A P2P Platform for Real-Time Multicast Video Streaming Leveraging on Scalable Multiple Descriptions to Cope with Bandwidth Fluctuations

Lorenzo Favalli[1], Marco Folli[1],
Alfio Lombardo[2], Diego Reforgiato[2], Giovanni Schembra[2]

[1]Dipartimento di Elettronica, University of Pavia, Via Ferrata 1, 27100 Pavia - ITALY
E-mail: : {lorenzo.favalli, marco.folli}@unipv.it

[2]Dipartimento di Ingegneria Elettrica, Elettronica e Informatica, University of Catania, V.le A. Doria 6, 95125 Catania - ITALY
E-mail: : {lombardo, diegoref, schembra}@diit.unict.it

Abstract [1] *In the immediate future video distribution applications will increase their diffusion thanks to the ever-increasing user capabilities and improvements in the Internet access speed and performance. The target of this paper is to propose a content delivery system for real-time streaming services based on a peer-to-peer approach that exploits multicast overlay organization of the peers to address the challenges due to bandwidth heterogeneity. To improve reliability and flexibility, video is coded using a scalable multiple description approach that allows delivery of sub-streams over multiple trees and allows rate adaptation along the trees as the available bandwidth changes. Moreover, we have deployed a new algorithm for tree-based topology management of the overlay network. In fact, tree based overlay networks better perform in terms of end-to-end delay and ordered delivery of video flow packets with respect to mesh based ones. We also show with a case study that the proposed system works better than similar systems using only either multicast or multiple trees.*

Keywords: Video-Transmission, Real-Time, P2P, MDC, QoS

1. INTRODUCTION

Transmission of video sequences over both the Internet and wireless networks is becoming increasingly popular. As the bandwidth available to users increases, video services are becoming a commodity offered by many Internet multimedia service content providers.

[1] This work was partially supported by the Italian MIUR PRIN projects “Profiles” and “Sorpasso”. Moreover, the work leading to this invention has benefited from a fellowship of the Seventh Framework Programme of the European Community [7_ PQ/2007-2013] regarding the Grant Agreement n. PIRG03-GA-2008-231021.

In this perspective, a lot of research work has been reported in the previous literature mainly regarding transmission of live streaming video flows in multipoint fashion. The traditional approach based on either multicast at the IP level [13,14] or content distribution networks (CDN) [12] motivated the choice of many servers/proxies to be located as close as possible to the users to reduce bandwidth usage and latency. This architecture is becoming outdated as it lacks flexibility, that on the contrary is strongly required in such dynamic world. So, the most promising and attractive alternative today is represented by peer-to-peer (p2p) networks [21], which are self-organizing overlay networks that aggregate large numbers of heterogeneous computers called nodes or peers. In p2p systems, peers communicate directly to each other for data sharing and data exchanging. The characteristics of p2p networks make them a very good choice for video broadcast over IP networks.

Differently from traditional file-sharing p2p networks, support of video content streaming highly exploits application-level multicast (ALM) to make an efficient use of bandwidth. The feasibility of this approach has been proved by systems such as Coolstreaming [22], PPlive [23], ESM [24]. Video streaming systems based on the p2p concept have already achieved a number of large-scale deployments, accommodating tens of thousands of simultaneous users [20].

In this context, a very huge number of research papers have been focused on the vulnerability of the overlay network. It is mainly due to the peer churn [15,16,17,18,19], that is, the instability due to nodes that dynamically join and leave the network.

To overcome this problem, the traditional strategy is to use a multiple tree topology (forest topology) and a multiple description coding (MDC) video source [31] [32]. In this way each node receives a position in each tree; the encoded video is subdivided in different substreams, all individually decodable, each at a lower quality than the original. These substreams are routed to each peer through independent paths along different trees. If all the descriptions are received, then the single coded video stream can be decoded with the maximum quality. This approach is very attractive since it is possible to exploit the inherent protection provided by path diversity among the different descriptions. Thanks to these properties, MDC has been integrated for example in CoopNet [33] and SplitStream [2]. However, although these systems are quite robust in presence of peer churn, at the same time, they organize the overlay network not taking into account the available bandwidth of the underlying network. So potential bottlenecks might occur. Moreover, these systems do not take into account the amount of bandwidth to reserve to each description.

Several other works pointed their attention to strategies aimed at creating and managing the overlay network topology and organizing it as a tree, a forest, or a mesh (see [9,10,11]). A good comparison between these different approaches can be found in [8], where it is shown that mesh structures are more efficient in terms of bandwidth utilization, load balancing and network stability, while tree and forest structures better perform in terms of end-to-end delay and ordered delivery of video packets flow.

On the other hand, in the immediate future a large slice of new Internet applications will be constituted by multipoint real-time video communications, as for example videoconference, tele-teaching, remote training sessions, cooperative work [7,6,1,2,3]. These applications share the following peculiarities:

– the number of participants is not huge, but of the order of hundreds;

– the most stringent requirement is end-to-end delay;

– usually participants enter the network at the beginning of the session, and leave it at the end, and consequently the overlay network does not suffer of instability due to peer churn;
– coding is made in real-time, and therefore the perceived quality at destination is strongly influenced by instantaneous bandwidth fluctuations.

With all this in mind, the target of this paper is to define a platform for multipoint real-time video distribution in the current Internet. For the above observations, in the considered application scenarios, peer churn is not a main issue. So, for the sake of simplicity, peer departure and peer arrival events, although managed by our platform described in Section 3, will not be accounted in the performance analysis because considered as rare events during the normal working of the platform. The impact of peer churn in transient periods is so considered out of the scope of this work, and planned in future works.

On the contrary, the scope of the paper regards bandwidth oscillations of the links in the underlying network, which can strongly deteriorate performance even in stable networks. This is an important problem that has received no attention in the previous literature. So, the key idea at the basis of this work is to apply scalable multiple description coding, hierarchical video streaming and topology management as a solution to tackle problems of bandwidth variation and end-to-end delay minimization. Thanks to the scalability of each description, only using two descriptions, eight possible levels of quality are possible at destination, depending on the actual combination of what is received by each peer.

A key feature of our system is that bandwidth bottlenecks are avoided when constructing the overlay network. Moreover, as we will show numerically, the system will result fair: each peer is equally served and this does not depend on the single bandwidth characteristics.

Likewise most of the systems currently used for our target applications, we use a centralized control realized with a node, in the following indicated to as the Topology Manager, to authenticate users and coordinate the session. Moreover, we assume that this node will also be in charge to determine the best video coding parameters to match actual network conditions.

We stress that the accent in our work is on the bandwidth constraints rather than on churn, and the target is to show the improved resilience and flexibility of the scalable multiple descriptions combined with a topology management algorithm.

A case study is presented to analyze bandwidth and protocol behaviors of the used topology management algorithms, and perceived quality for a given video sequence transmitted using our platform is evaluated via simulation.

More in detail, this paper is organized as follows. Section 2 introduces the scalable multiple description coding and the MDC algorithm we have used. Section 3 gives an overall view of the proposed platform. Details about the source and the generic peer are given as well as the algorithms adopted to create the multiple description trees. Finally, Section 4 shows the performance of the proposed system evaluated in the considered case study, analyzing both the protocol temporal behavior and the perceived quality in terms of PSNR of the video sequence received at destination. Finally, Section 5 ends the paper with some ideas on possible future works.

2. Related Work

In this section we present some background about multiple description coding (Section 2.1) and the multiple description coding (MDC) algorithm used in this paper (Section 2.2).

2.1 MDC Background

The concept of multiple description coding goes back to the end of the 1970s with theoretical works [45] on transmission over parallel channels. The idea was then applied in various ways to voice coding (e.g. the work in [46]). Application to video has been later introduced as the available network bandwidths made this viable, and many variants have been proposed since then as is well reported in the overview paper by Goyal [31]. For recent in-depth overview of MDC schemes and applications please refer to [48] and references therein. The basic concept of MDC is to generate independent decodable substreams (the descriptions) from the same data stream so that each of them has a lower quality, but their combination allows recovery of the original stream quality. Ideally this process should be performed without loosing bandwidth efficiency. However, real systems need to introduce in each description an overhead due to standard-related control information (e.g. sequence, picture, block headers) since they must be independently decodable. The key point of this approach is that the descriptions may be transmitted along independent routes thus exploiting one of the best known ways in telecommunications to achieve robust transmission: path diversity. Ideally, the higher the number of paths, the higher the resilience to losses. Practically speaking, we need to take into accounts that each description adds redundancy and then reduces coding efficiency [53], [54]. Furthermore it is possible to show (see for example [49],[4],[5]) that the most evident advantage is obtained going from a single description to two descriptions, while a further increase in the number of descriptions provides decreasing marginal gains.

A simple MDC scheme is the one proposed in [47] in which odd and even frames of a video sequence are divided into two separate, individually decodable descriptions, each having half the frame rate and that can be decoded using standard receivers. Efficiency of this scheme is clearly affected by the temporal correlation among the descriptions. A way to reduce this redundancy is described in [34]. Such a scheme, called Multiple Description Motion Compensation (MDMC), designs temporal predictors that exploit the temporal correlation not only within a description, but also across the descriptions.

Another simple MDC method is presented in [35]. It is based on the spatial subsampling of the original video sequence along rows and columns of each frame by means of a polyphase spatial sub sampler (PSS-MDC). In this case, inefficiencies are due to the spatial correlation among the descriptions. An improvement to this scheme has been introduced in [36], where two of the four subsequences are estimated from their neighboring subsequences; then the original and the predicted sequences are coded, respectively, using a standard coder and via discrete cosine transform (DCT) and universal variable length coding (UVLC).

The main drawback of the MDC schemes is that they solely aim at increasing robustness by exploiting link diversity, without addressing other important transmission challenges, such as bandwidth variations or device heterogeneity. A solution for the last two problems is obtained by using some sort of scalability that, on the contrary, lacks of robustness. The complementarily of the two approaches has been exploited to implement Multiple Description Scalable Coding (MDSC) techniques ([39], [40]). A new type of MDSC in which multiple

descriptions are obtained via spatial or temporal algorithms as well as SNR algorithms is introduced by several authors using wavelet-based coding (see for example [41] [42],[43]).

These methods, based on the wavelet transform, generate scalable multiple descriptions using an SNR algorithm applied at a temporal, spatial or hybrid temporal/spatial algorithm but are not compliant with the standard H.264 coder as it does not implement wavelets.

On the contrary, a simple PSS-MD can be implemented on top of almost all the coders, while the one proposed by [36] requires both a coder that supports differential prediction and additional implementation of the DCT and UVLC algorithms, and this causes substantial modification of the standard coders.

2.2 The ILPS-MDSC algorithm

In order to develop a system as much compliant as possible with standard codecs, the algorithm named Inter Layer Prediction Spatial - Multiple Description Scalable Coding (ILPS-MDSC) was introduced in [37], [38]. This algorithm leads to scalable multiple descriptions using a pre- and post-processing scheme based on spatial subsampling, and exploits some of the features provided by scalable extension of the H.264 coder.

As details of the algorithm may be obtained from [37], [38], here we limit description to summarize its main characteristics.

The first step is to apply a polyphase spatial subsampling pattern to the incoming frames. The reader may note that different subsampling patterns could be possible and an arbitrary number M of descriptions could be generated, at the cost of additional complexity and reduced coding efficiency. In this paper we refer to the simplest one, with a decimation of two along rows and columns that produces four different subframes.

These four subsequences are highly correlated and, consequently, highly redundant. This correlation is the key factor to exploit MDC as a reliable tool to protect information against losses. Nonetheless we may think to preserve only part of this redundancy while introducing scalability, exploiting some prediction mechanism.

In order to develop an MDC scheme as much compatible as possible with a standard H.264 coder, the idea has been to exploit some of the features provided by the scalable extension of the H.264, called H.264/SVC [44]. Specifically, the coding efficiency can be improved by adding inter-layer prediction tools that can be chosen on a macroblock or sub-macroblock basis, allowing an encoder to select the coding mode that gives the highest coding efficiency.

In our approach, MDSC is achieved reducing the number of descriptions by grouping two subsequences to form a single description. Scalability within each description is obtained by assigning the two subsequences of each description, respectively, one to the base layer and the other to the enhancement layer of the scalable coder as depicted in Fig. 1, and then exploiting the inter-layer prediction to eliminate redundancy. As detailed in [38] this approach allows to strongly reduce the extra bitrate introduced by the MDC approach. At the end, we obtain two coarse grain scalable (CGS) descriptions which are then transmitted.

At the decoder side, each description is first decoded independently and then, in the post processing part, the two descriptions are merged. Given the scalable structure of each description, the description is reconstructed by first decoding the base plus enhancement layer stream and then extracting the base layer in order to decode also the other subsequence In the post-processing part, the original sequence is obtained by merging the descriptions.

If some information is lost, the missing pixels are reconstructed using the interpolation from the nearest pixels depending on what has been actually lost: a base layer, both base layers, a whole description, remembering that each substream (either base or enhancement) has been obtained from the subsampling of the original sequence.

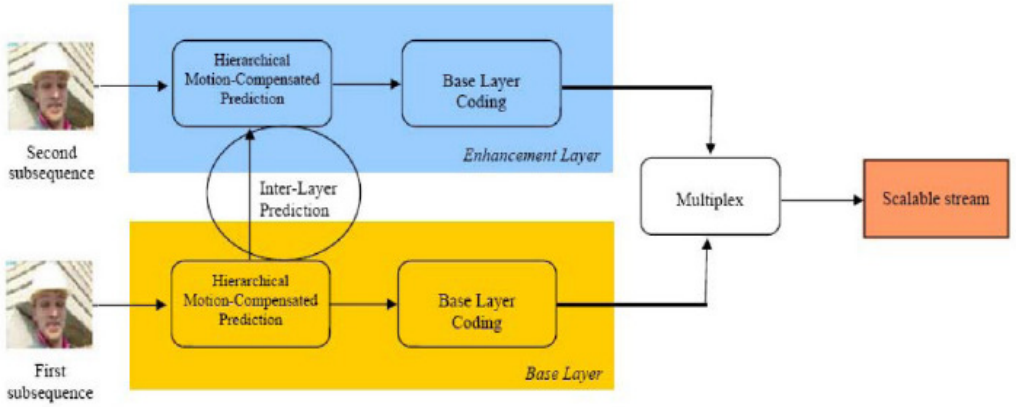


Fig. 1 ILPS-MDSC algorithm structure.

3. System Description

The target of this section is to describe the system we propose in this paper. It is an adaptive live video broadcast platform, where a video source distributes a video stream to a number of clients in a multipoint fashion. In this general description, we assume that video is encoded with an H.264 multiple description encoder with M descriptions. Each description consists of a base layer and an enhancement layer.

Multipoint communication is achieved by applying a p2p approach, configuring a multiple tree-structured overlay network. Clients are organized in M trees, one for each description. The root of each tree is the source, and specifically the video encoder of the description to be sent on that tree. In the following, clients will be also referred to as nodes of the trees, or peers.

3.1 Architecture of the Video Source

The architecture of the video source is shown in Fig. 2. As it can be seen, its core is the H.264/SVC Video Encoder, implementing the ILPS-MDSC algorithm described in Section 2.2, in order to achieve hierarchical multiple description coding. It receives a raw video stream in CIF format as its input and produces an H.264 video flow, made up of M different and independent descriptions, each to be forwarded to one of the M multiple-description trees. The raw video from the source is first sub-sampled and M descriptions (each corresponding to a QCIF video) are thus generated (each description has one base layer and one enhancement layer). In Fig. 2 they are indicated as $D_1,...,D_M$. For each multi-description tree, a Replicator is used in order to create as many streams as the number of the source's children, hereafter indicated as F_S (F_S is at most equal to the source fan-out parameter $F_S^{(FO)}$). The bits produced at each frame interval are queued in a Buffer for each downlink connection toward the F_S direct

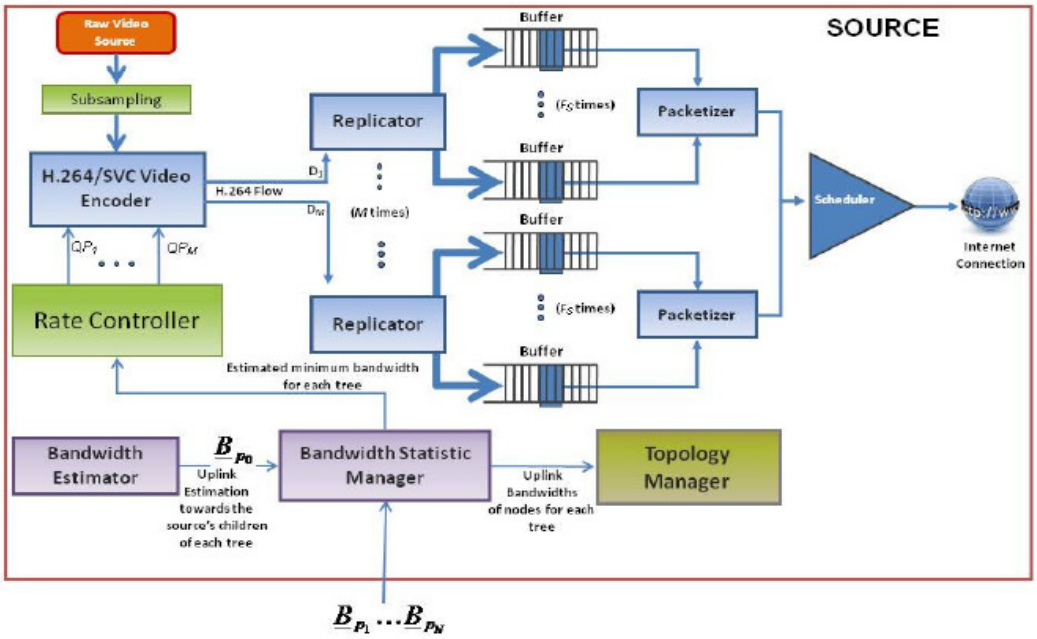


Fig. 2 Architecture block diagram of the video source.

children, in order to reduce possible losses due to bandwidth fluctuations. Each of the F_S streams coming from these buffers ends up into a Packetizer; the resulting M X F_S streams end up to the Scheduler, which applies a weighted round robin algorithm and sends the descriptions with an amount of bandwidth proportional to the uplink bandwidth estimated towards each children of the source in each tree.

The Rate Controller obtains the needed information about the bandwidth to be used on each tree from the Bandwidth Statistic Manager. Let us indicate the current number of peers in the network as N. The Bandwidth Statistic Manager periodically receives the source's children uplink bandwidth estimation from the local Bandwidth Estimator and, at the same time, the uplink bandwidth values estimated by all the other peers on each tree (see $\underline{B}_{P1}$, …, $\underline{B}_{PN}$ in Fig. 2 or $\underline{B}_{Pi}$ in Fig. 4). Specifically, let $\underline{B}_{pi} = [\ \underline{B}_{pi}^{(1)}, \ldots, \underline{B}_{pi}^{(M)}, \underline{B}_{pi}^{(up)}\ ]$ be the array received by the generic ith peer, containing both the estimated uplink bandwidth values toward all its children on all the M multiple-description trees, and its overall uplink bandwidth. Therefore, M x $|\underline{B}_{Pi}|$ is the number of values the bandwidth Statistic Manager receives each T_T seconds by each peer; as discussed in Section 3.3, the reader notice that, even for high values of M, this causes a negligible overhead as compared with the large amount of data constituting the video stream.

Every T_P seconds, the source changes the encoding target for each tree. More specifically, for the generic tree j, at the slot n it uses the set of bandwidths received by each peer, $B_{pi}^{(j)}(n)$, smoothed with an exponentially-weighted moving average (EWMA) filter defined as follows:

$$\overline{B}_{p_i}^{(ENC,j)}(n) = \beta^{(ENC)} \cdot \overline{B}_{p_i}^{(ENC,j)}(n-1) + (1-\beta^{(ENC)}) \cdot B_{p_i}^{(j)}(n) \qquad (1)$$

Where $\overline{B}_{pi}^{(j)}$ (n – 1) and $\overline{B}_{pi}^{(j)}(n)$ are the filtered bandwidths at the (n - 1)th and nth update events, $B_{pi}^{(j)}(n)$ is the instantaneous uplink bandwidth at the nth slot, and $\beta^{(ENC)}$ is the EWMA control parameter ranging from 0 to 1; values of $\beta^{(ENC)}$ close to 1 give more importance to the history of the bandwidth process, achieving a process that is less sensitive to high "frequencies" (therefore able to smooth the short-term variations), but having slower responses to bandwidth changes; conversely, very low values of $\beta^{(ENC)}$ give more importance to recent measures, achieving greater responsiveness. The encoding target for the generic tree *j* has been set as follows:

$$B^{(ENC,j)} = 0.9 \cdot K_j \cdot \min\{B_{p1}^{(j)}(n), \ldots, B_{pN}^{(j)}(n)\} \quad (2)$$

Where K_{j-1} is a design parameter that allows to reserve some amount of bandwidth for the next trees. The importance of its choice will be discussed in Section 4.3. The term 0.9 is used to further decrease the coding rate in order to absorb potential sudden uplink bandwidth oscillations of peers with respect to the estimated bandwidth.

The amount of bits to be used for each description to encode the base and the enhancement layers, and the relative encoding quality, are determined by the Rate Controller through the quantizer parameter *QP*, chosen for each level on each description in the range between 0 and 51: the greater the *QP* value, the poorer the encoding quality.

In addition, in order to make the overlay network more robust to sudden bandwidth oscillations, the 90% (which is the 0.9 term of (2)) of the minimum bit rate available on each tree of the overlay network is used as the target bit rate of the base layer encoder of the description to be sent on that tree. The rate for the whole description (base+enhancement layers) is determined as the 100% of the bandwidth available at the 40% of the nodes with the highest bandwidths.

Another important block in the video source architecture is the Topology Manager, which decides and maintains the network topology of each tree by deciding the position of each peer within the tree. The Topology Manager can be implemented as centralized unit, as in Coopnet [33] or distributed, as in [52]. In the case of a centralized Topology Manager, it can reside in the same host of the source, or in any other host in the Internet. In the other case, the Topology Manager can be distributed in *M* different units, one for each tree. In Section 3.3 we will discuss about the impact of this choice in terms of bandwidth overhead due to the distributed choice. However, we have observed in our experiments that the workload of the Topology Manager, even for large networks with some thousands of peers, is sustainable by a single host, given that topology refresh events are run every T_T seconds (in our experiments we have used T_T = 30 seconds. For this reason, also taking into account the overhead introduced by the distributed approach besides possible problems of synchronization between different Topology Manager entities, we decided to apply the centralized approach, and in the following we will limit our description to it. However, let us note that an extension to the distributed approach is very easy and involves implementation issues only. To adapt the overlay network topology to the bandwidth variations of the underlying network, the Topology Manager uses the uplink bandwidths of all the peers collected by the Bandwidth Statistic Manager. Moreover, it implements an algorithm to manage both peer arrival and departure events. In detail, the Topology Manager is responsible of the following tasks:

– When a new peer arrives, this peer issues an admission request to the Topology Manager in order to receive both its position in each tree, and the IP address of the peers that will be its parents. The new peer is temporally inserted as a leaf until the next topology refresh. Moreover, the Topology Manager sends the IP address of the new peer to its parent node on each tree. In this way, the parent node in each tree inserts the IP of the new child in the Replicator of the relating tree, in order to create another copy of the packets for the new child. The parent nodes create also a new buffer for their new child and communicate to the Packetizer to handle the new node child. Finally, the parent nodes insert the IP address of their new child in the Bandwidth Estimator to start estimation with it.
– The case of departures nodes has been handled as in CoopNet [33]. If a node leaves gracefully, it contacts the Topology Manager, telling it to find a new parent for each of its child nodes. Instead, when a node leaves ungracefully, its children notice its absence by monitoring their own incoming packet loss rate. The children are then responsible for contacting the Topology Manager and requesting a new parent. In both cases, the parent nodes delete the IP address of the departing peers from their Replicators. Moreover, they destroy the relating buffers and communicate to the Packetizer to stop serving that child node. The IP address of the departing peer is also removed from the Bandwidth Estimator.
– When the Topology Manager receives information about the uplink bandwidth of each peer, periodically (every T_T secs) it updates and optimizes the trees. To this end, using the values of $\overline{B}_i^{(up)}$, for each $i \in [1,N]$, received by the Bandwidth Statistic Manager, it implements an algorithm to manage run-time topology modifications after bandwidth variations.

In order to avoid excessively strong oscillations of both the encoding quality, the network topology and the system behavior, the bandwidth values $B_{pi}^{(up)}(n)$ are smoothed with an exponentially-weighted moving average (EWMA) filter like that defined in (1).

To this end, the Topology Manager EWMA ($EWMA_{TM}$) uses a specific control parameter $\beta^{(TM)}$.

Fig. 3 summarizes the smoothing behavior of the two EWMA filters used to smooth the bandwidth passed to the encoder and the one used by the Topology Manager.

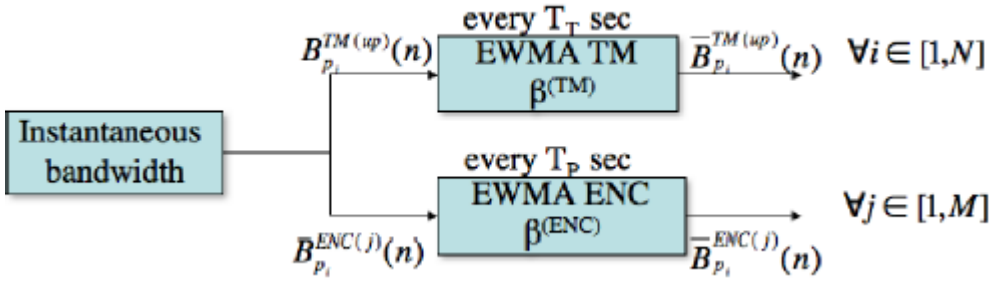


Fig. 3 EWMA filters applied to the instantaneous bandwidths.

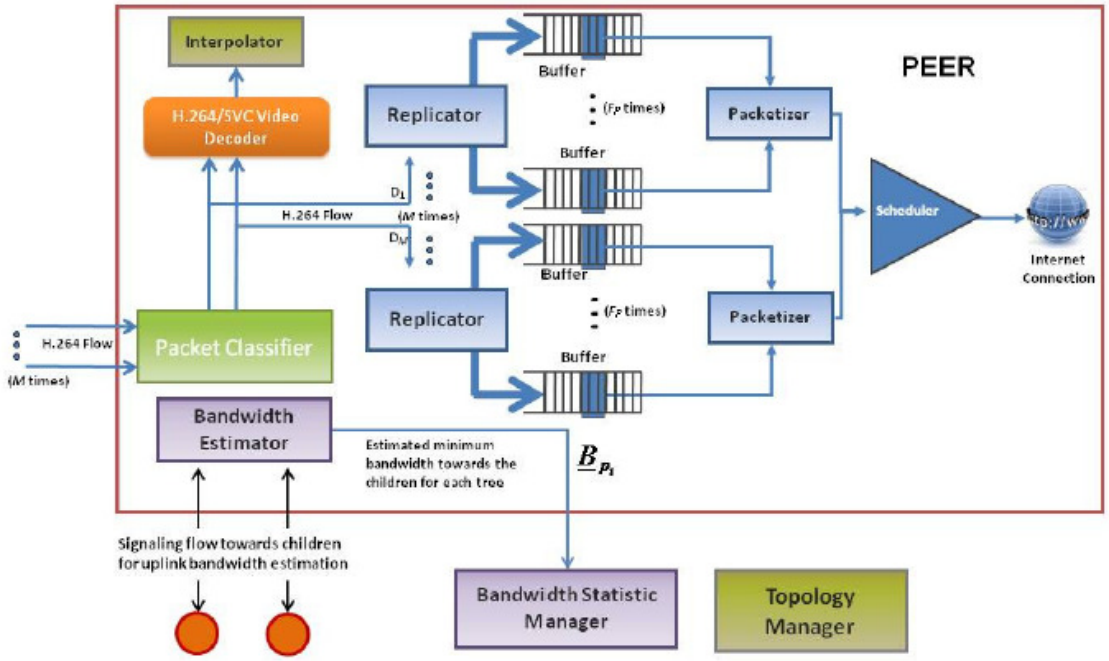


Fig. 4 Architecture block diagram of the generic peer p_i.

3.2 Architecture of the Generic Peer

It has been designed to mainly perform two functions:

1. video play-out and forwarding;
2. bandwidth estimation.

The architecture of the generic peer is shown in Fig. 4. Each peer receives IP packets containing M video descriptions which are first managed by the Packet Classifier and then sent to both the Decoder and the Replicators.

In the Decoder block an interpolator takes the sub-sampled descriptions and generates the original CIF format video, with a quality obviously dependent on the number of descriptions and, for each description, the number of levels that are available at the decoding time.

At the same time, they are transmitted along the M multiple-description trees. As for the case of the source, a Replicator block is needed for each multiple-description tree, in order to create at most F_P streams, with $F_P \leq F_p^{(FO)}$, one for each child of the generic peer. The bits received at each frame interval are queued in the Buffers. Of course, if estimation has been made correctly, the base layer of each description will find enough bandwidth on the transmission channels, and will not incur in any packet losses. However, given that the encoder target is decided on the bandwidth filtered with the $EWMA_{ENC}$ filter, and the decision is done at each T_P seconds, due to bandwidth variations some loss is possible in the Buffers.

Each of the F_P streams coming from these Buffers ends up into a Packetizer that, using the information received by the Topology Manager of the source about the IP addresses of the peer's children on each tree, is responsible for packet creation and transmission on the Internet.

Finally, packets are sent to the Scheduler which, as the one discussed for the source, applies a weighted round robin algorithm to serve flows of each description for each child, with an amount of bandwidth proportional to the estimated uplink bandwidths.

In addition to video play-out and forwarding, another important function performed by each peer is the uplink bandwidth estimation towards their children on each distribution tree. This task is performed by the Bandwidth Estimator block, which periodically sends the estimated bandwidths, the array $\underline{B}_{pi}$ of Fig. 4, to the Bandwidth Statistic Manager of the source, as discussed for the video source diagram of Fig. 2. The purpose of this function is to achieve the best performance in small-size networks. Conversely, if we consider large networks, the algorithm can be slightly modified to be more scalable, but with worse performances. This part of the system is completely general and any bandwidth estimation algorithm can be plugged in. The choice of it is beyond the purpose of this paper. For example, in addition to the basic estimation algorithm, a more sophisticated one to predict the bandwidth in the short or middle term (e.g. [30,29]) can be applied.

3.3 Tree construction Algorithm

In this section we will describe the algorithm we propose to construct and manage the overlay network topology. The algorithm is defined for a forest network constituted by M trees, each to be used to support one hierarchical description of the MDC video flow. As already said in Section 3, the MDC video source will be the root of all the trees; this algorithm will assign a position in each distribution tree to all the peers in the network. The target is to obtain an overlay network which is as stable as possible to peer bandwidth variations, and supporting bandwidth uplink sharing of each peer among the M trees.

Let us now define the algorithm to construct the M trees.

Tree 1 construction algorithm

To construct the first tree, we consider the bandwidth that the generic peer p_i reserves for the Tree 1; it is given by its overall uplink bandwidth:

$$\rho_{p_i}^{(1)} = \overline{B}_{p_i}^{(TM,1)} \; \forall i \in [1, N] \qquad (3)$$

The TM chooses the $F_S^{(FO)}$ peers having the highest available bandwidth calculated as in (3), and connects them as children of the source. Then, for each of these peers, the same operation is repeated, choosing the next $F_p^{(FO)}$ peers with the highest available uplink bandwidth. This algorithm is run recursively until all the peers get a position in the tree.

Tree j construction algorithm

To construct the generic Tree j, we consider the available bandwidth for the generic peer p_i given by its uplink bandwidth, reduced by the bandwidth used in the trees 1, …., j-1, that is:

$$\rho_{p_i}^{(j)} = \rho_{p_i}^{(j-1)} - B^{(ENC,j-1)} \cdot F_{p_i}^{(j-1)} \qquad (4)$$

where:

– $B^{(ENC,j-1)}$ is the estimated average source encoding rate on the tree j - 1; it is defined in (1);

– $F_{pi}^{(j-1)}$ is the number of children of the peer p_i on the tree j - 1.

Let us note that, as already said in Section 3.1, the MDC source encodes each description using a rate further decreased with a factor 0.9 to absorb potential sudden uplink bandwidth oscillations of peers with respect to the estimated bandwidth.

Once each peer reserves a portion of its bandwidth to the considered Tree j according to (4), the TM chooses the $F_S^{(FO)}$ peers having the highest available bandwidth $P_{Pi}^{(j)}$, and connects them as children of the source. Then, for each of these peers, the same operation is repeated, choosing the next $F_P^{(FO)}$ peers with the highest residual bandwidth. This algorithm is run recursively until all the peers get a position in the tree.

Let us note that the transmission overhead generated to perform this algorithm is small as compared to the transmitted video traffic: the only amount of data exchange to implement this protocol is due to the communications of the time-variant values of the estimated bandwidth from peers to the Bandwidth Statistic Manager, and the IP address of the parent on each tree from the Topology Manager to each peer. Taking into account that this occurs at each T_T seconds, the overhead bandwidth values are:

$$B_{Peers \to BM}^{(OH)} = \frac{S_B \cdot 8 \cdot M \cdot N}{T_T} \text{ bits/sec} \quad (5)$$

$$B_{TM \to Peers}^{(OH)} = \frac{S_{IP} \cdot 8 \cdot M \cdot N}{T_T} \text{ bits/sec} \quad (6)$$

where S_B is the number of bytes needed to represent the bandwidth value, S_{IP} is the number of bytes needed to represent an IP address, i.e. $S_B = 4$ and $S_{IP} = 4$. In the case the Topology Manager is distributed in M entities, one for each tree, an additional overhead is introduced for the transmission of information from the Bandwidth Statistic Manager to the TM entity managing the first tree, and from the generic $(j-1)$-th TM entity and the j-th one, for each $j \in \{2,\ldots,M\}$. In this case we have an additional overhead bandwidth for each TM entity given by:

$$B_{TM}^{(OH)} = \frac{S_B \cdot 8 \cdot N}{T_T} \text{ bits/sec.} \quad (7)$$

4. Case study

In this section we evaluate performance of the proposed system. To this end, taking into account that peer churn is not a matter for the considered scenarios of real-time multicast video streaming as discussed in the Introduction section, we will carry out a steady-state analysis, assuming that the number of peers in the network remains constant for the whole duration of the analysis. In this way our study does not depend on the particular algorithm used to manage the topology structure when peer arrivals or departures occur. Let N be the number of peers.
Let us note that this assumption does not influence the generality of the achieved results. In fact, peer churn can be considered as another cause of bandwidth variation, and therefore paper conclusions with this assumption can be extended to the case of peer churn.

In our analysis we consider a video sequence of 30 minutes, relative to an e-learning session captured from the Italian broadcast television. We assumed that it is transmitted to a number of $N = 1000$ clients accessing the Internet through different Internet Service Providers (ISP) and links, with different bandwidth capabilities.

As discussed so far, we assume that access links have downlink bandwidth much greater than the uplink bandwidth. Due to the intrinsic oscillations of the bandwidth for such a kind of links, and the fact that the considered video distribution application shares this bandwidth with other applications on the same host and with other hosts in the same local network, the net bandwidth available runtime to this application presents a behavior which may have a strong impact on the performance of the whole system.

In order to well capture a real scenario, we have considered clients with heterogeneous Internet access links, belonging to $C = 5$ different classes, organized with decreasing average uplink bandwidth values.

The source belongs to another class, with different statistics, specifically with a much greater average uplink bandwidth value. Randomly assigning classes to the 1000 peers to simulate a real case, we obtained:

– 194 peers of class 1;
– 203 peers of class 2;
– 204 peers of class 3;
– 187 peers of class 4;
– 212 peers of class 5.

As far as the source settings are concerned, as already said in Section 2.1, in [49] it was demonstrated that the biggest advantage is obtained using no more than two descriptions; a further increase in the number of substreams only leads to a marginal increment in performance as compared with the introduced bandwidth overhead. For this reason we will use $M = 2$ descriptions (therefore $M = 2$ trees). The other source parameters were set to $T_T = 30$ seconds, $T_P = 10$ seconds, $\beta^{(TM)} = 0.9$, $\beta^{(ENC)} = 0.7$.

Finally, we have considered a fan-out parameter $F = 2$ equal for all the nodes. In this way the network is constituted by two trees, each with 10 levels. Each level is full, except for the last one which contains 511 leaves. For further motivation on using these parameters, the reader is referred to [26],where a modified version of this system plugged with an FGS encoder has already been applied successfully.

The source coder is built on top of the H.264/SVC coder version 9. Further details on its implementation are given in Section 2.2. The main options provided by the original coder have been set as follows:

– YUV 4:2:0 CIF sequence with 25 fps
– 1/4 pixel accuracy for motion estimation
– a single reference frame
– GOP size of 8
– I frame every second (i.e. every 25 frames)
– 16x16, 16x8, 8x16, 8x8 inter-prediction blocks with SAD metric
– Context-adaptive binary arithmetic coding as entropy coding (CABAC)

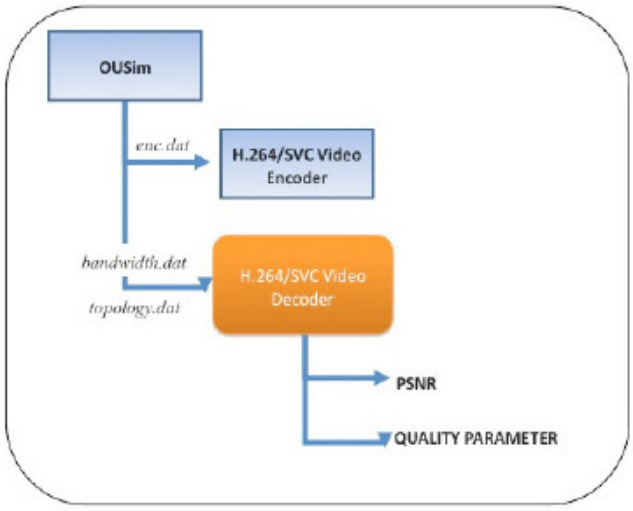


Fig. 5 Structure of the performance analysis system.

Experiments were conducted using a structured simulator, as depicted in Fig. 5. Simulation starts with the run of the block *OUSim* (Overlay and Underlying Network Simulator). Its targets are:

– randomly assign the number of users for each class;
– simulate the available uplink bandwidth for each peer, according to the class it belongs to;
– simulate the overlay network, by implementing the Topology Manager with the Tree Construction algorithm, and the consequent topology variations for each tree.

The output of the OUSim block is constituted by three files:

1. *enc.dat*, a trace file containing the bandwidth values calculated by the Bandwidth Statistic Manager of the source at each T_T seconds, and passed to the H.264/SVC Video Encoder to perform the video encoding operation;
2. *bandwidth.dat*, a trace file containing the available uplink bandwidth for each peer, generated with the bandwidth generation process described in Section 4.1;
3. *topology.dat*, a trace file containing the position of each peer on each tree for each time interval.

The first file is used by the Video Encoder block to achieve the encoded video traces. The Video Decoder block, using the file bandwidth.dat and the file topology.dat, evaluates the amount of bits that each peer receives for each frame and for each description, and decodes video applying these information. The output of the Video Decoder block are:

1. the PSNR trace for each frame received by each peer, taking into account the number of received descriptions;
2. the trace containing an overall Quality parameter for each peer, according to the definition in (9).

The *OUSim* simulator block is publicly available for download at [28]. Other sections of this chapter are Section 4.2, which analyzes the protocol behavior using our overlay network

simulator and Section 4.3 which presents some numerical results achieved by our platform and provides some hints to design the platform.

4.1 Bandwidth generation process

Let p_i be the generic peer and $c_i \in \{1, \ldots, C\}$ its class. In this section we describe the algorithm we used to generate the uplink bandwidth behavior capturing first- and second-order statistics of a given peer.

Let $BW_{pi}(n)$ be the uplink bandwidth process of the generic peer pi. We generate this process in two steps: first we generate a bandwidth process with a modified version of the Switched Batch Bernoulli Process (SBBP) (see [25,27] for a detailed description), the most general Markov modulated process in the discrete-time domain. Then we smooth the obtained process with a EWMA filter, as introduced in Section 3.1, with a EWMA control parameter α.

The SBBP process was modulated by an L-state underlying Markov chain, where the mean permanence is a geometrically-distributed random variable with mean value M_μ, equal for all the states. Once the underlying Markov chain leaves a state, it moves to one of the other states with the same probability. To this end, indicating two generic states of the underlying Markov chain as h and k, with $h \in \{1, \ldots, L\}$ and $k \in \{1, \ldots, L\}$, and M_μ as the mean time (expressed in slot) of each Markov chain state, we define the generic element from the state h to the state k as follows:

$$Q_{[h,k]} = \begin{cases} \frac{M_\mu - 1}{M_\mu} \cdot \frac{1}{L-1} & if\ h \neq k \\ \frac{1}{M_\mu} & if\ h = k \end{cases} \quad (8)$$

During the permanence of the Markov chain in the generic state h, the value of the uplink bandwidth of the peer Pi at the slot n, $BW_{pi}(n)$, is randomly chosen according to a Gaussian distribution characterized by:

1. the mean value associated to its class c_i calculated as:

$$\mu_h^{(c_i)} = [1 + 20\%(h - \lceil \frac{L}{2} \rceil)] \cdot (C - c_i + 1) \cdot \hat{W} \quad (9)$$

   with $\hat{W}$ equal to the average uplink value of the intermediate state of the class C;
2. a standard deviation calculated as the 8% of $\hat{W}$.

Let us stress that the bandwidth generation technique and the values described above are only provided for the reproducibility of the results achieved: any other bandwidth generation process can be used. Other bandwidth processes used by the authors have provided equally significant results and have led to the same conclusions discussed below.

In our case study we have used a slot duration of 1 sec, a number of states of the Markov chain $L = 5$, a mean duration of the permanence in each state of the underlying Markov chain $M_\mu = 60$ slots, $\hat{W} = 370.000$ bits/s, and an EWMA control parameter $\alpha = 0.8$.

The bandwidth generation tool and the uplink bandwidth processes used for all the simulations are available at [28].

4.2 Temporal Analysis of the Protocol Behavior

In this section we analyze the protocol behavior for an overlay network built using the algorithm described in Section 3.3, in the case study described so far. In this analysis we used a bandwidth distribution parameter $K_1 = 0.7$.

We first show in Figs. 6, 7, and 8 the behavior of three generic peers (a peer of class 1 on the left, a peer of class 3 in the middle, and a peer of class 5 on the right) taken as examples over an interval time of 600 seconds.

More specifically, Fig. 6 presents the instantaneous uplink bandwidth of the considered peers on each tree, calculated as in Equation (3) for the first tree, and Equation (4) for the second one. Then in Figs. 7 and 8 we observe how peers change level and number of children during their lifetime according to the behavior of the instantaneous uplink bandwidth on each tree.

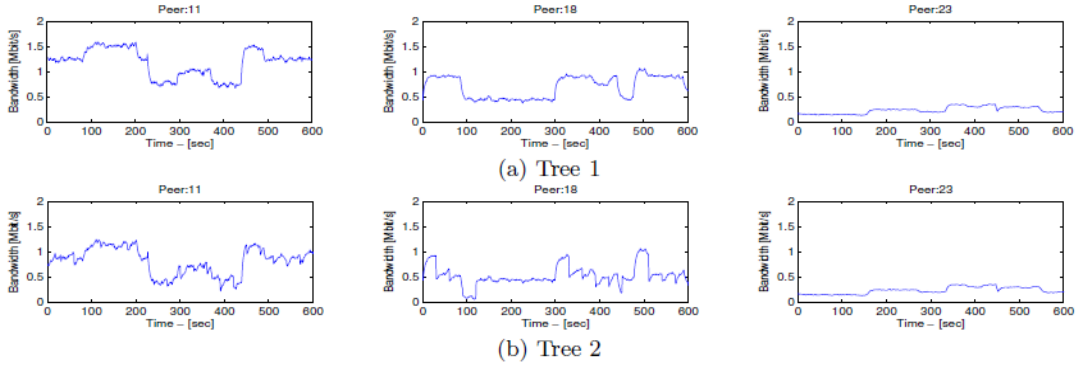


Fig. 6 Instantaneous uplink bandwidth for three generic peers in the two trees.

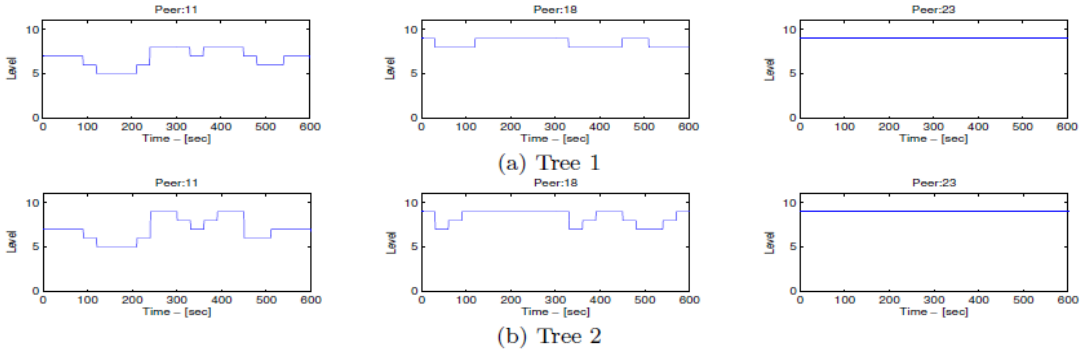
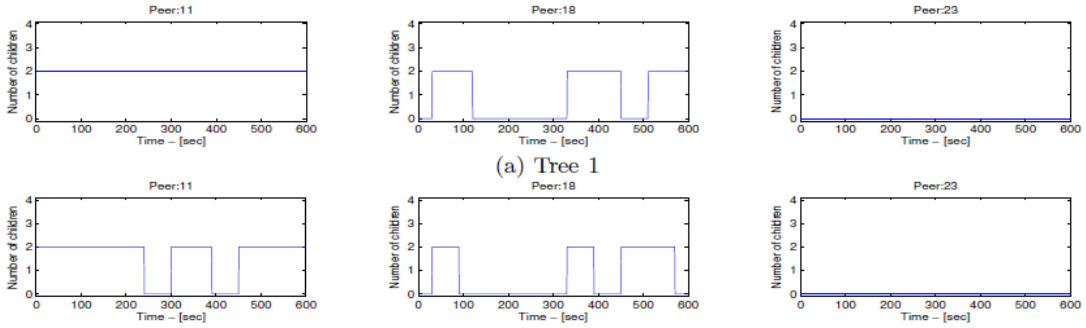


Fig. 7 Level behavior for three generic peers of the trees.

The reader notes that Figs. 6, 7 and 8 are strictly correlated. Peers with low bandwidth values get lower positions in Tree 1 and, consequently, they are more likely to be leaves; conversely, peers with higher bandwidth values lie in the upper levels in Tree1, and are more likely to be internal nodes. Similar considerations hold for Tree 2, where the available bandwidth is calculated as in Equation (4).



(a) Tree 1

(b) Tree 2

Fig. 8 Time behavior of the number of children of three generic peers of the trees.

4.3 Numerical results

Now, with the overlay network topology presented so far, we show some results in terms of PSNR and loss percentage of the transmitted sequences to the 1000 peers. As a performance measure, we evaluate the overall quality perceived at destination with the objective *Quality parameter* defined in [50], and modified here in order to take into account the frame loss probability. This parameter was introduced because simply averaging the PSNR over the non-corrupted video frames could be unfair since the smaller frames are clearly of poorer quality but also more likely to be received thanks to their reduced bandwidth requirement.

To represent the encoding quality, results are expressed in terms of *Peak Signal to Noise Ratio* (PSNR) of the Y component as a measure of the distortion. PSNR is defined as

$$PSNR = 10 \cdot log_{10} \left(\frac{MAX_I^2}{MSE} \right) \tag{10}$$

where MAX_I is the maximum possible pixel value of the original image (255 in luminance Y frames), and *MSE* stands for *Mean Squared Error*, defined as

$$MSE = \frac{1}{row \cdot col} \sum_{i=0}^{row-1} \sum_{j=0}^{col-1} \|I(i,j) - J(i,j)\|^2 \tag{11}$$

where I and J are two row × col monochrome images; I is the original image whereas J is the noisy encoded one.

As far as the number of frame losses are concerned, we will consider a frame loss when its corresponding base layer is not fully received in both the trees.

The Quality parameter we have modified models the overall video quality at destination as a function of both the mean PSNR in dB of the received frames and the frame rate. More specifically, the Quality parameter is defined as follows:

$$Q = 0.45 \cdot psnr + ((1\text{-}p)fr - 5)/10 - 7.9 \tag{12}$$

where *psnr* is the mean PSNR computed for the frames arrived at destination, *fr* is the video encoding frame rate, *p* is the frame loss percentage (that is, the number of lost frames divided by the total number of transmitted ones).

Every second, each peer reports to the rate controller of the video source the information about the available bit rate for each tree; then the Bandwidth Statistic Manager sends to the encoder the estimated minimum bandwidth (filtered using $EWMA_{ENC}$) every T_P seconds: such

an information is used to drive the coder to adapt the bit rate of the coded video to the link conditions of the worst tree. This means that the coder will adapt the bit rate according to the bandwidth variations.

As already described in Section 3.1, the 90% of the minimum bit rate available on each tree is used as the target bit rate of the base layer of each description, whereas the 90% of bit rate available at the 40% of nodes with the highest bandwidth at each second is used as the bit rate of the whole description (base + enhancement layers).

In order to highlight the performance of the multiple description algorithm, we first report the performance of the coder when receiving different combinations of the descriptions (i.e. no received at all, received only the base layer or received both the layers). Depending on what is actually received, for each frame of the simulation the following cases may occur:

1. only the base layer of the description on the Tree 1 is received;
2. only the base layer of the description on the Tree 2 is received;
3. the base layer of both descriptions is received;
4. the whole description on the Tree 1 is received;
5. the whole description on the Tree 2 is received;
6. the whole description on the Tree 1 and the base layer of the Tree 2 are received;
7. the base layer of the description on the Tree 1 and the whole other description of the Tree 2 are received;
8. both descriptions are received.

The impact of the above eight cases is depicted in Fig. 9 and have been averaged over all peers for values of K_1 ranging between 0.1 and 1. The vertical lines at the top of each column indicate the minimum and the maximum PSNR values measured for each case.
We have not considered the case $K_1 = 0$ because that would give all the available bandwidth to just the first tree, thus making useless the other tree. Clearly, as we increase the amount of information received, the performance at destination increases. The best performance is achieved in the case 8 (last bar of each graphic), when both the descriptions are fully received. In particular, classifying the eight cases shown above according to the number of received sub-descriptions (the whole description is formed by two sub-descriptions - base and enhancement layers), we obtain:

– cases 1 and 2 regard the reception of one sub-description only;
– cases 3, 4 and 5 regard the reception of two sub-descriptions;
– cases 6 and 7 regard the reception of three sub-descriptions;

We can derive the following remarks: the best performance seems to be achieved with $K_1 =$ 0:5 and $K_1 =$ 0:7, as we obtain a more uniform PSNR value. More specifically, the values of columns 1 and 2 for $K_1 =$ f0:5; 0:7g are closer each other with respect to other values of K_1. Same considerations hold for columns 3, 4 and 5, and columns 6 and 7. This uniformity happens because the bit rate information relative to the Tree 1 and 2 received by the Rate Controller, and used to drive the encoder, are similar to each other. Consequently, the video encoder adapts itself to the available bandwidth sending streams encoded with similar quality levels on the two trees.

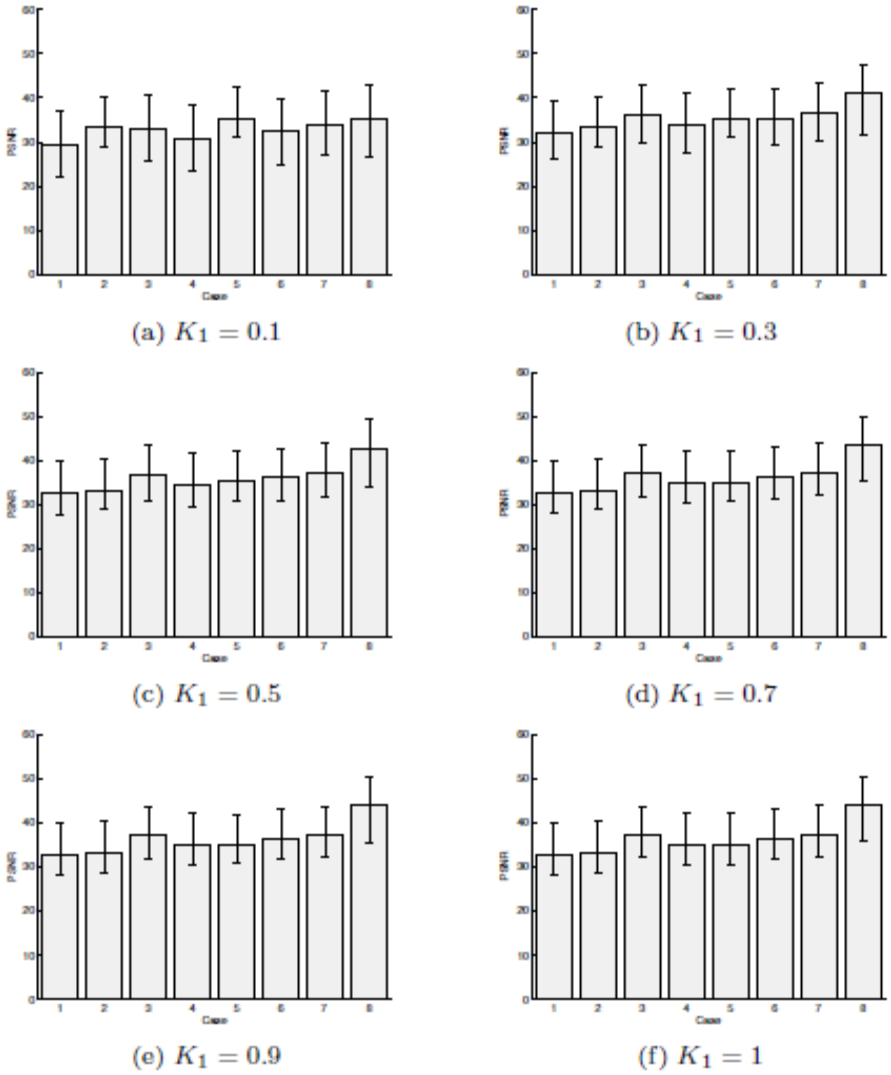
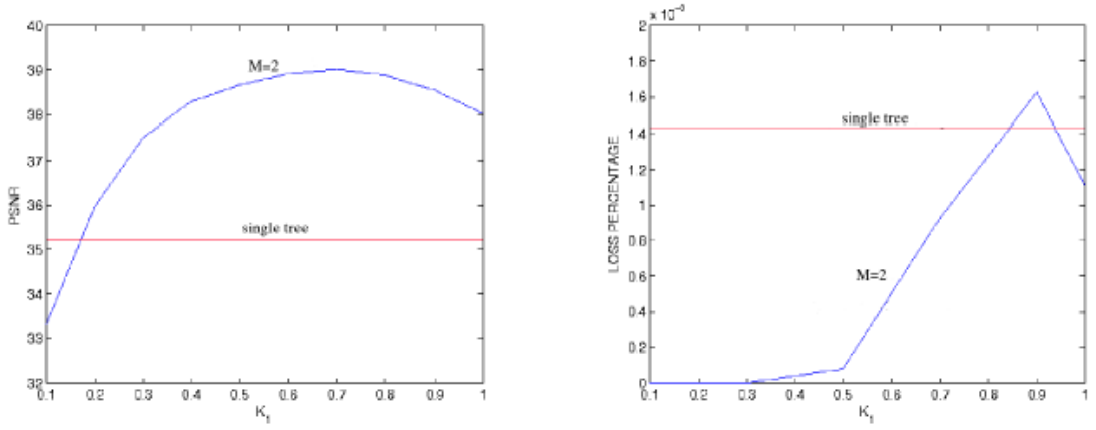


Fig. 9 PSNR results for the eight cases above for different values of K_1.

Conversely, for lower values of K_1, quality is higher for the Tree 2 which has higher bandwidth. Moreover, when using high values of K_1, if sudden bandwidth oscillations reach the minimum available bandwidth in the Tree 1 or Tree 2, peers may lose data; this usually produces worse performance. An observation has to be done for columns 3 and 8 where the used interpolation is different than the one used in columns 4 and 5. The vertical line for column 3 is sometime higher than the one in columns 4 or 5: in such a case the interpolator worked better. Interpolator is not used at all for column 8 because both the descriptions are successfully received in this case.

Fig. 10(a) shows the overall PSNR performance: the PSNR value is first averaged for each peer during its lifetime, and then averaged over all the peers. The constant line indicates the PSNR value in the case of a single tree. The best performances are obtained using K_1 = 0:7. Moreover, the reader notices that using M = 2 trees improves on the case of the single tree. Fig. 10(b) shows the overall loss percentage. No losses occur for K_1 = {0.1, 0.3}. The reader notices that

losses in the case of single tree are higher. This indicates that in those situation the allocation of bandwidth between the two different trees is made in a way that the available bandwidth is always greater than the bitrate of the stream.



(a) Average PSNR [dB] (b) Loss percentage

Fig. 10 Average PSNR and loss percentage for different values of K_1.

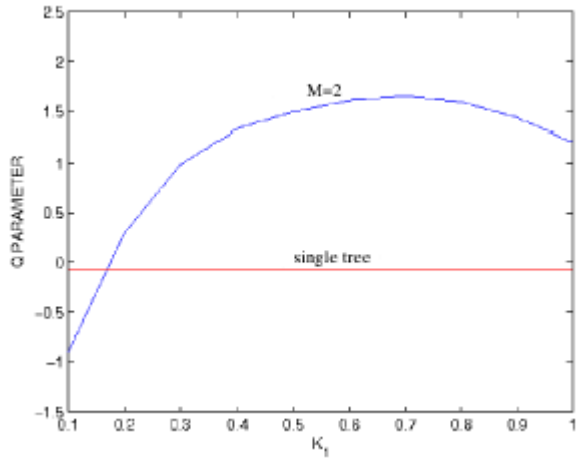


Fig. 11 Average Q values for different values of K_1.

In order to analyze together the above performance, Fig. 11 shows the overall Q parameter defined as in (12): also in this case, the Q value is first averaged for each peer during its lifetime, and then averaged over all the peers. As for the PSNR and the loss percentage, the Q value is worst in the case of the single tree. Moreover let us note that the curve representing Q is more smoothed than the PSNR for $K_1 \geq 0{:}5$. This happens because the Q computation is affected by the number of losses, and these last ones occur for $K_1 \geq 0{:}3$ (see Fig. 10 (b)). As for the PSNR, the best performances for the Q parameter are obtained using $K_1 = 0{:}7$.

Finally, in order to evaluate peer fairness, we evaluated the Jain Fairness Index [51] for the PSNR in Fig. 12; the case of $M = 2$ is compared to the case of the single tree. Again, the reader may notice that results are better when using the proposed approach also in terms of PSNR peer fairness. It follows that each peer is equally served and this does not depend on the single bandwidth characteristics.

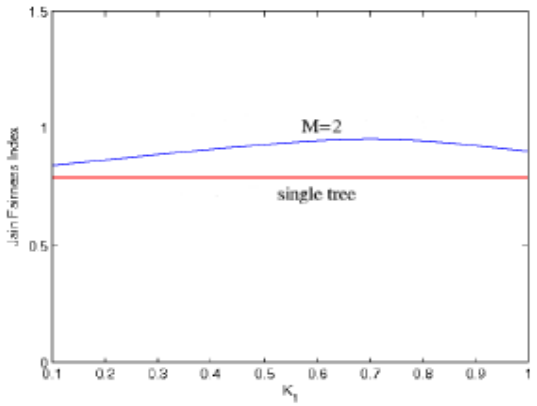


Fig. 12 Jain Fairness Index for different values of K_1.

5. CONCLUSIONS

In this paper we have introduced a video streaming platform over p2p networks that exploits a careful organization of the multicast distribution tree and controls the parameters of the video coder feeding back the actual bandwidths made available for the service by the peers. Reliability and flexibility are ensured by employing a video server equipped with a coder capable to generate scalable multiple descriptions streams which are routed through different trees. Comparisons between the proposed approach to build an overlay network and a single tree structure show that the former improves over the latter in terms of PSNR and frame loss. Moreover, using the Jain Fairness Index, we have shown that the system is fair: peers receives the video stream independently of their bandwidth characteristics. A detailed analysis allowed us to determine the best value for the K_1 parameter. This value can be thought as the best value able to balance the bandwidth between the M trees. In particular, the target of this paper has been to:

– define a real-time multimedia multipoint transmission platform, whose QoS constraints are stringent in terms of end-to-end delays; for such a purpose, we have adopted multiple trees topologies;
– define the architecture of the system;
– define a strategy for bandwidth distribution among the M trees which is applied in a distributed fashion by the MDC encoder and by each peer;
– define a strategy of topology management which takes into account the uplink bandwidth of each peer, in particular, the bandwidth that each peer is able to provide and use in each tree.
– tackling bandwidth oscillations of the links of the underlying networks by applying scalable multiple description coding, hierarchical video streaming and topology management.

Further work is foreseen to improve granularity of the adaptation algorithm and the possibility to generate unbalanced substreams even when the number M of trees is greater than two. Moreover, peer departures and arrivals, which affect system connectivity, represent another problem to be taken into account in our future research. Last but not least, we want to study the effects of the introduction of a fine grain scalablity (FGS) to encode the base and enhancement layers of the descriptions.

Acknowledgement

Authors would like to thank the anonymous reviewers for their insightful ideas and suggestions.

References

1. Y. Chu, S. Rao, S. Seshan and H. Zhang, "Enabling conferencing applications on the internet using an overlay muilticast architecture", vol. 31, n. 4, pp. 55-67.
2. M. Castro, P. Druschel, A-M. Kermarrec, A. Nandi, A. Rowstron and A. Singh, "Split- Stream: High-bandwidth content distribution in a cooperative environment", IPTPS'03, Berkeley, CA, February, 2003.
3. P. Baccichet, J. Noh, E. Setton and B. Girod, "Content-Aware P2P Video Streaming with Low Latency", Proc. IEEE International Conference on Multimedia and Expo (ICME 2007), Beijing, China, July 2007, 2007.
4. S. Mao, S. Lin, Y. Wang, S. Panwar, Y. Li, "Multipath video transport over ad hoc networks", IEEE Wireless Comm. vol. 12 ,no. 4, aug. 2005, pp. 42-49.
5. E. Setton, Y. Liang, B. Girod, "Adaptive multiple description video streaming over multiple channels with active probing", Proceedings of IEEE ICME, Baltimore, MD, July 2003, pp. 509-512.
6. E. Ishikawa and C. L. de Amorim, "Cooperative Video Caching for Interactive and Scalable VoD Systems", Proceedings of the First International Conference on Networking-Part 2, vol. 2094, pp. 776-785, 2001.
7. M. Handley, J. Crowcroft and C. Bormann, "Multimedia conferencing on the internet", 1996.
8. O. Jorg and D. Jegadish "Peer-to-peer Media Streaming", www.netlab.hut.fi/opetus/s383152/2008/slides/5-p2p-streaming.pdf.
9. L. Zhenjiang, Y. Yu, X. Hei, D. H. K. Tsang, "Towards low-redundancy push-pull P2P live streaming", in Proceedings of 5th ICST International Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness (QShine 2008), 2008.
10. M. Zhang, Q. Zhang, L. Sun and S. Yang, "Understanding the power of pull-based streaming protocol: Can we do better?", Selected Areas in Communications, IEEE Journal, vol. 25, n. 9, pp. 1678-1694, 2007
11. X. Hei, Y. Liu, K.W. Ross, "IPTV over P2P Streaming Networks: the Mesh-pull Approach", Communications Magazine, IEEE, vol. 46, n. 2, pp. 86-92, 2008.
12. O. Heckmann, N. Liebau, V. Darlagiannis, A. Bock, A. Mauthe and R. Steinmetz, "A peer-to-peer content distribution network", From Integrated Publication and Information Systems to Information and Knowledge Environments, pp. 69-78, 2005.
13. M. Berka, "Optimal Level of Hierarchy for Multicast in IP Networks", IJCSNS International Journal of Computer Science and Network Security, vol. 6, n. 8B, pp. 31-34, 2006.
14. S. E. Deering and D. R. Cheriton, "Multicast routing in datagram internet-works and extended LANs", ACM Transactions on Computer Systems (TOCS), vol. 8, n. 2, pp. 85-110, 1990.
15. K. Dasqupta, R. Singh, B. Viswanathan, D. Chakraborty, S. Mukherjea, A. A. Nanavati and A. Joshi, "Social ties and their relevance to churn in mobile telecom networks", in Proc. of the 11th international conference on Extending database technology, pp. 668-677, 2008.
16. B. P. Godfrey, S. Shenker and I. Stoica, "Minimizing churn in distributed systems", in Proc. of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications, pp. 147-158, 2006.
17. F. Atalla, D. Miranda, J. Almeida, M. A. Goncalves and V. Almeida, "Analyzing the impact of churn and malicious behavior on the quality of peer-to-peer web search", in Proc. of the 2008 ACM symposium on Applied computing, pp. 1137-1144, 2008.
18. D. Stutzbach and R. Rejaie, "Understanding churn in peer-to-peer networks", in Proc. of the 6th ACM SIGCOMM conference on Internet measurement, pp. 189-202, 2006.
19. B. Mitra, S. Ghose, N. Canculy and F. Peruani, "Stability analysis of peer-to-peer networks against churn", in Pramana-Journal of Physics, vol. 71, n. 2, pp. 263-273, 2008.

20. X. Hei, C. Liang, J. Liang, Y. Liu, and K. W. Ross, "A measurement study of a largescale P2P IPTV system", IEEE Transactions on Multimedia, vol. 9, n. 8, pp. 1672-1687, Dec. 2007.
21. J. Liu, S. G. Rao, B. Li, and H. Zhang, "Opportunities and challenges of peer-to-peer Internet video broadcast,"Proc. of the IEEE, vol. 96, n. 1, pp. 11-24, Jan. 2008.
22. X. Zhang, J. Liu, B. Li, T.-S.P. Yum, "DONet/CoolStreaming: A data-driven overlay network for peer-to-peer live media streaming"Proc. of INFOCOM 2005, Miami, FL, USA, Mar. 2005.
23. PPLive, http://www.pplive.com.
24. Y. Chu, A Ganjam, T.S.E. Ng, S.G. Rao, K. Sripanidkulchai, J. Zhan, H. Zhang, "Early Experience with an Internet Broadcast System Based on Overlay Multicast"Proc. of USENIX Annual Technical Conference, Boston MA, July 2004.
25. A. Lombardo and G. Schembra, "Performance evaluation of an Adaptive-Rate MPEG encoder matching IntServ Traffic Constraints"IEEE Transactions on Networking, vol. 11, no 1, pp. 47-65, Feb 2003.
26. A. Lombardo and D. Reforgiato and G. Schembra, "P2P and MPEG FGS Encoding: A Good Recipe for Multipoint Video Transmission on the Internet", International Journal of Digital Multimedia Broadcasting, vol. 2009, Article ID 453471, 21 pages, 2009. Doi 10.1155/2009/453471
27. O. Hashida, Y. Takahashi and S. Shimogawa, "Switched Batch Bernoulli Process (SBBP) and the Discrete-Time SBBP/G/1 Queue with Application to Statistical Multiplexer Performance" IEEE Journal on Selected Areas in Communicatins, ISSN 0733-8716, 1991.
28. "Bandwidth Generator Tool and Tree Overlay Construction Algorithm", Available at http://www.diit.unict.it/arti/Tools/BandwidthSim.zip.
29. B. Krithikaivasan et al., "ARCH-based Traffic Forecasting and Dynamic Bandwidth Provisioning for Periodically Measured Nonstationary Traffic"IEEE/ACM Transaction on Networking, June, 2007.
30. P.J. Brockwell et al., "Introduction to time series and forecasting" Springer, Berlin. 1996.
31. V.K. Goyal, "Multiple Description Coding: Compression meets the network," Signal Processing Magazine, vol. 18, no. 5, pp. 74 - 93, Sept. 2001.
32. Y. Wang, R. Reibman and S. N. Lin, "Multiple description coding for video delivery," Proc. IEEE, vol. 93, no. 1, pp. 57-70, Jan 2005.
33. V.N. Padmanabhan, H.J. Wang, P.A. Chou, K. Sripanidkulchai, “Distributing Streaming Media Content Using Cooperative Networking,” In Proc. NOSSDAV, May 2002.
34. Y. Wang and S. Lin, "Error resilient video coding using multiple description motion compensation," IEEE Transaction Circuits and Systems for Video Technology, vol. 12, no. 6, pp. 438 - 452, June 2002.
35. M. Caramma, M. Fumagalli, and R. Lancini, "Polyphase down-sampling multiple description coding for IP transmission," SPIE 2001 Visual Communications and Image Processing, San Jose, CA, pp. 545-552, 2001.
36. Zhe Wei, Canhui Cai, and Kai-Kuang Ma , "A Novel H.264-based Multiple Description Video Coding Via Polyphase Transform and Partial Prediction," International Symposium on Intelligent Signal Processing and Communications, Yonago, Japan, pp. 151 - 154, December 2006.
37. M. Folli, L. Favalli, and M. Lanati "Parameter optimization for a scalable multiple description coding scheme based on spatial subsampling," Mobimedia ’08, Oulu, Finland, July 2008.
38. L.Favalli, M. Folli, "ILPS: a scalable multiple description coding scheme for H.264," Second International Workshop on Future Multimedia Networking, Coimbra, Portugal, June 2009.
39. M. Liu, Ce Zhu, "Multiple description video coding using hierarchical B pictures," IEEE International Conference on Multimedia and Expo, Beijing, China, pp. 1367 - 1370, July 2007.
40. N. Franchi, M. Fumagalli, R. Lancini, and S. Tubaro, "Multiple Description Video Coding for Scalable and Robust Transmission Over IP," IEEE Transactions on Circuits and Systems for Video Technology, vol. 15, no. 3, pp. 321 - 334, March 2005.
41. M. Van Der Schaar, D. S. Turaga, "Multiple Description Scalable Coding Using Wavelet-Based Motion Compensated Temporal Filtering," International Conference on Image Processing, vol. 3, pp. 489 - 492, September 2003.

42. Huihui Bai and Yao Zhao, "Multiple Description Video Coding Based on Lattice Vector Quantization," First International Conference on Innovative Computing, Information and Control, vol. 2, pp. 241 - 244, August 2006.
43. Mei Yu, Zhang Wenqin, Gangyi Jiang, Zhu Yin, "An Approach to 3D scalable multiple description video coding with content delivery networks," IEEE International Workshop on VLSI Design and Video Technology, pp. 191 - 194, May 2005.