Successfully Mapping DASH Over a P2P Live Streaming Architecture

Laura Natali and Maria Luisa Merani, *Senior Member, IEEE*

***Abstract*— This paper proposes to combine peer-to-peer (P2P) and dynamic adaptive streaming over HyperText Transfer Protocol (HTTP) (DASH), leveraging the scalability and self-organization properties of the former and the capability of the latter to adapt the rate of the delivered video to varying operating conditions. The devised P2P-DASH architecture is a multioverlay P2P platform, where every peer implements a decentralized rate control strategy that steers its transitions from one overlay to another, so as to achieve a good viewing quality and also preserve the good functioning of the entire system. At peer's site, the new architecture displays the following salient features: 1) an augmented pool of neighbors, to provide every node with the knowledge of the streaming process within adjacent overlays; 2) a priority mechanism to speed up the delivery of video chunks to peers that newly enter the system or switch from the current to a new overlay, in order to quickly turn them into active peers; and 3) some alternative rate control algorithms to guarantee, with different degrees, more resources to overlays that deliver higher streaming rates, therefore reducing the misalignment among streaming processes which occur in different overlays, and ultimately improve the viewing quality of the single user. System behavior is extensively investigated through simulation, and some clear design guidelines are provided.**

***Index Terms*— Dynamic adaptive streaming over HTTP (DASH), peer-to-peer (P2P), switching delay, video streaming.**

I. Introduction

THE usage of video as the primary communication form over the Internet is a constantly growing experience for the majority of contemporary societies: the trend manifests itself through several emblematic service offerings, some of which are very recent and pervasive. YouTube and NetFlix, the most popular Web platforms for sharing videos and distributing movies and television programs, respectively, are by now the forerunners of the phenomenon, which is unceasingly enriched by new applications and features. Periscope and Meerkat are offering their users a way to directly stream videos from the smartphone and likewise to follow videos being streamed by others; even Facebook is introducing video as a profile picture. This steep increase in multimedia content fruition has a profound impact on the amount of traffic that revolves around Internet video servers. YouTube and Netflix consume about 50% of North American downstream bandwidth during peak hours and 30% in Europe [1], [2], and this poses powerful challenges on the quality of the offered services, also threatened by the varying traffic conditions in the underlying network infrastructure. In parallel, content delivery networks have boomed, and one of their transversal features is to be heavily based on the combined use of HTTP and Transmission Control Protocol (TCP). This entails several benefits, most notably, the employment of commodity servers and stateless services.

To cope with increasing video requests from the users, varying network conditions and to preserve HTTP servers via an inter-operable standardized approach, dynamic adaptive streaming over HTTP (DASH) has been developed [3], [4]. A DASH server typically encodes video content at multiple bit rates and provides a manifest file to describe the structure of the media segments and the different video versions that it makes available. It is up to the client to decide which version, also termed representation, to stream. The large diffusion of DASH is testified by its adoption from companies that are leaders in video on demand and video streaming and according to the survey in [5], most of the European broadcasters will provide DASH contents in the very near future too. Yet, servers and available bandwidth remain critical elements of such architectures: scalability and network resources emerge as the real challenges, which await for a sound solution. The peer-to-peer (P2P) paradigm, often put aside and discarded with conceit, can answer this question. This is the premise embraced by this paper, whose proposal rests on a multioverlay P2P architecture, where there are as many swarms as the number of available DASH representations, and every swarm is responsible for distributing one of them. Some alternative control strategies rule the transitions of the peers from one overlay to another. The control logic is implemented at the peer's site and hinges on locally measured parameters, as well as on macroscopic system indicators, to secure the quality that the single peer experiences, as well as the good functioning of the entire platform.

For the proposed architecture, the goal of this paper is multifaceted.

1) On the DASH side, given the system is thought for near-real-time services, the aim is to understand how and to what extent the flexibility that the DASH offers in switching from one representation to another, which can occur with a time granularity as low as the segment duration, plays in favor of a timely migration among overlays.
2) On the P2P side, the goal is to put forth and analyze the performance of various system solutions and of two different control algorithms steering the node

Manuscript received October 21, 2015; revised January 7, 2016; accepted February 26, 2016. Date of publication March 8, 2016; date of current version June 5, 2017. This paper was recommended by Associate Editor L. Zhou.

The authors are with the Engineering Department Enzo Ferrari, University of Modena and Reggio Emilia, Modena 41125, Italy (e-mail: laura.natali@unimore.it; marialuisa.merani@unimore.it).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSVT.2016.2539611

movements among overlays, so as to ensure that the video is streamed in the smoothest way and that the node experiences no interruptions when moving from one DASH representation to a different one. At the same time, the examined solutions strive to have the user view the video at a rate, which is as close as possible to the desired one, with a satisfying quality.

We propose a system that exhibits the following salient features: 1) an increased number of neighboring peers from different overlays to boost the exchange of buffer maps that consent the peer to know the streaming position of its target overlay in advance, as well as which video chunks are currently circulating in adjacent overlays; 2) the prioritized scheduling of video chunks by parent peers, which serve first the children peers that have just switched to a new overlay; and 3) two different flavors of the rate control algorithm, to either judiciously grant more resources to overlays distributing video at higher streaming rates or to privilege the user's satisfaction.

System behavior is extensively explored, placing an emphasis on the switching delay, i.e., the delay incurred by the peer when transiting from one DASH representation to another, and our major findings are summarized as follows.

1) The time granularity of DASH plays in favor of P2P, as it allows to reuse a significant fraction of the streaming buffer content when the peer migrates from one overlay to a new one.
2) The buffer reuse mechanism and the increased neighborhood positively combine to reduce the switching delay, to an extent that depends on the misalignment of the streaming processes among overlays.
3) The introduction of priority scheduling reduces the switching delay regardless of the alignment among overlays.
4) The rate control algorithm can be tuned so as to achieve a better alignment, hence a low switching delay, combined with good operating conditions, at the expense of a suboptimal streaming rate experienced by the peers.

In the remainder of this paper, Section II offers an overview of the related work. Section III delineates the essential DASH features and it is preparatory to Section IV that illustrates the proposed architecture resting on DASH and P2P, its constituent elements, and the merit figures examined to assess its performance. Section V discusses the numerical results, and Section VI draws the conclusion.

II. Related Work

As regards existing contributions in the literature, the bulk of the work on DASH focuses on client–server architectures, and in this setting, different algorithms are proposed to dynamically choose the most appropriate DASH representation, in order to cope with bandwidth variations and guarantee the final user the best quality of experience. Several rate adaptation techniques are based on the analysis of bandwidth variations and the streaming buffer level. Referring to bandwidth-based solutions, Li *et al.* [6] propose a strategy that relies upon the TCP download throughput of the client to determine the available bandwidth in the presence of congestion, whereas it constantly probes the network and adapts to its new conditions otherwise. De Cicco *et al.* [7] use feedback control theory to design an adaptive streaming algorithm with the objective to maximize the bit rate in the download and to guarantee a stable buffer level. Liu *et al.* [8] suggest a rate adaptation algorithm relying on the smoothed HTTP throughput to determine if the current media bit rate matches the end-to-end network bandwidth. In other cases, the rate adaptation algorithm is based on the status of the playback buffer, as in [9], that strives to guarantee the minimum buffer length. Still other solutions have been recently explored. In [10], fuzzy logic is the core of the rate controller. In contrast to most DASH solutions, Shuai *et al.* [11] implement a server-side open-loop rate control. Overall, whether the rate control is implemented at the client or server side, current conditions are monitored in order to maximize video quality, the focal point being the client perspective: the client is opportunistic in its decision taking process, as the aim is to optimize its viewing experience only. To the best of our knowledge, in the algorithms proposed in the literature, neither the user is aware of the implications that its local decision has on the entire system, nor it possesses any notion of friendliness toward concurrent connections, this task being left to TCP. Unlike the previous investigations, our proposal strives to guarantee a good performance to the user without loosing the overall system sight. The reason is that in a system built upon the notion of mutual cooperation, as any P2P architecture happens to be, the actions that the single peer performs immediately reflect on all other peers. It is, therefore, mandatory that in its decision-making process, the peer considers not only its perspective, but also a global prospect. Our former study in [12] already began moving along this path. In a multioverlay scenario, it put forth a rate control algorithm and adopted an integer linear programming approach to provide a benchmark to system performance, also demonstrating that the P2P users can experience a very satisfying performance, better than in a conventional P2P setting, where DASH is not employed. In [20], we performed an extensive numerical analysis to demonstrate the effectiveness of the envisioned solution, even in the critical occurrence of flash crowds, finding that they are rapidly detected and the peers steered to the most appropriate overlay.

In this paper, not only we perfect the formerly proposed algorithm, but also conceive innovative structural features to enhance system effectiveness.

There are some explicit contributions in the literature that jointly deal with P2P and DASH and that we deem worth being mentioned, namely [13]–[15]. Lederer *et al.* [13] propose a standard compliant solution to integrate peer-assisted streaming in the conventional DASH client–server systems and it represents a suitable approach in a Content Delivery Network type of environment. Unlike in this paper, the adaptation logic employed in [13] to modify the requested video rate solely serves the purpose of achieving a reduced system complexity and to guarantee satisfactory savings of server bandwidth. The system heavily relies upon the server presence, and the number of clients considered in the simulation setting is modest, of the order of magnitude of few tens. On the contrary, in this paper, the server contribution is minimal and we have several

thousands of nodes to accommodate. Accordingly, the rate control algorithm that we propose has to be carefully crafted. In [14], P2P is integrated in a personalized learning system, the basic idea being to exploit the cooperation among peers in those settings, where a given number of users request the same video content, as it is typically the case for a learning environment, such as a campus network. However, content is always made available by the server to a gateway, and P2P distribution is restricted to the local site, with a very low number of participating nodes being involved. In our proposal, no restriction is posed on the user location and no intermediary devices are foreseen. Tian *et al.* [15] apply DASH to a P2P architecture and exploit cooperation game theory to rule the node switching process among different representations. The authors present a cost-sharing mechanism that favors the cooperation among peers watching the same representation. Again, the server role is predominant, the size of the peers group is modest, and no attention is paid to the complete statistical characterization of system performance.

Overall, as regards the comparison between the current contribution and the correlated works existing in the literature, the most notable remarks are that we do not examine a Video on Demand (VoD) system whose population size is very limited, nor we interpret P2P as a secondary, although beneficial feature of the video distribution architecture. On the contrary, in order to tackle the scalability issue, we believe that a pure P2P solution is the real challenge. This paper, therefore, investigates the feasibility and the achievable performance of a system that delivers a live streaming channel to a large floor of users, with very little contribution from the server. Hence, we explore more sophisticated algorithms and system solutions, assessing system performance from numerous new facets.

III. DASH Fundamentals

DASH [3] is a standard that serves the purpose of delivering media content through the network via conventional HTTP servers to HTTP clients. As clients may request data from standard Web servers with no DASH-specific capabilities, the standard is centered on the data format and on the way to describe—alias present—the media content, as opposed to client–server actions.

In DASH, the media content is composed of a sequence of consecutive temporal periods, and each period in turn consists of multimedia components of different types, e.g., the audio component and the video component, that are typically available in different versions. As an example, there might be video versions exhibiting different resolutions and, in the case of prerecorded material, audio versions in different languages. Within a period, the multimedia content is arranged into adaptation sets, and each set describes a group of inter-changeable versions of one or several media content components that are termed representations. Clients can dynamically switch from one representation to another in response to, e.g., varying network conditions: if the switching takes place properly, it is experienced as seamless by the user.

Within every representation, content is further divided into segments that contain media data and/or metadata to access, decode, and play out the media content. Each segment is made accessible providing a URL. In a representation, segments normally exhibit the same duration, which may vary from representation to representation. For streaming services, the segment duration is rather short, of the order of a few seconds, not to aggravate the end-to-end delay budget.

The multimedia presentation description (MPD), a document in the form of a manifest file, details each period, providing its starting time, its duration, and the available adaptation sets. It is the MPD that also provides the URL, where segments can be referenced at, together with a segment availability interval, that is, the temporal window in wall-clock time during which the segment can be accessed via the HTTP–URL. The MPD is initially passed to the DASH client, which parses it and next proceeds to issue HTTP get requests to retrieve the segments of the desired representations.

The DASH standard does not indicate which algorithm to employ to rule the switching from representation to representation that is left entirely open. However, for the dual purpose of enabling synchronization and flawless switching, DASH adopts a common timeline shared among different media components and versions. The segments contain precise timing information about the presentation of media content that in principle allows clients to switch from representation to representation at an arbitrary point in time. In practice, however, this turns out complex for different reasons, most notably, because there are coding dependence within the same representation and because for live streaming services, segments get available over time, depending on their position in the media presentation timeline. Hence, the most immediate choices suggested by the standard to ease the switching process are: 1) to have each segment begin with a stream access point, a position in the media stream that enables the play out to start relying only upon the information in the representation data from that position onward and 2) to keep the boundaries of segments aligned across different representations in the same adaptation set. This paper adheres to these suggestions and examines the case of live streaming services provided through DASH to a large population of users.

IV. Proposal

A. Its Rationale

P2P is the significant feature that rules the distribution of multimedia content within the proposed architecture. Users are, therefore, organized in a virtual network, which is built at application layer, and are actively involved in the video diffusion process. They relay portions of the content that resides in their local streaming buffer to other users—their peers—through the virtual links of the overlay. We consider the case where a single video channel has to be streamed to a population of users, whose size is N, and assume that K DASH representations of the video are instantaneously available. Unlike VoD client–server architectures, where K takes on relatively high values, in the examined scenario, K is intentionally confined to lower values, not to excessively complicate the design of a live system, which has to respect tighter latency constraints.

We assume that the bit rate is what distinguishes the different available DASH representations, each being distributed

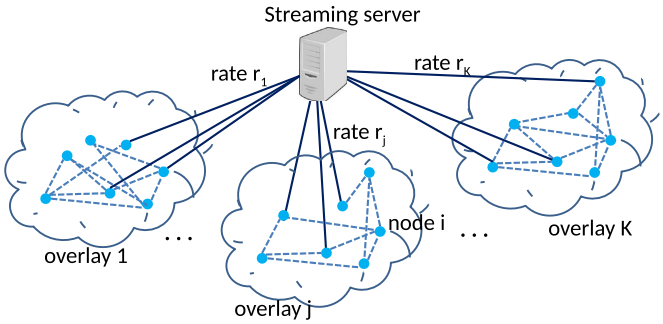

Fig. 1. System architecture.

within a separate P2P overlay. The jth representation displays a streaming rate r_j and is delivered within the jth overlay, equivalently termed swarm, $1 \leq j \leq K$. Without loss of generality, we set $r_j < r_{j+1}$, $\forall j$, $j = 1, 2, \ldots, K-1$. As shown in Fig. 1, the server is responsible for initially igniting the diffusion process. It delivers each representation to the corresponding overlay as a sequence of video chunks, relatively small fragments of the encoded video. Every peer belongs to one overlay at a time and contributes to sustain the process with its own upload bandwidth, further spreading the video chunks that it receives to other peers that belong to the same overlay and that request them. C distinct peer classes are assumed, each exhibiting different upload and download capacities. Hence, peers contribute to the system good functioning differently, depending on the class they belong to.

The examined P2P overlays are mesh-based, and within each of them, every peer exchanges video chunks with M randomly chosen neighbors, implementing a pull-based protocol. The peer periodically informs its neighbors about the video chunks that it holds in its streaming buffer by forwarding them its buffer maps. Moreover, the peer asks its neighbors for the chunks that are missing within its current request window W. Such window identifies the range where chunks that can be requested by the node fall: its right edge coincides with the highest chunk sequence number that the node is aware of from the inspection of the buffer maps it received, and its left edge is X s behind. The request window slides forward whenever the peer learns through the new buffer maps that it receives from its neighbors that a chunk with a higher sequence number is available. When a chunk that is yet to be received falls outside the current request window, it will not be claimed any longer, as outdated. For the sake of clearness, Table I collects the definition of the most relevant adopted terms, as they will frequently appear throughout this paper.

"A peer desires the DASH representation that offers the rate that the peer intends to use to view the video content. We observe that it depends on the combined limits set by the user's access network and by the display characteristics of the user's terminal. The user's preferences might also contribute to determine it, as it is the case when different charging rates are applied for distinct video qualities.

A peer requesting the video channel for the first time begins watching representation 1, that features the lowest bit rate, r_1, and, hence, the lowest quality. This choice allows to start the play out in a reasonably short time, and positively influences the overall quality that the user experiences [16]. After joining overlay 1, if the peer's desired representation is not the video alternative at rate r_1, the peer attempts to move upward and possibly succeeds; however, during the streaming, the peer might also have to move downward to a lower quality representation, depending on current system conditions, and then, it will dynamically attempt to advance again. These actions translate into migrations from one overlay to another. We require that a peer exclusively moves from its current overlay to an adjacent one, i.e., from overlay j to $j+1$ or to $j-1$ if $2 \leq j \leq K-1$, from overlay j to overlay $j+1$ if $K = 1$, and from K to $K-1$ if $j = K$. This guarantees the minimal gap between two consecutive representation play outs, as recommended in [17] and [18], to confine the amplitude of rate variations and their negative effects on the perceived video quality. Furthermore, we assume that every DASH segment has the same duration and it is made of the same number n of video chunks, so that the chunk duration t_{chunk} is the same in every overlay. This translates into a different amount of media bits being placed in every chunk, indicated by L_j such size for the chunks that are distributed within overlay j, we have

$$L_j = t_{\text{chunk}} \cdot r_j. \tag{1}$$

The choice of equal duration segments allows a smooth transition between different representations, as switching involves playing to the end of a segment in a representation and then playing from the beginning of the next segment in the new representation. In the realistic circumstance of a time lag between different overlays, the constant segment duration still protects the smoothness of the transitions, as long as the user's buffer can absorb time shifts among different representations.

TABLE I
GLOSSARY

Term	Definition
node	generic entity belonging to a network
peer	entity belonging to a P2P network, where all nodes have the same importance, that is, they are all "peers"
swarm/overlay	group of peers connected through virtual links at application layer
video chunk	small fragment of a video sequence, lasting a few seconds at the most
buffer map	a map representing which video chunks are currently available within the node buffer
representation	specific version of the source multimedia content. This work assumes that each representation is identified by a different bitrate

B. Core Algorithm

In a conventional client-server scenario, DASH provides the user with the functionalities needed to perform adaptive streaming and leaves open the selection of the algorithm that rules the switching among different representations. The client is assigned the task to check its current conditions via the monitoring of various parameters, and to ask the server for a different representation, if needed.

As previously observed, proposals in the literature rely on the observation of several indicators and put forth different criteria to govern the switching. However, it is exclusively the local perspective that plays a role in the design of the control algorithm. Within the current proposal, although the primary aim of the peer remains to experience a satisfying streaming quality and to efficiently deploy network resources, the user's decisions cannot any longer be taken in isolation, as the peer has to be fair to all other peers, given its viewing quality is strongly influenced by the quality of the streaming process of the entire overlay, that is, it depends on the status of all other peers. It is, therefore, required that every peer employs two distinct types of status indicators, local and global, to decide if, when and where to migrate, whereas the former indicators indirectly supply information about the peer's video quality and are locally measured by the peer itself, and the latter parameters provide a clue about the current health of the overlays and are periodically handed out by the server to all peers that rely upon both to enforce the adaptive rate selection algorithm.

Among local indicators, we utilize the following.

1) The delivery ratio (DR) is defined as the ratio between the number of video chunks that meet the playback deadline over the total number of chunks that a peer should receive, measured with a periodicity of $\tilde{t}$ s. Such ratio indicates the throughput that the peer experiences and indirectly signals the quality of the received video in the very recent past.
2) The request window state (RWS) is defined as the ratio between the number of downloaded video chunks within the current request window W and the size of such window measured in number of video chunks, $0 \leq \text{RWS} \leq 1$. This indicator provides an indirect forecast of video quality in the near future, as its value reflects the imminent status of the play out buffer at the peer's site.

Among global indicators, we select those that the monitoring server of a P2P platform typically determines and that can distribute to peers with very little effort as follows.

1) The instantaneous resource index [19] of the jth overlay, $\sigma_j(t)$, $j = 1, 2, \ldots, K$, defined as

$$\sigma_j(t) = \frac{C_{Sj} + \sum_{i \in N_j(t)} c_i}{|N_j(t)| \cdot r_j} \tag{2}$$

where C_{Sj} is the capacity that the server commits to the jth overlay to distribute the representation with rate r_j, c_i is the upload capacity of the ith peer belonging to overlay j, $N_j(t)$ is the set of active nodes within such overlay at time t, and $|N_j(t)|$ is the set cardinality.

2) The instantaneous efficiency of the jth overlay, defined as

$$E_j(t) = \frac{U_{Sj}(t) + \sum_{i \in N_j(t)} u_i(t)}{|N_j(t)| \cdot r_j} \tag{3}$$

where $U_{Sj}(t)$ is the actual upload rate that the server provides to overlay j at time t and $u_i(t)$ is the actual upload rate at time t of the ith user within the same overlay.

When the instantaneous resource index $\sigma_j(t)$ takes on a value greater than or equal to 1, in principle, the jth overlay can successfully guarantee the video delivery to all of its members, whereas when its value falls below 1, the overlay operates in a critical regime; hence, σ_j is a high-level indicator of the overlay health. However, the instantaneous efficiency $E_j(t)$ provides a more accurate picture, as it captures some system behavior that would otherwise go unseen, if σ_j only were examined. To depict a scenario where this happens, it suffices to think of a flash crowd of viewers that abruptly enters the system, wishing to stream the video: newly incoming peers initially act as free riders, as they momentarily have no video chunks to share. As a consequence, the value of the instantaneous efficiency drops, revealing a potentially critical operating condition that the resource index cannot seize.

Let us next consider the generic peer i within overlay j and indicate by $r_d(i)$, the streaming rate of its desired representation. The steps that the algorithm enforced by the peer goes through every Δt seconds are listed in the following.

1) Peer i checks its current streaming rate r_j against $r_d(i)$ and if $r_j < r_d(i)$, i.e., if the peer is not satisfied, it first verifies whether it can leave its current overlay or it has to defer its departure. This last circumstance occurs if overlay j is not in good health and node i upload capacity is beneficial to it, that is, if σ_j is lower than 1 and if the peer upload capacity c_i is greater than the streaming rate r_j. In this case, it is convenient that the peer does not move upward to overlay $j+1$. If on the contrary nothing prevents the departure, peer i further verifies if its future contribution to the target overlay $j+1$ will be positive, i.e., if its upload capacity c_i is greater than the streaming rate r_{j+1}. If so, the peer migrates, as it will be beneficial to overlay $j+1$. If not, the node further checks that overlay $j+1$ has abundant overall upload capacity and that the video diffusion process within the overlay is taking place in an efficient and noncritical manner, and hence, the target overlay is able to accommodate a new peer, regardless of it being a relatively poor contributor. In detail, the peer moves to overlay $j+1$ if $\sigma_{j+1} > 1$ and $E_{j+1}(t) > E_{\text{thres}}$, where E_{thres} is a properly set threshold.
2) Peer i also verifies its local status quality indicators, updating the weighted average of its DR and RWS in the following manner:

$$\text{DR}_i^{(t)} = w_D \cdot \text{DR}_i^{(t)} + (1 - w_D) \cdot \text{DR}_i^{(t-\Delta t)} \tag{4}$$

and

$$\text{RWS}_i^{(t)} = w_W \cdot \text{RWS}_i^{(t)} + (1 - w_W) \cdot \text{RWS}_i^{(t-\Delta t)} \tag{5}$$

respectively, where w_D and w_W are the weight coefficients. If $\text{DR}_i^{(t)}$ and $\text{RWS}_i^{(t)}$ are below their predefined thresholds, $\text{RWS}_{\text{thres}}$ and DR_{thres}, respectively, the viewing quality is not deemed acceptable and the peer scales down to a lower rate representation, and hence, it moves to overlay $j-1$.

The pseudocode describing the algorithm locally implemented by the peer to steer its movements among overlays is reported in Algorithm 1.

Algorithm 1 Rate Switching Control Algorithm

```
Node i in overlay j every Δt seconds
;verifies its satisfaction
if (r_j < r_d(i)) then
  ;verifies the current overlay status
  if ((σ_j < 1) and (c_i >= r_j)) then
    do not migrate to overlay j + 1;
  else
    ;verifies the destination overlay status
    if ((c_i > r_{j+1}) or (σ_{j+1} > 1 and E_{j+1} > E_thres)) then
      migrate to overlay j + 1; exit;
    end if
  end if
end if
;verifies its viewing quality
if ((DR_i^(t) < DR_thres) and (RWS_i^(t) < RWS_thres)) then
  migrate to overlay j − 1; exit;
else
  stay in overlay j; exit;
end if
```

Note that in the algorithm, the upgrade decision is conservatively influenced by the peer's local metrics—its current rate r_j and its desired streaming rate $r_d(i)$—and also by the global indicators of the current and destination overlays, whereas the downgrade process is exclusively governed by local indicators. The rationale behind this choice is that the global indicators point to the overall status of the system and, therefore, let the peer learn if the migration from its current swarm to the overlay distributing a higher bit rate representation can be safely performed. On the other hand, through local indicators, the peer indirectly measures its streaming quality and decides if it is time to scale down to a lower DASH representation, in order to preserve a good viewing experience.

C. Silent Neighborhoods, Buffer Reuse, and Priority Queuing

In this section, we discuss some structural features that we introduce in the design of the P2P platform. We begin observing that, in the transition from the current overlay to an adjacent one, the peer looses some time in disconnecting from its old neighbors, building up the new neighborhood, and then waiting for buffer maps. Only at this point in time, the peer is able to request video chunks. These actions are necessary for the successful insertion of the peer into its target overlay, but inevitably require some time that might in turn translate into a discontinuous play out. In order to confine such delay that we term setup time, the proposed system relies upon an additional feature that we call the silent neighborhood, whose underlying idea is to have the peer maintain contacts with nodes both in its current and also in its adjacent overlays. Namely, we request the peer in overlay j to create and simultaneously maintain a neighborhood in three overlays, the jth, the $j+1$th, and the $j-1$th, and to exchange buffer maps with the nodes belonging to all three neighborhoods. However, as shown in Fig. 2, only one of such sets of nodes is active at a time, i.e., the

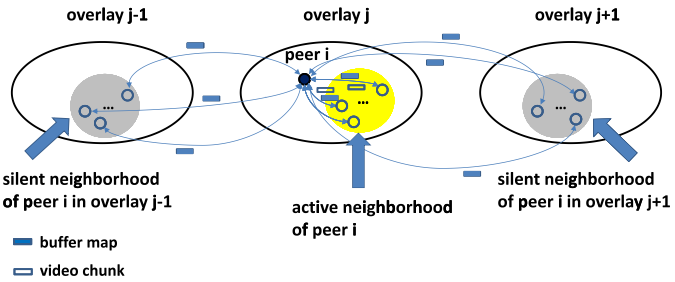

Fig. 2. Silent neighborhood mechanism.

peer exchanges both buffer maps and video chunks with the nodes that belong to the active neighborhood in the jth overlay where the peer resides, whereas it only exchanges buffer maps with its neighboring peers in the two adjacent, silent overlays. The construction of the silent neighborhood obeys the same rules as the active one. M peers are randomly chosen within the overlay, and whenever a node departs, it is immediately replaced by a new one. Moreover, a reciprocal relationship holds: if node i belongs to node h silent neighborhood, then node h is a silent neighbor of i.

There is one first and immediate benefit that the silent neighborhood solution brings in: when the peer migrates to an adjacent overlay, its setup time is zeroed, as the node can immediately ask its silent neighbors to accept it as an active node, assigning some of their actual upload bandwidth to the communication and beginning to deliver content to the peer. Within the process, if the uplink bandwidth of some among the potential new neighbors is exhausted, the peer will look for alternative partners; however, the peer will very unlikely start from scratch in building up its new neighborhood; rather, it will swiftly begin receiving the video chunks currently distributed in the target overlay.

It is indisputable that the silent neighborhood solution implies some signaling overhead, as the generic peer constantly transmits and receives more buffer maps than in the conventional architecture, but the burden is more than acceptable, as it will be quantified in Section V.

As regards the streaming process within the target overlay, note that it can be early with respect to the origin overlay of the peer, meaning that the swarm distributes newer video chunks, or it can be late, that is, the overlay distributes older video chunks, and the node has to adapt to both conditions. For doing so, it determines the newest advertised video chunk in the target overlay and from there it begins to request chunks backward for the entire width W of its request window. Assuming that the head of the newest chunk corresponds to time t_{curr} within the video stream, the requests will fall within the $R = [t_{\text{curr}} - W, t_{\text{curr}}]$ interval. Both the cases are shown in Fig. 3. In analogy to what happens when the node joins the system for the first time and has to collect a reasonable amount of content before the play out begins, we consider the migration to the new overlay successfully completed when the node has accumulated a given number, say S, of consecutive video chunks within its buffer, and we call the delay that the node incurs to complete this process, i.e., the switching delay. Luckily, because of the previous

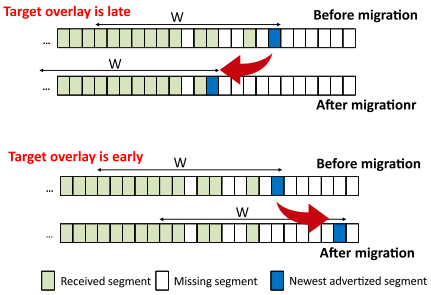

Fig. 3. Early and late target overlays.

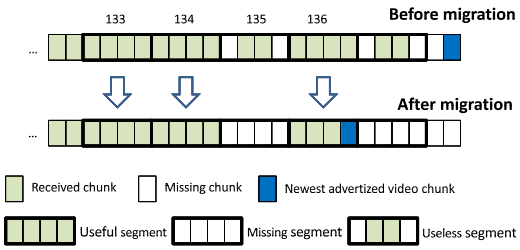

Fig. 4. Example of missing fragments.

assumptions, DASH moves from one representation to another with a time granularity which is equal to the size of a segment, so that at the time the switching takes place, the node can profitably inherit all the segments—not necessarily contiguous—within its buffer that fall within R. This buffer content reuse contributes to avoid video stalls and eases the transition to the new video representation. Note that if the buffer reuse mechanism only was introduced, with no silent neighborhoods, the video chunks already present in the peer's buffer at the time of the transition would not be exploited in the most effective manner. As a matter of fact, to identify which DASH segments can be reused, a node needs to know the streaming position of the target overlay. To this end, it has to construct the new active neighborhood within the target overlay and then wait for some buffer maps. During this time interval, some of the video segments stored in the streaming buffer might become outdated, and if this happens, they have to be discarded to make room for more recent video chunks. On the contrary, when the silent neighborhood mechanism is implemented, the migrating peer is already hooked to the newest advertised chunk in the target overlay and immediately proceeds to check what chunks can be profitably inherited to accumulate S seconds of consecutive video.

We aim at quantifying the benefit that the buffer reuse allowed by the DASH guarantees, and to have a sound term of comparison, we consider the reference case where the node joining the new overlay is forced to remove the video chunks of the representation from its buffer, which it was previously receiving, and to collect the video chunks of the new representation from scratch. As every DASH segment is composed by n video chunks, at migration time, it might happen that the node has not received all chunks of one or more segments. Segment fractions are useless and have to be discarded, whereas only the completely received segments are inherited. This statement is made clear by the example shown in Fig. 4, where the segment with sequence number 135 is missing after the migration process. Yet, noncontiguous segments numbered 133, 134, and 136 can be reused.

A further technique that we propose to ameliorate system performance is the introduction of priority for the delivery of video chunks from the parent node to the requesting peers. This solution implies that when the peer moves to an adjacent overlay, its requests for video chunks are taken care of by its parent nodes with higher priority than the requests raised by nodes already resident within the swarm, so as to favor a swift acquisition of video content. This privileged condition lasts for an adequate, although finite, time interval, which is again equal to the time needed to accumulate S seconds of consecutive video. Accordingly, in every overlay, chunk requests are categorized as either low or high priority. Moreover, a low-priority chunk request being received, while the parent node is busy, is lost, whereas some limited waiting is allowed for high-priority requests that are lost only when the corresponding queue, i.e., buffer, is full. In order not to excessively penalize the video distribution process toward the nodes that already present within the overlay, the size of such queue is limited, and its value depending on the upload capacity of the distributing node and on the size in bytes of the video chunks being spread in the overlay. Indicating by $Q(l, j)$ the queue length of the generic parent node belonging to class l, with upload capacity c_l, within overlay j, and by t_{req}s the periodicity by which the nodes issue new chunk requests, we set

$$Q(l, j) = \frac{t_{\text{req}}}{\frac{t_{\text{chunk}} \cdot r_j}{c_l}} \quad \forall l = 1 \ldots C \quad \forall j = 1 \ldots K \tag{6}$$

so as to guarantee that if a video chunk is queued, it will be sent to the requesting peer at most after t_{req}s. For this feature too, in Section V, we will numerically demonstrate what effects it introduces on system behavior.

D. Diverse Requirements for Diverse Peer Classes and Overlays: The Modified Algorithm

As our interest is in devising mechanisms that allow DASH a fluid play out, we finally conceive a change to the core algorithm so as to guarantee an even sleeker overlay transition. The idea is to force the streaming process within the different overlays to be as much as possible aligned. The transition will, therefore, be facilitated, since a larger portion of the peer buffer content can be advantageously reused. A way to achieve this is to harden the conditions that rule the peer transition to an overlay that distributes video at a higher rate than the peer's capacity, and to make such conditions stiffer when the rate of the target overlay increases. Accordingly, the original algorithm, Algorithm 1, is modified as detailed in Algorithm 2, where we introduce a matrix of resource index thresholds, whose generic element is $\sigma_{\text{thres}}(l_i, j)$, l_i being the class that node i belongs to and j the index of the overlay, obeying the following rules: $\sigma_{\text{thres}}(l_i, j) > \sigma_{\text{thres}}(l_h, j)$, under the hypothesis that peer i belongs to a class whose upload capacity is lower than the class peer h belongs to,

Algorithm 2 Modified Switching Control Algorithm

```
Node i in overlay j every Δt seconds
;verifies its satisfaction
if (r_j < r_d(i)) then
  ;verifies the current overlay status
  if ((σ_j < σ_thres(k_i, j)) and (c_i >= r_j)) then
    do not migrate to overlay j + 1;
  else
    ;verifies the destination overlay status
    if ((c_i > r_{j+1}) or (σ_{j+1} > σ_thres(k_i, j+1) and E_{j+1} > E_thres)) then
      migrate to overlay j + 1; exit;
    end if
  end if
end if
;verifies its viewing quality
if ((DR_i^(t) < DR_thres) and (RWS_i^(t) < RWS_thres)) then
  migrate to overlay j − 1; exit;
else
  stay in overlay j; exit;
end if
```

and $\sigma_{\text{thres}}(l_i, j+1) > \sigma_{\text{thres}}(l_i, j)$, recalling overlay $j+1$ distributes video at a higher rate than overlay j. When compared with the original algorithm, the new approach forces weaker peers to reside within the overlays that stream the video at the lower rates, and hence, in some cases, it prevents them to reach the desired swarm. Yet, we will numerically provide evidence that the overall performance they experience is more rewarding, which is the final result that every system should strive to achieve.

E. Merit Figures

To assess the performance of the proposed architecture and gain a thorough understanding of the effects that its different features bring in, various merit figures have to be considered. In detail, we have focused upon and evaluated the following indicators.

1) The DR experienced by system users within the different overlays that indicates how efficiently the streaming process takes place.
2) The playback delay is defined as the interval that elapses from the point in time when the video chunk is generated by the streaming server up to the instant when it is received by the viewing user, so as to gain an insight on the time lag that the P2P distribution inevitably introduces, and also to verify to what extent the solution is able to fulfill the users' requests of near-real-time services.
3) The distribution of the users within the overlays to understand how closely their original streaming rate requests can be satisfied.
4) Finally and most importantly, the switching delay incurred by the generic user when it moves from the current overlay to a new one, an action which DASH adoption inherently favors.

TABLE II
USER CAPACITY PROFILES AND PERCENTAGES

	Class 1	*Class 2*	*Class 3*	*Class 4*
Upload capacity (kbit/s)	704	1024	1500	10000
Download capacity (kbit/s)	2048	8192	10000	50000
% of peers	20	21	42	17

V. NUMERICAL RESULTS

A. Simulation Setup

The behavior of the proposed system is investigated with the help of an event-driven simulator, already employed in [12]. The simulator builds a replica of a multioverlay P2P system, where every swarm distributes one DASH representation. The system is populated by $N = 2000$ active peers that dynamically enter and leave. Nodes join the system within the first 20 s, and in this interval, their inter-arrival times are exponentially distributed with an average of 0.1 s. After the first 20 s, the inter-arrival times are modulated so as to keep nearly constant the number of peers in the system. Session times are exponentially distributed, with an average duration of 1500 s. Every simulation lasts for 3000 s. Nodes belong to $C = 4$ different classes whose upload and download capacity values stem from the current European Internet connection offerings [22]–[24]. The percentages of users belonging to the different classes are drawn from the Akamai European average connection speed report [25]. The employed values are reported in Table II. $K = 4$ video representations are available at rates 700, 1500, 2500, and 3500 kb/s. Such values have been chosen having in mind the typical streaming rates of Internet standard definition (SD) video and high-definition (HD) video. The streaming server allocates only a small amount of its upload capacity to each overlay, which is equal to four times the rate of the video, i.e., $C_{Sj} = 4 \cdot r_j$, indicating that the focus is on a pure P2P system. Moreover, the size of the current request window that every peer works with is $X = 20$ s, the request interval for video chunks is set to $t_{\text{req}} = 0.8$ s, buffer maps are sent out every $t_{\text{BM}} = 1$s, the segment duration is 2 s, and every segment is divided into $n = 10$ chunks. Video chunks are then fragmented and carried inside packets whose payload is 1250 B, transmitted over the underlying wide area network, where they suffer latencies that are captured via a suitable delay matrix. Each pair of nodes in the overlay is randomly mapped to a pair of elements of the latency matrix, with and average end-to-end delay of 79 ms. Furthermore, the DR is locally computed every $\tilde{t} = 5$ s, and the threshold values we employ in the core and the modified algorithm are the following: $E_{\text{thres}} = 0.9$, $\text{DR}_{\text{thres}} = 0.55$, and $\text{RW}_{\text{thres}} = 0.3$. we have picked these values after a careful tuning, where we verified that they guarantee the optimal system performance, that is, the lowest playback delay and the highest DR [20]. The coefficients for the computation of the weighted average of DR_i and RWS_i are $w_D = (1/3)$ and $w_W = (2/3)$, respectively, revealing that with regard to the local DR, we privilege stability in the estimate in (4), whereas for the RWS in (5), we give priority to what happens at current time.

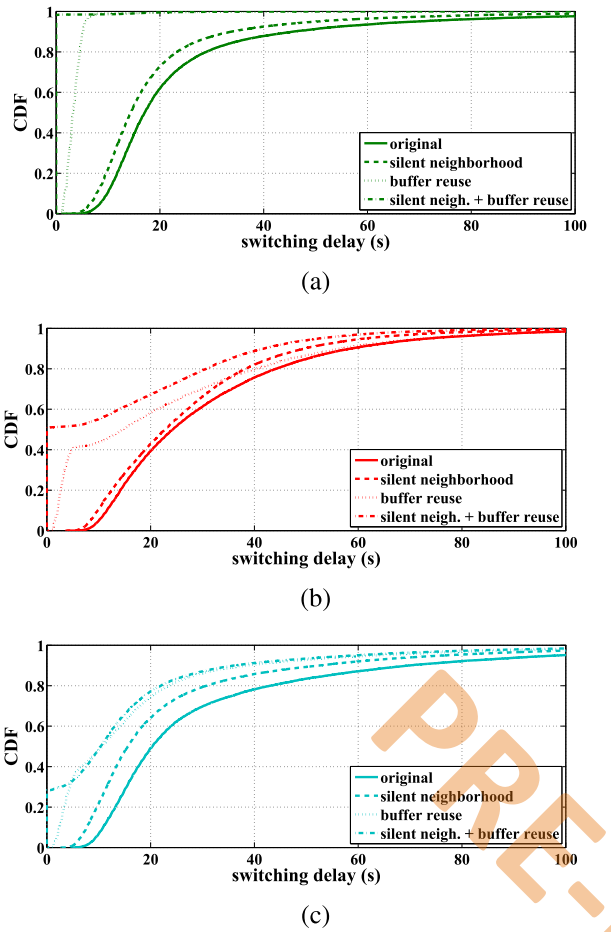

Fig. 5. Switching delay cdf. (a) Overlay 2. (b) Overlay 3. (c) Overlay 4.

Finally, the periodicity that the rate control algorithm employs is $\Delta t = 4$ s.

We investigate system behavior when every peer aims at streaming the video at the best possible representation, i.e., at the highest bit rate, which results lower than the peer download capacity. In this circumstance, all peers aim at streaming representation 4, except for class 1 peers that are confined to representation 2. Within this scenario, it is impossible to fulfill all users' requests. If peers were uncritically placed within the overlays distributing the desired representations (2 and 4), the aggregate upload capacity of each swarm would not be enough to satisfyingly deliver all peers the video, the resource index values being markedly lower than 1, namely, $\sigma_2 = 0.48$ and $\sigma_4 = 0.91$.

To estimate the switching delay introduced in Section IV-C, we set $S = 8$ s that we deem a reasonable duration (similar values are encountered in the literature when estimating the play out delay).

B. Silent Neighborhood and Buffer Reuse

Fig. 5(a)–(c) shows the cumulative distribution function (cdf) of the switching delay for the system that leverages either the silent neighborhood or the buffer reuse mechanism, or both, and for a reference system that we call the original system that does not employ them. Fig. 5(a)–(c) reveals that such features statistically improve the switching delay, although in a different manner. Moreover, these actions are significant for all overlays, except for overlay 1, where the majority of inputs are caused by the arrival of new peers in the system, which explains why the corresponding results are omitted. When the system employs only the silent neighborhood mechanism, the improvement consists in reducing the setup time to zero. In the original system, this time is modeled as a random variable, uniformly distributed between 500 and 4000 ms. Hence, the gain is moderate, and the horizontal lag between the corresponding curves (solid line versus dashed line) is of the order of magnitude of its average, equal to 2250 ms. We also observe that the combined use of the silent neighborhood mechanism and the buffer reuse (dashed-dotted lines) leads to the best results. The former action allows the nodes to immediately know which complete segments can be reused, on the basis of the streaming position within the migration overlay. Indeed, every silent neighbor advertises what video chunks it owns, so that a migrating node already knows the newest video chunk distributed within the target overlay and immediately proceeds to request the missing video chunks. As an example, Fig. 5(a) referring to overlay 2 indicates that a node in the original system (solid line) experiences a switching delay equal to or greater than 25 s with probability 0.8, while in the system that introduces only the buffer reuse (dotted line), the delay decreases to 4.5 s and it is 0 s for the system that jointly combines the two actions (dashed-dotted line). The exclusive introduction of the buffer content reuse (dotted lines) allows to reduce the switching delay in a different way depending on the considered overlay, namely, the advantage decreases when the distributed video streaming rate increases, mainly due to the mismatch between the streaming point of adjacent overlays. The fact that overlay 3 is a transit overlay, where there are not stable nodes, contributes to misaligning the streaming process between overlay 2 and overlay 3, as well as between overlay 3 and overlay 4. Notably, this is observed in Fig. 5(c) that refers to overlay 4. Here, to know in advance how many and which DASH segments the node can reuse leads to a minimal benefit, since there is a high probability that the number of inheritable segments is very low.

The silent neighborhood mechanism inevitably introduces some overhead, as a generic node has to construct and manage three neighborhoods in the place of one, and several control packets are necessary to connect, disconnect, replace the silent neighbors, and to exchange buffer maps. So, in addition to quantifying the significant benefits, this mechanism introduces when employed in conjunction with the buffer reuse, it is also mandatory to assess the price to be paid. To this aim, we define the control overhead as the ratio between the control bytes and the overall bytes sent by a node calculated with a time periodicity of $\Delta = 5$ s. Fig. 6(a)–(d) shows the overhead averaged over all nodes in each overlay as a function of the simulation time. In the original system (solid lines), the control overhead decreases from overlay 1 to overlay 4. This had to be expected, as the size of the control packets is the same in all overlays, whereas the size of the video chunks increases in the overlays that distribute the video at higher streaming rates,

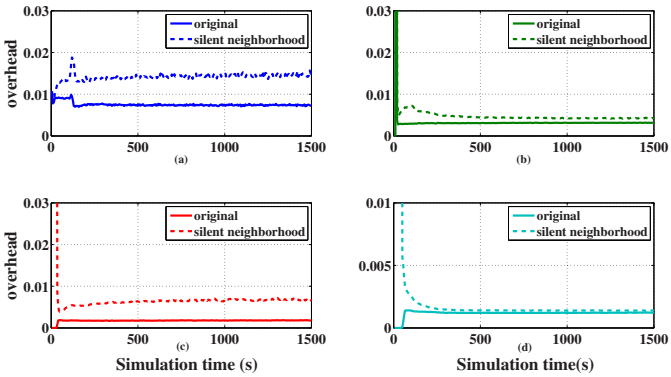

Fig. 6. Control overhead with and without silent neighborhoods. (a) Overlay 1. (b) Overlay 2. (c) Overlay 3. (d) Overlay 4.

as indicated in (1). When the silent neighborhood is introduced, the overhead increases, in particular for those overlays, where the joining events take place more frequently, i.e., for overlay 1, as all nodes joining the system enter from here, and for overlay 3, an overlay where a node transits toward overlay 4, but where it often comes back if its viewing quality is not satisfying. Even though the values of overhead double and triple for overlay 1 and overlay 3, in the worst cases, they are confined to about 1%–2%, so we can safely affirm that the percent increase of control packets due to the silent neighborhood management is abundantly counterbalanced by the switching delay reduction.

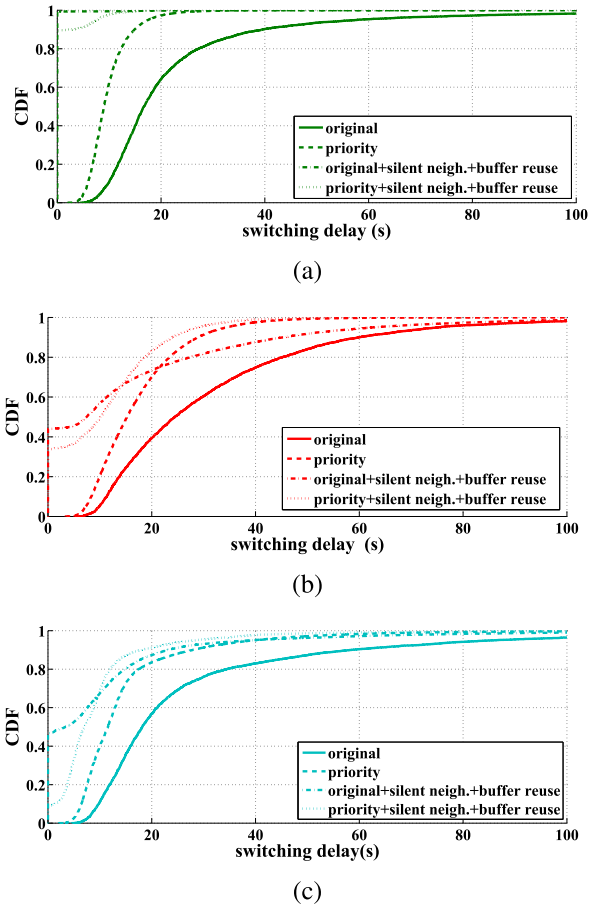

Fig. 7. Switching delay cdf. (a) Overlay 2. (b) Overlay 3. (c) Overlay 4.

C. Priority Queuing

This section reports the results obtained when the scheduling policy adopted by the parent nodes to provide video content to the requesting peers is modified, in order to assign higher priority of retrieving video chunks to those users that have just entered the overlay and need to fill up their streaming buffer, to either start or continue the play out at a different rate. As regards the $Q(l, j)$ values defined in (6) that have be employed, some numerical examples are reported next: for a class 4 node with upload capacity $c_4 = 10\,000$ kb/s belonging to overlay 1 that distributes video at $r_1 = 700$ k/s, the queue length is $Q(4, 1) = 57$, whereas if a node of the same class belongs to overlay 4 that distributes video at $r_4 = 3500$ kb/s, its queue length is $Q(4, 4) = 11$.

Fig. 7(a)–(c) shows the cdf of the switching delay experienced by the peers in overlays 2–4. As before, the viewgraph referring to overlay 1 is omitted. Fig. 7(a)–(c) allows to compare the original system (solid line) to the architecture that introduces the priority policy (dashed line), to the system where the silent neighborhood and the buffer reuse techniques are introduced (dashed-dotted line), and finally to the one that jointly combines the priority mechanism to the silent neighborhood management and the buffer reuse. Comparing the original system with the one that introduces the priority scheduling policy, from Fig. 7(a) referring to overlay 2, we observe that a generic node within the latter system accumulates 8 s of video in 12.3 s or fewer seconds with probability 0.8, against the 26.6 s required by a peer in the original system. Similar gains are observable for overlays 3 and 4, as shown in Fig. 7(b) and (c). When examining the system that combines the priority scheduling policy with the silent neighborhood management and the buffer reuse, we conclude that here too a statistical improvement of the switching delay is achieved, as we note comparing the dotted and solid lines in Fig. 7(a)–(c). The delay reduction is different for the three overlays. It is huge in overlay 2, where with probability 0.9, and the switching delay becomes 0 when the buffer-reuse strategy is introduced and still very appreciable for overlays 3 and 4. However, the combined adoption of priority and neighborhood plus the buffer reuse mechanism has a contrasting effect.

1) For low values of switching delay, the curves that refer to the system with no priority (dashed-dotted lines) are better.
2) For high values of switching delay, the curves that refer to the system with priority scheduling (dotted lines) are better.

With reference to the first remark, we anticipate that the introduction of priority scheduling to some degree deteriorates the streaming process in overlay 3, which results more misaligned with respect to the adjacent overlays. Numerical evidence of this statement will be provided in the next figures. Any misalignment implies a lower efficiency of the silent neighborhood and buffer reuse techniques, since there are not so many useful

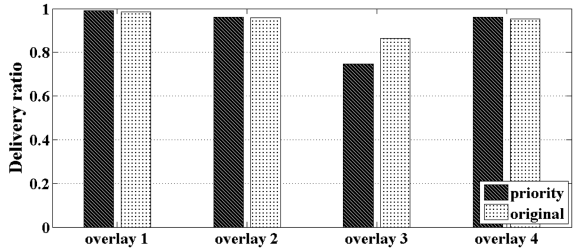

Fig. 8. Average DR for the original system and for the system with priority scheduling.

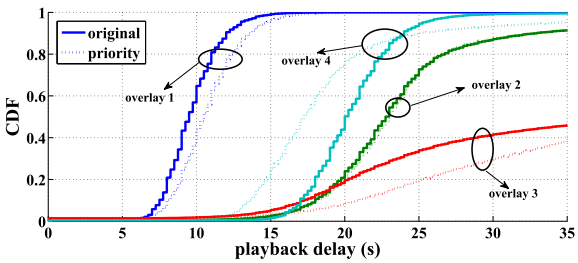

Fig. 9. Playback delay cdf for the original system and for the system with priority scheduling.

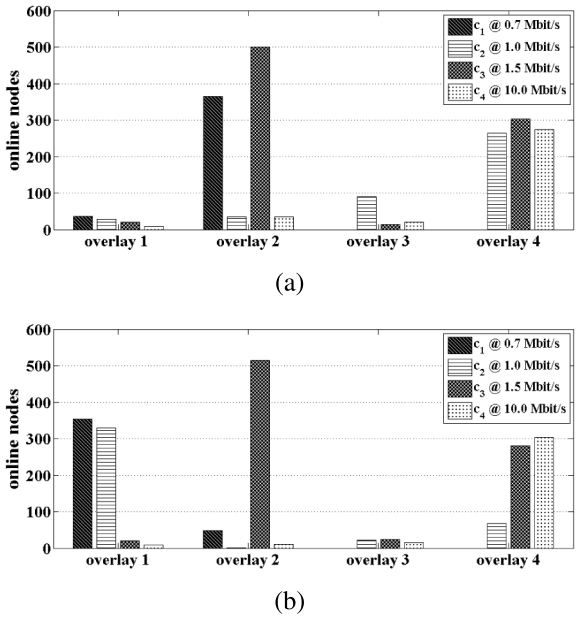

Fig. 10. Per class node distribution within each overlay. (a) Original system. (b) System with differentiated resource index thresholds.

TABLE III
AVERAGE VALUES OF THE RESOURCE INDEX

	overlay 1	overlay 2	overlay 3	overlay 4
original system	2.5	1.0	1.0	1.1
σ_{thres} system	1.4	1.0	1.4	1.5

DASH segments to inherit. For this reason, we assist to a decrease in the probability of a generic peer to experiment no switching delay at all and to an initial worsening of the cdf curves. This effect is predominant for low values of the switching delay, that is, for the nodes that migrate with an almost full buffer and take a little time to fill it. However, we can also state that the adoption of the priority scheduling accelerates the collection of the video chunks needed to fill up the buffer, so the slope of the curves of the system with priority (dotted lines) is steeper than that of the system with the silent neighborhood and the buffer reuse (dashed-dotted lines). This effect is more evident for high values of the switching delay, that is, for the nodes that join the new overlay with an almost empty buffer and that cannot benefit of the buffer reuse effect. Moreover, these figures indicate that the system with priority guarantees lower values of the maximum switching delay.

We next complete the investigation on the effects of priority scheduling on system performance and determine the average DR and the cdf of the playback delay. Fig. 8 shows the average DR of each overlay for the system that introduces the priority scheduling and compare it with the original architecture. The values of DR have been calculated as averages over the total number of peers and also over the last 1000 s of simulation, when the system exhibits a stable behavior, for ten different simulation runs. We notice that only the DR of overlay 3 decreases, whereas the remaining values in the two systems are comparable. This occurs because overlay 3 is a poorly populated overlay, subject to frequent join events and departures, as no peer actually wants to reside in it. The few older nodes are busy serving the numerous newcomers with higher priority, at the expense of the video diffusion process toward its old peers.

Fig. 9 shows the cdf of the playback delay experienced by a generic node in the different overlays, and as anticipated before, it reveals that the spreading process in the odd overlay moderately worsens, due to the introduction of the priority scheduling policy, and that the same holds for overlay 1, the system entry point. Overall, it is noteworthy to remark that the priority policy applied in isolation represents an effective tool to reduce the switching delay. When combined with other features, it nevertheless guarantees to confine such delay to modest values with high probability.

D. Modified Control Algorithm

As an alternative to the previous system with all its possible variations, we next investigate the behavior of the modified algorithm introduced in Section IV-D. To this aim, we employ the matrix given in the following. As required for the algorithm to be effective, the $\sigma_{\text{thres}}(l, j)$ values increase or do not vary moving from left to right and from top to bottom

$$\sigma_{\text{thres}}(l, j) = \begin{bmatrix} 1.0 & 1.2 & 1.6 & 1.8 \\ 1.0 & 1.1 & 1.4 & 1.6 \\ 1.0 & 1.0 & 1.2 & 1.4 \\ 1.0 & 1.0 & 1.0 & 1.0 \end{bmatrix}.$$

The adoption of the new algorithm has an immediate impact on the average resource index of each overlay, as it is shown in Table III, as well as on the distribution of nodes among the various overlays, as shown in Fig. 10(a) and (b). Table III shows that, as desired, more resources are redirected toward overlays 3 and 4, those that distribute the video at higher streaming rates, at the expense of overlay 1.

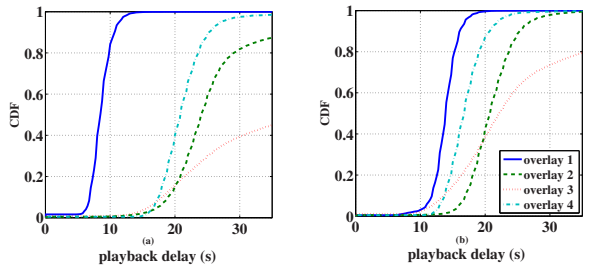

Fig. 11. Playback delay cdf. (a) Original system. (b) System with differentiated resource index thresholds.

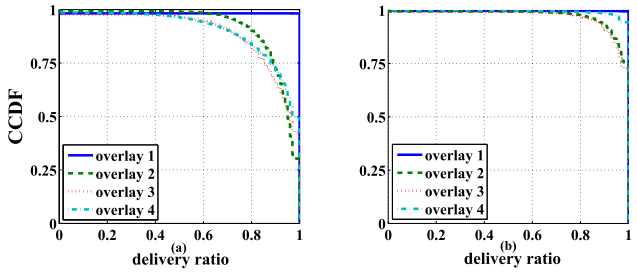

Fig. 12. DR CCDF. (a) Original system. (b) System with differentiated resource index thresholds.

TABLE IV

PERCENTAGE OF NODES THAT STREAM A REPRESENTATION n LEVELS AWAY FROM THE DESIRED

distance	original system	σ_{thres} system
0	60.4	35.0
1	8.1	20.7
2	28.6	26.4
3	2.9	17.9

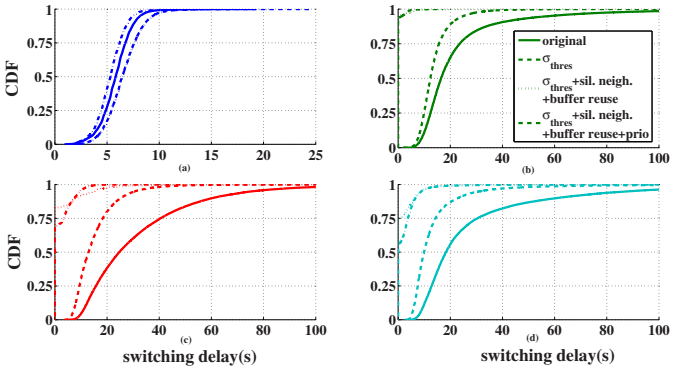

Fig. 13. Comparison between the switching delay cdfs. (a) Overlay 1. (b) Overlay 2. (c) Overlay 3. (d) Overlay 4.

Fig. 10(a) and (b), that report the node distribution in the original system and in the system with differentiated σ_{thres} thresholds, reaffirm that the modified algorithm tends to place the weaker nodes within the overlays that distribute the video at a streaming rate lower than or equal to their upload capacity. Here too, the reported values have been obtained as averages over the last 1000 s of the simulation time and over 10 runs. The introduction of tighter thresholds limits the possibility of weaker nodes to reach the overlays that distribute the higher streaming rates, even if they desire it. At the same time, it guarantees a better content diffusion process, i.e., higher values of DR and lower playback delays, as shown in Figs. 11 and 12.

Fig. 11(a) and (b) allows to compare the cdf of the playback delay of the two systems. We note that the system that employs differentiated σ_{thres} thresholds is able to guarantee closer values of playback delay among overlays, which equivalently means more aligned streaming processes, exactly as desired. Moreover, all the curves improve (i.e., move to the left) for the system with differentiated σ_{thres} thresholds, except for overlay 1, where a reduction of resources occurs, as shown in Table III. To complete the picture, Fig. 12 shows the complementary cdf (CCDF) of the DR in the two compared systems, highlighting that the platform implementing the modified switching algorithm consistently achieves high values of DR with higher probability.

Taking a different perspective, for both systems, Table IV shows the percentage of nodes as a function of the distance between the overlay, where the node actually resides, and the overlay streaming the representation the node would like to view: $d = 0$ means that the peer is streaming the desired representation, whereas $d = 1, 2, 3$ indicates the peer is $1, 2, 3$ overlay(s) away from its target, respectively. The modified system is able to place fewer peers than the original platform within the desired overlay. Yet, Figs. 11 and 12 indirectly suggest that the viewing quality that the nodes experience is more remarkable, and this is the key advantage. In other words, it is better to stream at a lower bit rate with good quality, without noticing significant switching delays in the transition from one representation to another and experiencing good operating conditions, than to stream high rate videos in scarce conditions, as also suggested in [26].

To complete the investigation, Fig. 13(a)–(c) shows the switching delay cdf and indicates that we obtain a considerable improvement if we differentiate the σ_{thres} thresholds on the basis of the class that the node belongs to and of the destination overlay (dashed lines opposed to solid lines). As a remarkable example, we observe that in overlay 4 for the probability value 0.8, the switching delay reduces from 36.7 s in the original system (solid lines) to 16.2 s for the σ_{thres} system (dashed lines). In overlay 3, it reduces from 45.5 to 20.3 s, and for overlay 2, the gain is approximately 10 s; only for overlay 1, the cdf of the switching delay slightly deteriorates, as resources are less abundant than in the original system.

With the help of the same figures, it is interesting to investigate how the priority scheduling policy and the silent neighborhood management with buffer reuse influence the system that employs differentiated σ_{thres}. Fig. 13 reveals that the buffer reuse action (dotted lines) achieves a significant improvement in terms of switching delay, because of the greater alignment among overlays that the modified algorithm introduces. We report some numerical examples to emphasize the good results. With probability 0.8, a node within overlay 4 gathers 8 s of video chunks in 3.2 s, far less than the required 16.2 s if the buffer reuse is not active, and for overlays 2 and 3

TABLE V
NEW USER CAPACITY PROFILES AND PERCENTAGE

	Class 1	*Class 2*	*Class 3*	*Class 4*
Upload capacity (kbit/s)	1024	2048	3072	10000
Download capacity (kbit/s)	3072	8192	10000	50000
% of peers	20	21	42	17

TABLE VI
AVERAGE DR AND PLAYBACK DELAY VALUES

	Average Delivery Ratio	*Average Playback Delay (s)*
Overlay 1	0.997	13.5
Overlay 2	0.995	21.9
Overlay 3	0.960	29.2
Overlay 4	0.989	26.7

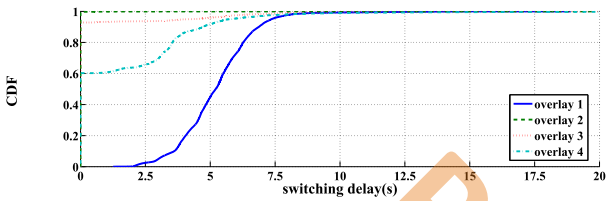

Fig. 14. Switching delay cdf.

with probability 0.8, a node experiences no delay at all. If in addition to the buffer reuse mechanism we also activate the newest priority scheduling policy (dashed-dotted lines), the improvement is negligible [in Fig. 13(c), it is just noticeable for high cdf values]. This is due to the fact that if the video diffusion processes of the overlays are adequately aligned, the buffer reuse action allows to inherit a very high percentage of previously gathered chunks, reducing to a minimum the number of new chunks to collect, so that the priority scheduling policy has a very low impact.

E. Different Scenarios

At last, we test the performance of the P2P-DASH platform equipped with all the proposed features, that is, augmented neighborhood, priority scheduling, buffer reuse, and the modified control strategy, for a different profile of user capacities and requests. We consider higher upload capacities with respect to the previous scenario, as shown in Table V. Moreover, according to the bit rates employed by Netflix [21] to distribute SD, HD, and full HD videos, we set the rates of the available representations at 1000, 2500, 4500, and 5800 kb/s.

Even in this new setting, the simulation results show that the P2P-DASH architecture grants very high values of DR and confined playback delays: Table VI shows that all overlays exhibit average DRs above 0.96. In addition, the average playback delays are comparable with those previously computed.

Moreover, Fig. 14 shows that the switching delay is successfully confined too, confirming the validity of our proposal when diverse operating conditions are examined.

VI. CONCLUSION

This paper has put forth a P2P-DASH live streaming architecture that delivers different representations of the same video content, each being streamed by a single swarm of a multi-overlay platform. This paper has been mainly centered on the reduction of the switching delay, a critical metric that markedly affects how smooth the streaming process is perceived by the user. The proposed system exhibits some peculiar features that aim at confining such delay and simultaneously guarantee a good functioning of the entire platform. There is a silent extension to the peer neighborhood. DASH segments that are completely received and stored within the peer-streaming buffer at transition time are reused. The scheduling of video chunks assigns higher priority to requests raised by peers that newly join the system or migrate to a different overlay. Both a core algorithm and a modified alternative are proposed, where the latter more conservatively allocates weaker peers to overlays that stream video at lower rates, therefore reducing their misalignment and improving the experienced performance.

System behavior is extensively investigated through simulation, and the major findings are as follows. The benefits attained by the introduction of the silent neighborhood mechanism and the buffer content reuse are the more impressive, and the more the streaming processes in adjacent overlays are aligned. Priority scheduling is the only approach able to reduce the switching delay regardless of such misalignment. The modified thresholds introduced by the alternative control algorithm guarantee better alignment and a very good viewing quality. We demonstrated that when such rate control algorithm is employed in conjunction with the silent neighborhood and the buffer reuse mechanisms, an excellent performance is achieved, i.e., high values of DR, low values of playback, and switching delays. However, a tradeoff is inevitable, as the users that exhibit the lowest upload capacity stream the video—with good quality—at a reduced rate.

REFERENCES

[1] Sandvine Inc., Waterloo, ON, Canada. (May 2015). *Global Internet Phenomena, Latin America & North America*. [Online]. Available: https://www.sandvine.com/downloads/general/global-internet-phenomena/2015/global-internet-phenomena-report-latin-america-and-north-america.pdf

[2] Sandvine Inc., Waterloo, ON, Canada, (May 2015). *Global Internet Phenomena, Asia-Pacific & Europe*. [Online]. Available: https://www.sandvine.com/downloads/general/global-internet-phenomena/2015/global-internet-phenomena-report-apac-and-europe.pdf

[3] *Information Technology—Dynamic Adaptive Streaming Over HTTP (DASH)—Part 1: Media Presentation Description and Segment Formats*, ISO/IEC Standard 23009-1:2014, 2014.

[4] T. Stockhammer, "Dynamic adaptive streaming over HTTP—Standards and design principles," in *Proc. ACM MMSys*, Feb. 2011, pp. 133–144.

[5] (May 2013). *Survey of European Broadcasters on MPEG-DASH*. [Online]. Available: http://dashif.org/wp-content/uploads/2015/04/Survey-of-the-European-Broadcasters-on-MPEG-DASH-Whitepaper-V2.1.pdf

[6] Z. Li *et al.*, "Probe and adapt: Rate adaptation for HTTP video streaming at scale," *IEEE J. Sel. Areas Commun.*, vol. 32, no. 4, pp. 719–733, Apr. 2014.

[7] L. De Cicco, V. Caldaralo, V. Palmisano, and S. Mascolo, "ELASTIC: A client-side controller for dynamic adaptive streaming over HTTP (DASH)," in *Proc. IEEE 20th Int. Packet Video Workshop (PV)*, Dec. 2013, pp. 1–8.

[8] C. Liu, I. Bouazizi, and M. Gabbouj, "Rate adaptation for adaptive HTTP streaming," in *Proc. ACM MMSys*, Feb. 2011, pp. 169–174.

[9] R. K. P. Mok, X. Luo, E. W. W. Chan, and R. K. C. Chang, "QDASH: A QoE-aware DASH system," in *Proc. ACM MMSys*, Feb. 2012, pp. 11–22.
[10] D. J. Vergados, A. Michalas, A. Sgora, and D. D. Vergados, "A fuzzy controller for rate adaptation in MPEG-DASH clients," in *Proc. IEEE 25th Int. Symp. PIMRC*, Sep. 2014, pp. 2008–2012.
[11] Y. Shuai, G. Petrovic, and T. Herfet, "OLAC: An open-loop controller for low-latency adaptive video streaming," in *Proc. IEEE ICC*, Jun. 2015, pp. 6874–6879.
[12] L. Natali and M. L. Merani, "A novel rate control strategy for adaptive video streaming in P2P overlays," in *Proc. IEEE GLOBECOM*, Dec. 2015, pp. 1–7.
[13] S. Lederer, C. Muller, and C. Timmerer, "Towards peer-assisted dynamic adaptive streaming over HTTP," in *Proc. IEEE 19th Int. Packet Video Workshop (PV)*, May 2012, pp. 161–166.
[14] L. Rovcanin and G.-M. Muntean, "A DASH-based performance-oriented adaptive video distribution solution," in *Proc. IEEE Int. Symp. BMSB*, Jun. 2013, pp. 1–7.
[15] G. Tian, Y. Xu, Y. Liu, and K. Ross, "Mechanism design for dynamic P2P streaming," in *Proc. 13th IEEE P2P*, Sep. 2013, pp. 1–10.
[16] A. Zambelli, "IIS smooth streaming technical overview," Microsoft Corp., Redmond, WA, USA, 2009.
[17] L. Yitong, S. Yun, M. Yinian, L. Jing, L. Qi, and Y. Dacheng, "A study on quality of experience for adaptive streaming service," in *Proc. IEEE ICC*, Jun. 2013, pp. 682–686.
[18] M. Zink, O. Künzel, J. Schmitt, and R. Steinmetz, "Subjective impression of variations in layer encoded videos," in *Proc. 11th IEEE-ACM IWQoS*, Jun. 2003, pp. 137–154.
[19] D. Wu, C. Liang, Y. Liu, and K. Ross, "View-upload decoupling: A redesign of multi-channel P2P video systems," in *Proc. IEEE INFOCOM*, Apr. 2009, pp. 2726–2730.
[20] L. Natali and M. L. Merani. (2015). "Adaptive streaming in P2P live video systems: A distributed rate control approach." [Online]. Available: http://arxiv.org/abs/1510.04472.
[21] Netflix. *Internet Connection Speed Recommendations*, accessed on Oct. 2015. [Online]. Available: http://www.netflix.com
[22] Swisscom. *Swisscom Internet Offers*, accessed on Oct. 2015. [Online]. Available: http://www.swisscom.com
[23] Orange. *Orange Offres Internet*, accessed on Oct. 2015. [Online]. Available: http://www.orange.fr
[24] T.net. *T.Net Connettivitá e Voce*, accessed on Oct. 2015. [Online]. Available: http://www.tnet.it
[25] "Second Quarter, 2014 State of the Internet Report," Akamai Technol., Cambridge, MA, USA. [Online]. Available: http://www.akamai.com/stateoftheinternet/
[26] S. Medjiah, T. Ahmed, and R. Boutaba, "Avoiding quality bottlenecks in P2P adaptive streaming," *IEEE J. Sel. Areas Commun.*, vol. 32, no. 4, pp. 734–745, Apr. 2014.

Laura Natali received the M.Sc. degree in computer engineering from the University of Modena and Reggio Emilia, Modena, Italy, in 1997, where she is currently pursuing the Ph.D. degree with the ICT International Doctorate School.

Her current research interests include multimedia networking and peer-to-peer and live video streaming.

Maria Luisa Merani (M'93–SM'03) received the M.Sc. (*summa cum laude*) and Ph.D. degrees in electrical engineering from the University of Bologna, Bologna, Italy, in 1987 and 1992, respectively.

She was with the Computer Science Department, University of California at Los Angeles, Los Angeles, CA, USA, in 1991. She has been with the Department of Engineering Enzo Ferrari, University of Modena and Reggio Emilia, Modena, Italy, since 1993, where she is currently an Associate Professor. She has authored the textbook entitled *Hands-on Networking: From Theory to Practice* (Cambridge University Press, 2009). Her current research interests include radio communications, with an emphasis on wireless multimedia, and distribution architectures for adaptive video streaming.

Dr. Merani has been very active within the Technical Program Committee of several flagship conferences, including the IEEE International Conference on Communications, the IEEE Globecom Conference, the IEEE Wireless Communications and Networking Conference, and the IEEE International Symposium on Personal, Indoor and Mobile Radio Communications. She served as the Technical Program Co-Chair of the Wireless Networking Symposium of the IEEE Globecom Conference in 2007 and 2009, and of the Second IEEE International Symposium on Wireless Communication Systems in 2005. She was the General Chair of the IEEE International Symposium on Wireless Pervasive Computing in 2010. She served as an Editor of the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS from 2010 to 2013.