal to the number of frames that will be affected by its loss.

To generalize the priority scheme, the exponent ω provides a free parameter to better calibrate weights. With $\omega = 0$ we obtain the random chunk selection strategy, all chunk weights equal to 1; with $\omega = 1$, we obtain the same values as in [12], [13]; with $\omega > 1$ we assign a further importance to more important frames, while for $\omega < 1$ we reduce the effect of importance.

Fig. 14 reports, for the three video sequences, the individual

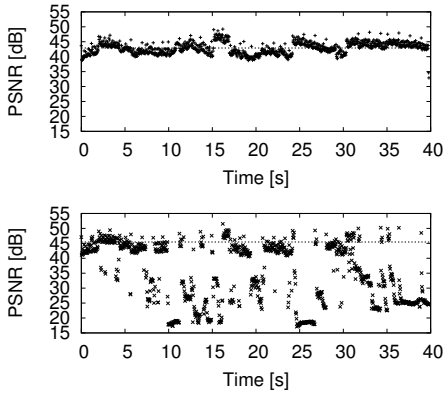

Fig. 13. PSNR variation of a random peer versus time. $\rho = 0.6$ ($r_s = 780$kb/s) and $\rho = 1.0$ ($r_s = 1290$kb/s) in the top and bottom plot, respectively.

PSNR of 44 randomly selected peers (again choosing 11 peers per class) for 4 different choices of ω. The plots refer to a case in which $\alpha = 0.1$ and $r_s = 1.41Mb/s$ so to result in an average network load $\rho = 1.1$. The choice of $\rho = 1.1$ has been done to make the system work in a high loss regime, where we expect that video aware schemes can be more effective. Notice that, in realistic situations, system bandwidth in P2P systems is not known and may also fluctuate over time for effect to peer churning. This implies that periods of time in which the system is overloaded are likely in P2P systems.

Observe that for all the three sequences, video aware scheduling algorithms guarantee slightly better performance, especially for peers which are in unfavorable conditions (i.e., those experiencing largest losses). This because video aware schedulers increase the speed at which precious chunks are distributed by the system increasing the chances for peers in unfavorable conditions to collect them. However, observe that the gain margin is rather limited and may not apply to all the peers; furthermore results exhibit a rather strong dependence on the video sequence (for example the optimal choice of ω depends on the sequence). This is mainly due to the fact that significantly different chunk size distributions correspond to the different sequences as shown in Table III; thus the effects on the distribution process of chunk priorization may be significantly different. The gain margins are higher and more uniform over the peers for the sequence *Pink*, in which the average difference in size of different frame types are less pronounced.

Reducing the system load to $\rho \in [0.8, 0.9]$ (we do not report corresponding results for lack of space) the performance improvement of video aware schedulers tends to vanish for the sequences *Foreman* and *Paris*; while it is still appreciable for the sequence *Pink*. Furthermore, for the first sequences, a too pronounced choice of weights (ω too large) can lead to a slight degradation of the overall performance. In some cases, indeed, low priorities chunks can result too penalized, being distributed to only a very limited set of peers. In different words, schemes adopting too pronounced weights at the scheduler level tend to reduce the diversity of chunks that

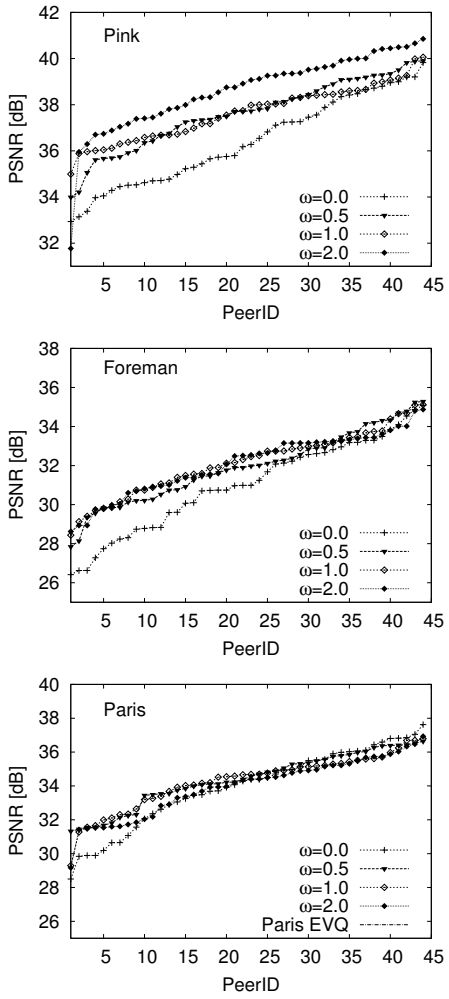

Fig. 14. PSNR for different Peers and values of ω with $\rho = 1.1$, $D_{max} = 5s$ and $H = 0.10$. Pink, Foreman and Paris video sequences from top to bottom.

are simultaneously being distributed in system, which is key ingredient to make it possible to efficiently exploit the system bandwidth in mesh/pull based systems.

In conclusion, while the adoption of video-aware priority schemes can be really effective in tree based systems, as previously shown in [12], [13], in mesh based systems its effectiveness is smaller, but still interesting, especially under overloaded conditions.

VIII. Conclusions

In this paper, we have analyzed, through an extensive simulation campaign, the performance of pull P2P-TV systems in scenarios where peer bandwidth heterogeneity, network latencies and properties of encoded video streams are accounted for. We have provided simple guidelines for the design of the overlay topology and the chunk scheduling algorithm. Our main findings are the following.

- By carefully designing the overlay topology it is possible to achieve the twofold goal of partially localizing the traffic while improving the user QoE.
- By prioritizing chunks that encapsulate more valuable pieces of information at the scheduler level, system performance can be slightly improved in overloaded conditions.

Acknowledgment

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] L. Massoulié, A. Twigg, C. Gkantsidis, and P. Rodriguez, "Randomized decentralized broadcasting algorithms," in *IEEE INFOCOM*, Anchorage, Alaska, USA, May 2007.
[2] S. Sanghavi, B. Hajek, and L. Massoulié, "Gossiping with multiple messages," in *IEEE INFOCOM*, Anchorage, Alaska, USA, May 2007.
[3] T. Bonald, L. Massoulié, F. Mathieu, D. Perino, and A. Twigg, "Epidemic live streaming: optimal performance trade-offs." in *SIGMETRICS*, Annapolis, Maryland, USA, June 2008.
[4] A. C. da Silva, E. Leonardi, M. Mellia, and M. Meo, "A bandwidth-aware scheduling strategy for P2P-TV systems," in *IEEE P2P*, Aachen, Germany, September 2008.
[5] L. Abeni, C. Kiraly, and R. Lo Cigno, "On the optimal scheduling of streaming applications in unstructured meshes," in *IFIP Networking*, Aachen, Germany, May 2009.
[6] D. Ren, Y. T. H. Li, and S. H. G. Chan, "On reducing mesh delay for peer-to-peer live streaming," in *IEEE INFOCOM*, Phoenix, Arizona, USA, April 2008.
[7] T. Small, B. Liang, and B. Li, "Scaling laws and tradeoffs in Peer-to-Peer live multimedia streaming," in *ACM Multimedia*, Santa Barbara, CA, USA, October 2006.
[8] R. Lobb, A. P. Couto da Silva, E. Leonardi, M. Mellia, and M. Meo, "Adaptive overlay topology for mesh-based p2p-tv systems," in *ACM NOSSDAV*, Williamsburg, Virginia, USA, June 2009.
[9] T. Locher, R. Meier, S. Schmid, and R. Wattenhofer, "Push-to-pull peer-to-peer live streaming," in *Lecture notes in computer science*, Berlin, Germany, 2007.
[10] H. Xie, R. Y. Yang, A. Krishnamurthy, Y. G. Liu, and A. Silberschatz, "P4P: provider portal for applications," in *SIGCOMM*, Seattle, WA, USA, August 2008.
[11] Z. Shen and R. Zimmermann, "Isp-friendly peer selection in p2p networks," in *ACM Multimedia*, Beijing, China, October 2009.
[12] E. Setton, J. Noh, and B. Girod, "Low latency video streaming over peer-to-peer networks," in *IEEE ICME*, Toronto, Canada, July 2006.
[13] ——, "Congestion-distortion optimized peer-to-peer video streaming," in *IEEE ICIP*, Atlanta, Georgia, USA, October 2006.
[14] [Online]. Available: http://www.napa-wine.eu
[15] M. Zhang, L. Zhao, Y. Tang, J. Luo, and S. Yang, "Large-scale live media streaming over Peer-to-Peer networks through global Internet," in *Workshop on advances in Peer-to-Peer multimedia streaming*, Singapore, November 2005.
[16] Y. Liu, "On the minimum delay peer-to-peer video streaming: how realtime can it be?" in *ACM Multimedia*, Augsburg, Germany, September 2007.
[17] J. Seedorf, S. Kiesel, and M. Stiemerling, "Traffic localization for p2p-applications: The alto approach," in *IEEE P2P*, Seattle, WA, USA, September 2009.
[18] J. Douceur, J. Mickens, T. Moscibroda, and D. Panigrahi, "Collaborative measurements of upload speeds in p2p systems," in *IEEE INFOCOM*, San Diego, CA, March 2010.
[19] X. Zhang, J. Liu, and T. Yum, "Coolstreaming/donet: A data-driven overlay network for peer-to-peer live media streaming," in *IEEE INFOCOM*, Miami, Florida, USA, March 2005.
[20] F. Picconi and L. Massoulié, "Is there a future for mesh-based live video streaming?" in *IEEE P2P*, Aachen, Germany, September 2008.
[21] V. Pai, K. Kumar, K. Tamilmani, V. Sambamurthy, and A. E. Mohr, "Chainsaw: Eliminating trees from overlay multicast," in IPTPS, Ithaca, NY, USA, February 2005.
[22] [Online]. Available: http://www.internetworldstats.com/stats.htm