Hindawi Publishing Corporation
International Journal of Digital Multimedia Broadcasting
Volume 2009, Article ID 453471, 21 pages
doi:10.1155/2009/453471

Research Article

P2P and MPEG FGS Encoding: A Good Recipe for Multipoint Video Transmission on the Internet

Alfio Lombardo, Diego Reforgiato, and Giovanni Schembra

Dipartimento di Ingegneria Informatica e delle Telecomunicazioni, University of Catania, V.le A. Doria 6, 95125 Catania, Italy

Correspondence should be addressed to Diego Reforgiato, diegoref@diit.unict.it

Received 6 March 2009; Accepted 4 June 2009

Recommended by Robert Briskman

In the last years Peer-to-Peer (P2P) systems have gained ground for content sharing between communities, determining a real revolution on the Internet. The characteristics of P2P systems make them a very good choice for multimedia content distribution over IP networks. However, although P2P technology gives new opportunities to define an efficient multimedia streaming application, at the same time it involves a set of technical challenges and issues due to the best-effort service offered by the Internet and its dynamic and heterogeneous nature. The most of existent protocols for video communications over P2P mainly focus on tree topology maintenance, without paying any attention to the encoding problem. The idea of this paper is to propose a multipoint video broadcast framework over a heterogeneous content distribution P2P network. In the proposed system the source generates the video flow by using an MPEG-4/FGS encoder, in such a way that no losses occur at the Baselayer stream even in the presence of short-term bandwidth fluctuations. Although in the past the FGS was not employed due to its encoding complexity, today, thanks to advances in hardware technology, we were able to develop an MPEG-4/FGS encoder on low-cost PCs which turned out to be more feasible and appealing for its flexibility. The FGS layer is sent together with the Base layer, but with a lower priority. The source uses a rate controller to regulate the encoding rate of the Base layer. To this aim, a protocol is defined in order to provide the source with information related to the most stringent bottleneck link on the overlay network. A technique to reorganize the content distribution tree is proposed and discussed. To evaluate the performance of the proposed framework a case study is introduced; improvements obtained with respect to several reference cases where FGS is not applied are also shown.

1. Introduction

In recent years, with the widespread deployment of broadband access, broadcast video transmission over the Internet is becoming increasingly popular [1]. The main problem of such a kind of systems is to realize multipoint communication on the current Internet that does not natively support it. In the first generation of such systems the most widely used approach has been the employment of multiple parallel unicast streaming. According to this approach, multimedia data are transmitted by the source to each receiver in a point-to-point fashion: whenever a new client accesses the service, a dedicated stream is allocated until the end of it connection. However, since this approach requires a separate streaming bandwidth for each client, the multimedia source and the consumed network bandwidth resources inevitably grow linearly with the user population. Therefore this approach is not scalable with the number of users, and thus unfeasible in video broadcast scenarios, where the number of users is very high.

The first solution to the scalability problem was the application of IP network multicasting. In this way, unlike multiple parallel unicast transmission, a multicast multimedia stream can be shared by more than one receiver. The network switches/routers automatically replicate the multicast video for multiple receivers without adding any extra streaming workload on the multimedia server. However, IP multicast is not widely deployed, mainly due to practical and political issues. For example, multicast is not available in many low-cost Internet access points, like domestic ADSL.

In the meanwhile, Peer-to-Peer (P2P) systems have gained ground for content sharing between communities, determining a real revolution on the Internet. Unlike traditional distributed systems, P2P networks are self-organizing

networks that aggregate large numbers of heterogeneous computers called nodes or peers (nodes and peers will be used interchangeably in this paper). In P2P systems, peers can communicate directly with each other for data sharing and data exchanging. Peers also share their communication and storage resources.

The characteristics of P2P systems make them a very good choice for video broadcast over IP networks [2]. According to the P2P approach, peers interested in the same video transmission organize themselves into an application-layer multicast tree. A peer in the application layer multicast tree receives video packets from it parent, then duplicates and forwards them to it children [3, 4]. Peers in the overlay network are cooperative in the sense that they share data and exchange group control information. A multitude of protocols have been proposed in previous literature to manage tree-structured overlay networks for video broadcast [5–8].

However, although P2P technology gives new opportunities to define an efficient multimedia streaming application, at the same time, it involves a set of technical challenges and issues due to the best-effort and dynamic nature of the service offered by the Internet, and the heterogeneity of terminals used by clients to access the service. First of all, although peers access the network through different access links with totally different bandwidth characteristics, often dominated by the uplink connections, P2P communication infrastructure is often built upon an overlay network whose topology does not depend on the underlying physical network. seconds, network bandwidth, delay and loss behaviors rapidly change in time. Last, but not least, a large number of different terminals are becoming very popular, ranging from powerful high-performance home receivers to mobile handheld video devices; they present different capabilities and requirements in receiving and decoding video content.

The work in [9] describes state-of-the-art strategies that allow the deployment of efficient streaming solutions in P2P systems, focusing on both delivery architectures and adaptive streaming mechanisms. These solutions enable resource-demanding and delay-constrained applications over unstructured networks, and demonstrate the powerful of the P2P paradigm, along with adaptive streaming mechanisms, providing an interesting alternative for low-cost and effective multimedia communication applications.

In this context many other protocols based over P2P have been defined in the last few years. Unfortunately, the most of them are mainly related to the maintenance of the tree topology (see [10] for further details); they are not focused to the encoding aspects which, especially in heterogeneous and dynamic scenarios like the one considered in this paper, are fundamental issues that have to be accounted very carefully. In fact, a common hypothesis for all of them is that the source should encode video flows with the knowledge of the instantaneous network bandwidth of all peer-to-peer connections over the tree, and modify it output rate according to bandwidth variations. Otherwise, if no instantaneous knowledge of the link bandwidths is available at the source, encoding should be performed with an average quality, providing

(i) "good" peers (i.e., peers with a high-speed Internet access and a high-performance receiver device) with a bandwidth waste and an inadequately low-quality level,

(ii) "bad" peers (e.g., peers accessing the service with handheld devices, or accessing the Internet with low-rate links) with unpredictable and large number of losses.

However, even when the source knows the instantaneous bandwidth of each link in the distribution tree, the decision of the encoding bit rate is very hard. In fact, the solution that avoids losses would be to follow the worst peers, but in such a case the overall quality of the broadcasted video should be dominated by them: one peer with a very poor bandwidth is able to strongly degrade the quality of the video received by all the other peers.

Some rare cases of works accounting video encoding in P2P multipoint video distribution are present in literature, but they are often limited to nonrealtime video on demand (VOD) applications. For example, [11] presents a P2P multicast protocol and analyzes the gains that video coding and prioritized packet scheduling at the application layer can bring to the overall streaming performance. A rate-distortion model which predicts end-to-end video quality was presented in throughput-limited environments, using a source that applies H.264 encoding with SP and SI frames in order to adaptively stop the error propagation due to packet loss. However, in that paper, retransmission requests issued by receiving hosts are used to recover the most important missing packets while limiting the induced congestion, and therefore cannot be applied in video broadcast scenarios, where retransmissions may be the cause of unacceptable delay jitter.

Keeping all this in mind, the idea of this paper is to apply fine granularity scalability (FGS) encoding [12–17] for video transmission. FGS is an evolution of the scalable hierarchical video encoding; it was defined some years ago to deliver multimedia applications in heterogeneous network environments with different bandwidth and loss behaviors.

An FGS stream has only two layers: a Base layer that must be received to make possible video decoding, and an enhancement layer, henceforward indicated as the FGS layer, which can be delivered optionally where bandwidth is available. FGS allows the source to adjust the relative sizes of both Base and FGS layers, therefore allowing the FGS layer to be broken up and allowing the decoder to decode any portion of the received FGS layer. The source or any intermediate node is responsible to do that.

The authors of this paper think that, although not widely applied in the past due to it encoding complexity, today, thanks to the evolution in hardware and software technologies, the FGS appears a good and feasible solution for multipoint multimedia broadcast systems for the immediate future. Besides, [18] is an invention of 2006 that discloses methods, devices and systems for effective fine granularity scalability coding and decoding of video data.

In fact, the MPEG-4 FGS encoder developed by the authors is able to encode and decode video in realtime. The encoder has been implemented in Visual C++ using the

Intel Integrated Performance Primitives (Intel IPP) [19], an extensive library of multicore-ready, highly optimized software functions for multimedia data processing, and communications applications. Running on a start-level dual-core personal computer equipped with 2 GB of RAM, it is able to encode about 100 fps and decode about 400 fps for CIF video streams.

The target of this paper is to propose a multipoint broadcast video transmission framework over a heterogeneous content distribution P2P network. More specifically, the purpose of this work is to make the following contributions:

(i) launching and encouraging the application of a rate-controlled FGS encoding in broadcast video transmission in P2P networks; to the best of our knowledge, this is the first work with such an idea;

(ii) defining the architecture and a network managing protocol for the proposed broadcasting P2P platform;

(iii) analyzing and discussing an algorithm to manage the P2P network tree topology in a case study, giving the readers the possibility to change and tune the algorithm parameters according to the domain of interest: people who want to use the proposed architecture can freely decide to define any other tree topology management protocol and apply that to the Topology Manager entity described along the paper.

In the proposed platform the source generates the Base layer of the MPEG-4 stream in such a way that no losses occur at the Base-layer stream even in the presence of short-term bandwidth fluctuations. The FGS layer is sent together with the Base layer, but with a lower priority. In this way, peers connected to other peers through bottleneck links guarantee an efficient transmission of the Base layer, discarding only portions of the FGS layer.

The source uses a rate controller to regulate the encoding rate of the Base layer. A protocol is defined in order to provide the source with the necessary information related to the bandwidth of the most stringent bottleneck link. The framework works on a tree-structured P2P network, and any tree construction and management protocols [5–8] can be adopted.

A case study is introduced to evaluate both the performance of the proposed framework and the improvements obtained with respect to a reference case in which FGS is not applied.

The paper is structured as follows. Section 2 provides a brief description of FGS. Section 3 describes the proposed platform and the definition of the protocol. Section 4 analyzes a case study; first, the simulator tool we have implemented to generate the bandwidth processes at the overlay network level is described. Then, a statistical analysis of the performance at both the overlay network and the application levels will be carried out. Specifically, performance is calculated analyzing video at the receiving side, accounting frame loss and encoding PSRN simultaneously. Performance comparison with other platforms is made in order to evaluate the improvements introduced by that proposed in this paper. Finally, Section 5 concludes the paper.

2. FGS: Overview and Some Statistics

In this section we will provide a brief overview of the main characteristics of the FGS encoding technique in order to facilitate the understanding of the remainder sections of the paper. For a more detailed description of FGS, the reader is referred to [12–14].

FGS is a scalable encoding technique. Generally, there are three types of scalability, that is, temporal scalability, spatial scalability and SNR scalability. In all the three cases, the Base-layer pictures are encoded based on subsampling with either less frame rate (for temporal scalability), smaller picture size (for spatial scalability), or coarser picture quality (for SNR scalability). Full-quality video is obtained by the combination of both Base and FGS layers.

FGS is an evolution of the scalable hierarchical video encoding. It emerged to deliver multimedia applications in heterogeneous network environments with different bandwidth and loss behavior. It was defined in [13] with the main target of achieving a good balance between coding efficiency and scalability.

Its encoding is designed to cover any desired bandwidth range while maintaining a very simple scalability structure. The basic idea of FGS is to encode a video sequence into two layers only, a Base layer and an enhancement layer, in the following called the FGS layer. A MPEG-4 encoding is used: the Base layer is obtained with a classical MPEG-4 encoder using nonscalable coding, whereas the FGS layer is coded using a fine-granular scheme. The latter encodes the difference between the original picture and the reconstructed one with the use of bit-plane coding of the DCT coefficients.

An important application of FGS regards multihop connections with heterogeneous bandwidths; in such a case, each node may truncate where desired. The encoder needs to know just the minimum bandwidth over which it has to code the content. The bit stream of the FGS layer may be truncated by intermediate nodes into any number of bits per picture. The decoder will be able to reconstruct an enhancement video by combining the Base layer and the truncated FGS layer received bit streams. The FGS-layer video quality is proportional to the number of bits decoded by the decoder for each picture.

As far as the source is concerned, the encoding is feasible both offline and online, thanks to the availability of new hardware and software routines.

The idea of this paper is to apply MPEG-4 FGS encoding for video transmission, which allows intermediate nodes to truncate the bit stream of each frame at any point, thus only degrading the quality proportionally to the current available bandwidth. The Base layer of the MPEG-4 stream is generated in such a way that it encoding bandwidth is a little lower than the current minimum bandwidth in the tree. Therefore losses are negligible at the Base-layer stream even in the presence of short-term bandwidth fluctuations (specifically, in our system the Base layer is generated frame-by-frame at 90% of the minimum link bandwidth in the

whole network). The FGS layer is sent together with the Base layer, but with a lower priority. By so doing, peers connected to other peers through bottleneck links guarantee the transmission of the Base layer, discarding only portions of the FGS layer.

This scheme is sketched in Figure 1, where an example of frame transmission on the tree is depicted. The amount of frame bits transmitted along the tree is represented by rectangles: the light gray portion of the frame represents the Base layer part, which is the same over all the links from the source to all the peers in the network. The FGS-layer part of the frame is represented with dark gray portions, with a length proportional to the number of bits forwarded to the next peer in the tree within a frame interval. Each link shows the transmission bandwidth between two connected nodes of the tree; specifically, the indicated bandwidth is the uplink bandwidth of each node towards it children. In this figure we can observe that higher-level peers receive a larger amount of bits, and therefore a better quality; quality degrades along the tree at each bottleneck uplink. In the example, the Base layer is encoded at 0.5 Mbit/s in such a way that even $P_{2,7}$, that is, the node with the worst connection with it parent in the tree, is able to receive it with no losses; the quality perceived by $P_{1,2}$ is greater than that perceived by $P_{2,6}$ because of the portion of FGS-layer discarded by $P_{1,2}$ due to the bandwidth value of the link $\{P_{1,2} \rightarrow P_{2,6}\}$, which is lower than that of the link $\{S \rightarrow P_{1,2}\}$. Of course, the performance perceived by each peer and, as a consequence, the overall performance of the proposed system are strongly influenced by the position of each peer in the tree, and by the tree topology itself. For this reason, the tree organization strategy plays a fundamental role in the performance of the whole system, and the related issues will be deeply discussed in Section 3.3.

3. System Description

The target of this section is to describe the system we propose in this paper. It is a live video broadcast platform where a video source distributes a video stream to a number of clients in a multipoint fashion. Multipoint communication is achieved by applying a P2P approach, configuring a tree-structured overlay network where the root is the video source, while the other clients, henceforward referred to as peers, are internal nodes or leaves. In particular, Sections 3.1 and 3.2 present, respectively, the architecture of the video source and the generic peer; Section 3.3 describes the algorithm we have used to organize the overlay tree.

3.1. Architecture of the Video Source. The architecture of the video source is shown in Figure 2. As it can be seen, it core is the *MPEG-4/FGS Video Encoder*, which receives a raw video stream at it input and produces the MPEG-4 video flow, made up of two separate streams, one related to the Base layer and one to the FGS layer. A *Replicator* is needed in order to create as many streams (for both Base and FGS layers) as the number of the source's children; this number is upper bounded by the so-called fan-out parameter F, defined as the maximum upload connections each peer can support at the overlay network layer. The bits produced at each frame interval are queued in the *Base-layer Buffer* and *FGS-layer Buffer*, respectively, in order to avoid possible losses. Then they are first grouped in packets by a *Packetizer* and, next, sent to the *Intra-Flow Scheduler*, which applies a round-robin strict-priority scheduling algorithm that considers the data in the second buffer (FGS layer stream) only in case the first one (Base layer Buffer) is empty. More specifically, the buffer gives priority to I-frames since they are the most important to be received as discussed in Section 4.4.2. At the end of each GoP, if the buffers contain data (Base and FGS layer) they are emptied in order to avoid excessive buffering delays. At most $2F$ streams coming from these buffers end up into the *InterFlow Scheduler*, which applies a weighted round robin algorithm to send the streams with an amount of bandwidth proportional to the uplink bandwidth estimated towards each children of the source.

The amount of bits to be used in the Base layer, and the relative encoding quality, are determined by the *Rate Controller* through the quantizer scale parameter qsp, that is chosen in the range between 1 and 31: the greater the qsp value, the poorer the encoding quality. As usual, it uses the rate distortion curves of the movie being coded [20–22].

The *Rate Controller* obtains the needed information about the bandwidth from the *Bandwidth Statistic Manager*. The latter periodically receives the source's children uplink bandwidth estimation from the *Bandwidth Estimator* and, at the same time, the uplink bandwidth estimation by all the other peers which are internal nodes in the tree (see $B_{p_1} \cdots B_{p_N}$ of Figure 2 or B_{p_i} of Figure 3).

The qsp is chosen by the Rate Controller using the minimum value between the uplink bandwidth estimations made by all the peers in the tree at each interval.

EWMA Filter. In order to avoid excessively strong oscillations of both the encoding quality and the system behavior, the bandwidth values are first smoothed with an exponentially-weighted moving average (EWMA) filter with parameter $\beta^{(\mathrm{RC})}$, defined as follows:

$$\hat{B}_n = \beta^{(\mathrm{RC})} \cdot \hat{B}_{n-1} + \left(1 - \beta^{(\mathrm{RC})}\right) \cdot B_n, \tag{1}$$

where $\hat{B}_{n-1}$ and $\hat{B}_n$ are the filtered bandwidths at the $(n-1)$th and nth update events, B_n is the instantaneous bandwidth at the nth update event, and $\beta^{(\mathrm{RC})}$ is the Rate Controller filter parameter whose range is between 0 and 1; values of $\beta^{(\mathrm{RC})}$ 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 $\beta^{(\mathrm{RC})}$ give more importance to recent measures, achieving greater responsiveness to the process variations. In our implementation we have used the Rate Controller EWMA ($\mathrm{EWMA}_{\mathrm{RC}}$) with $\beta^{(\mathrm{RC})} = 0.8$.

Another important block in the video source architecture is the *Topology Manager*, which decides and maintains the tree network topology by deciding the position of each peer within the tree. It receives the uplink bandwidths of all the

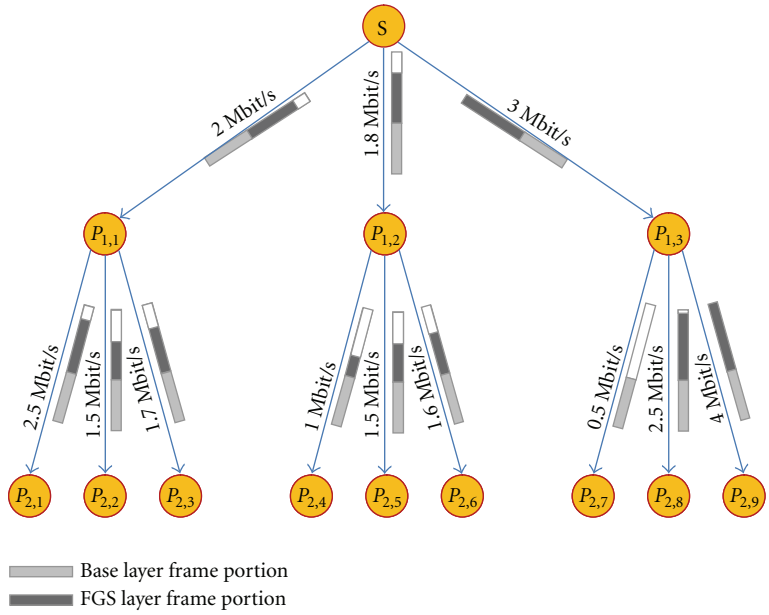


Figure 1: FGS video quality degradation along the tree due to the presence of bottlenecks.

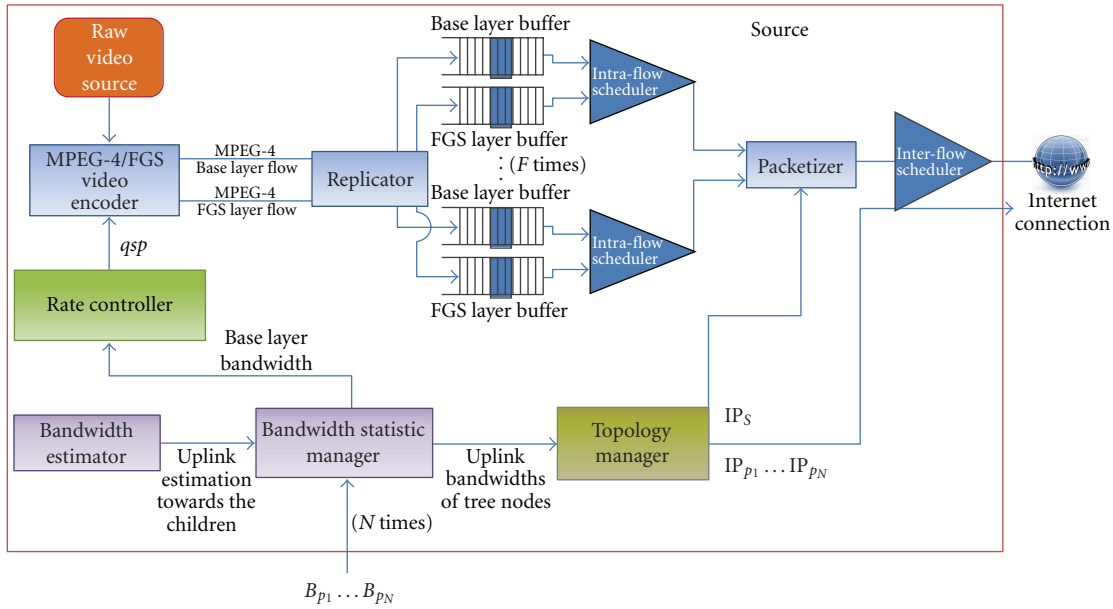


Figure 2: Architecture block diagram of the video source.

peers from the *Bandwidth Statistic Manager*, and 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 a position in the current tree and the IP address of the peer that will be it parent. The Topology Manager sends also the IP address of the new peer to it parent node. Specifically, the parent node inserts the IP of the new child in it Replicator in order to create another copy of the packets for the new child. The parent node creates also a new buffer pair (for the Base and the FGS layers) for it new child, and communicates to the Packetizer and the InterFlow Scheduler to handle the new node. The latter will be inserted in the weighted round robin buffer. Finally, the parent node inserts the IP of it new child in the Bandwidth Estimator.

When an existing peer departs, the process just described for peer arrival is reversed. The parent node deletes from the Replicator the IP of the departing peer. It also cancels the buffer pair (for the Base and the FGS layers) and communicates to the Packetizer and the InterFlow Scheduler

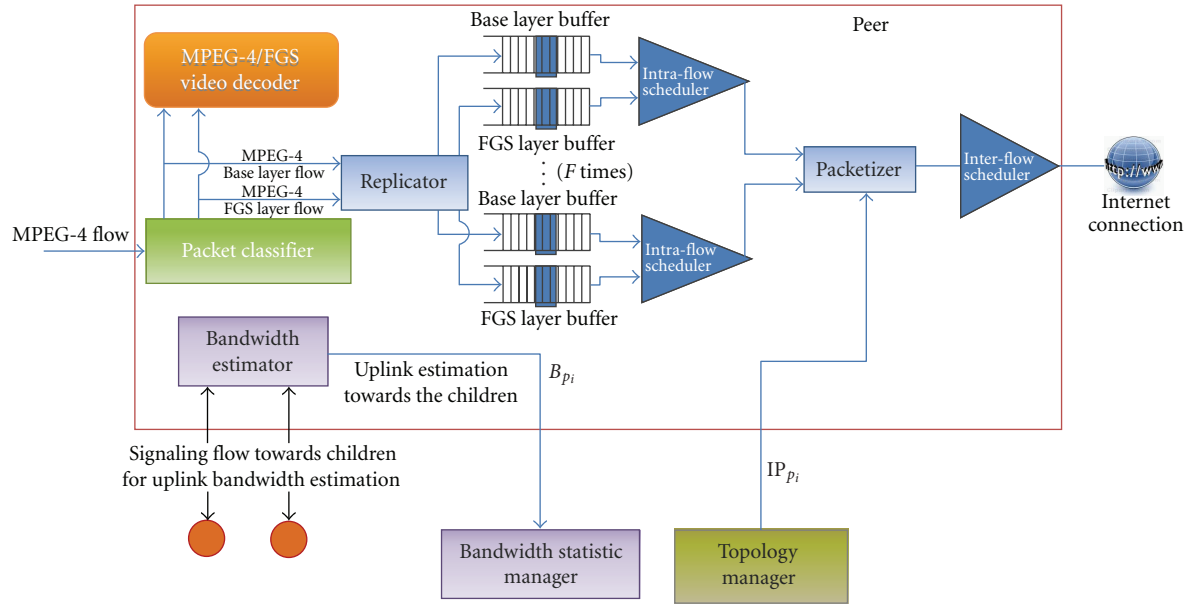


Figure 3: Architecture block diagram of the generic peer p.

to stop serving it old child node. The IP of the departing peer is also removed from the Bandwidth Estimator.

When the Topology Manager receives the information about the uplink bandwidth of each peer, periodically (in our implementation we have considered a period of $T_T = 30$ Seconds) it updates and optimizes the tree. To this end, using the same information received by the *Bandwidth Statistic Manager*, it implements an algorithm to manage topology modifications run-time after bandwidth variations. To avoid strong oscillations of peers when reorganizing them in the updated topology, the received bandwidth values are first passed under a EWMA filter, the so-called Topology Manager EWMA ($EWMA_{TM}$) filter, which works like the one defined in (1) with parameter $\beta^{(TM)}$ (in our implementation we used $\beta^{(TM)} = 0.9$).

3.2. Architecture of a Generic Peer. The architecture of a generic peer is shown in Figure 3. It mainly performs three functions:

(1) video play-out,

(2) forwarding,

(3) bandwidth estimation.

Video bit streams, organized in IP packets, are received by the *Packet Classifier*, subdivided and queued in two different buffers, according to the type of packets: Base layer and FGS layer. At the same time, packets are given to the local *MPEG-4/FGS Video Decoder* for playback. As for the case of the source, a Replicator block is needed in order to create at most F streams (for both Base and FGS layer), one for each child of the generic peer. The bits produced at each frame interval are queued in the *Base-layer Buffer* and the *FGS-layer Buffer*, respectively. The two buffers are served by an *Intra-Flow Scheduler* as those discussed previously for the source. Of course, if estimation has been made correctly, the Base layer will find enough bandwidth on the transmission channels, and will not incur in any packet loss. On the other hand, the FGS-layer bits in each frame will be enqueued during the GoP, and transmitted only at the end of it, after the transmission of the Base layer of all the frames in the same GoP. If the time available for the GoP transmission elapses and bits of the new frame of the next GoP arrive, the Buffers are emptied deleting the remaining bits of the previous GoP. A smoothing operation is therefore performed in order to provide the Base layer with a higher priority.

For this reason, as already discussed previously, the decoding quality at lower levels of the tree will result worse. Each of the $2F$ streams coming from these buffers, together with the information about the peer topology sent by the *Topology Manager* of the source, ends up into a *Packetizer* responsible for packets creation and transmission on the Internet. Packets are sent to the *InterFlow Scheduler*, which, as the one discussed for the source, applies a weighted round robin algorithm and serves each flow with an amount of bandwidth proportional to the uplink bandwidth towards each source of children.

In addition to video play-out and forwarding, another important function performed by each peer is the uplink bandwidth estimation towards their children. This task is operated by the *Bandwidth Estimator*, which periodically sends the estimated bandwidth values to the *Bandwidth Statistic Manager* of the source discussed for the video source diagram of Figure 2. The purpose of this function was to achieve the best performance in small- and medium-size networks. Conversely, if we consider large networks, the algorithm can be slightly modified to be more scalable, but with worse performance. This part of the system is completely general and any bandwidth estimation algorithm can be plugged in. The choice of it goes beyond the purpose

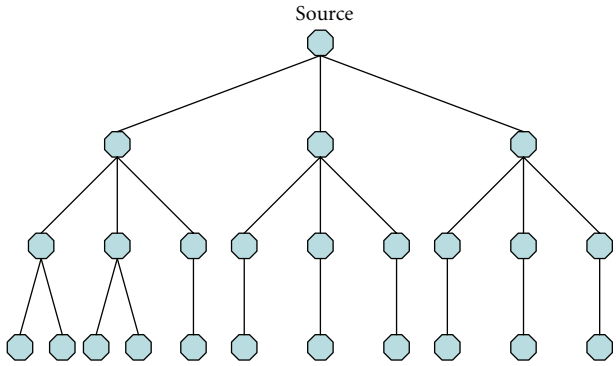


Figure 4: Construction of a tree with 24 peers and $F = 3$.

of this paper. In addition, a sophisticated algorithm to predict the bandwidth in the short or middle term [23, 24] can be applied.

3.3. Organization of the Tree. In order to have our protocol working efficiently by applying FGS, it is obvious that a node with a low uplink bandwidth behaves as a bottleneck; if it was located in the upper part of the tree it would penalize all it descendants. Thus, the best peers should be located at the top of the tree.

For this reason, taking into account that link bandwidth varies in time, the Topology Manager has to optimize the tree topology runtime, in order to avoid configurations with bottlenecks at the highest levels of the tree. We will assume that the downlink bandwidth of each peer is higher than the corresponding uplink bandwidth. In fact:

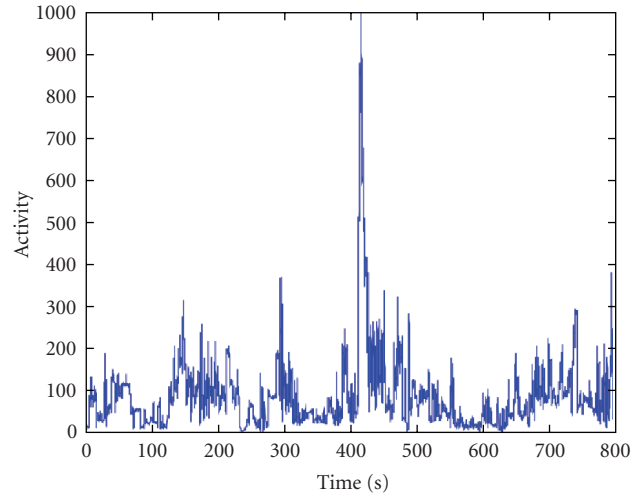
(1) there are many scenarios (e.g., ADSL) where the uplink bandwidth is lower than downlink bandwidth;

(2) the uplink bandwidth is shared with tree nodes whereas the downlink bandwidth of each peer is used as a whole to connect them with their parent only.

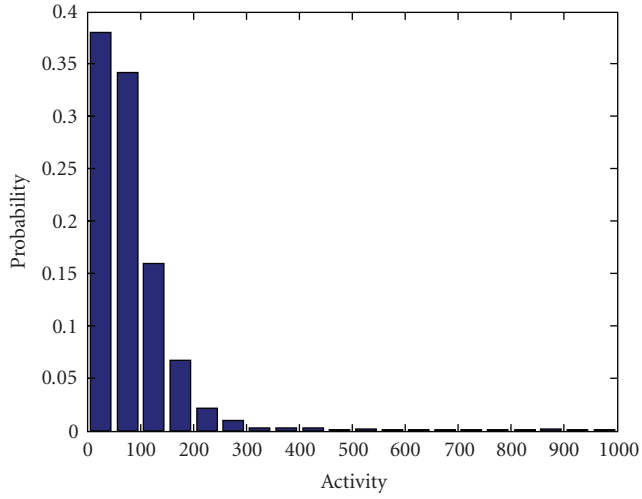
The Topology Manager obtains the bandwidth estimations every T_P seconds ($T_P = 10$ seconds in our implemented system). Then, every T_T seconds ($T_T = 30$ in our implementation) it updates the tree topology as follows: it chooses the F peers with the highest uplink bandwidth, and connects them as children of the source. Then, for each of these peers, the same operation is repeated, choosing the next F peers with the highest uplink bandwidths. This algorithm is run recursively until all the peers get a position in the tree. To optimize the bandwidth performance and to further balance the tree, the nodes in the last level (the leaves) are evenly distributed among the peers in the upper level of the tree (the parents nodes) in a round robin fashion; this avoids the scenario where some peer is overloaded with it uplink bandwidth whereas others are not. Figure 4 shows an example of a tree built as explained, with 24 peers and $F = 3$.

Basically, this tree is balanced and complete (all the levels are full except for the bottom level).

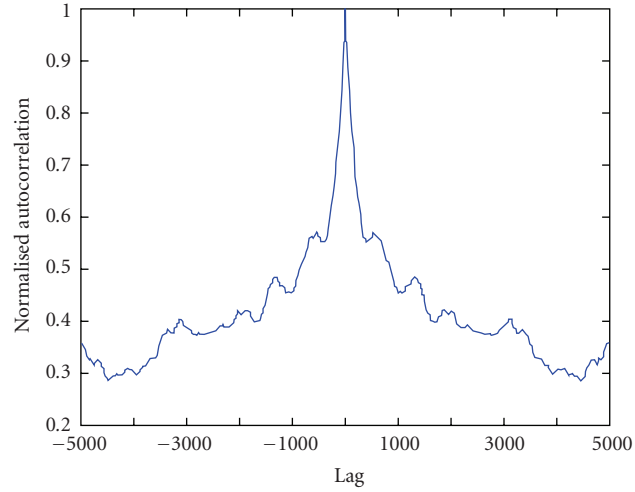
Let us note that changes in the topology structure can be deleterious for video decoding of peers; changing their



(a) Activity process

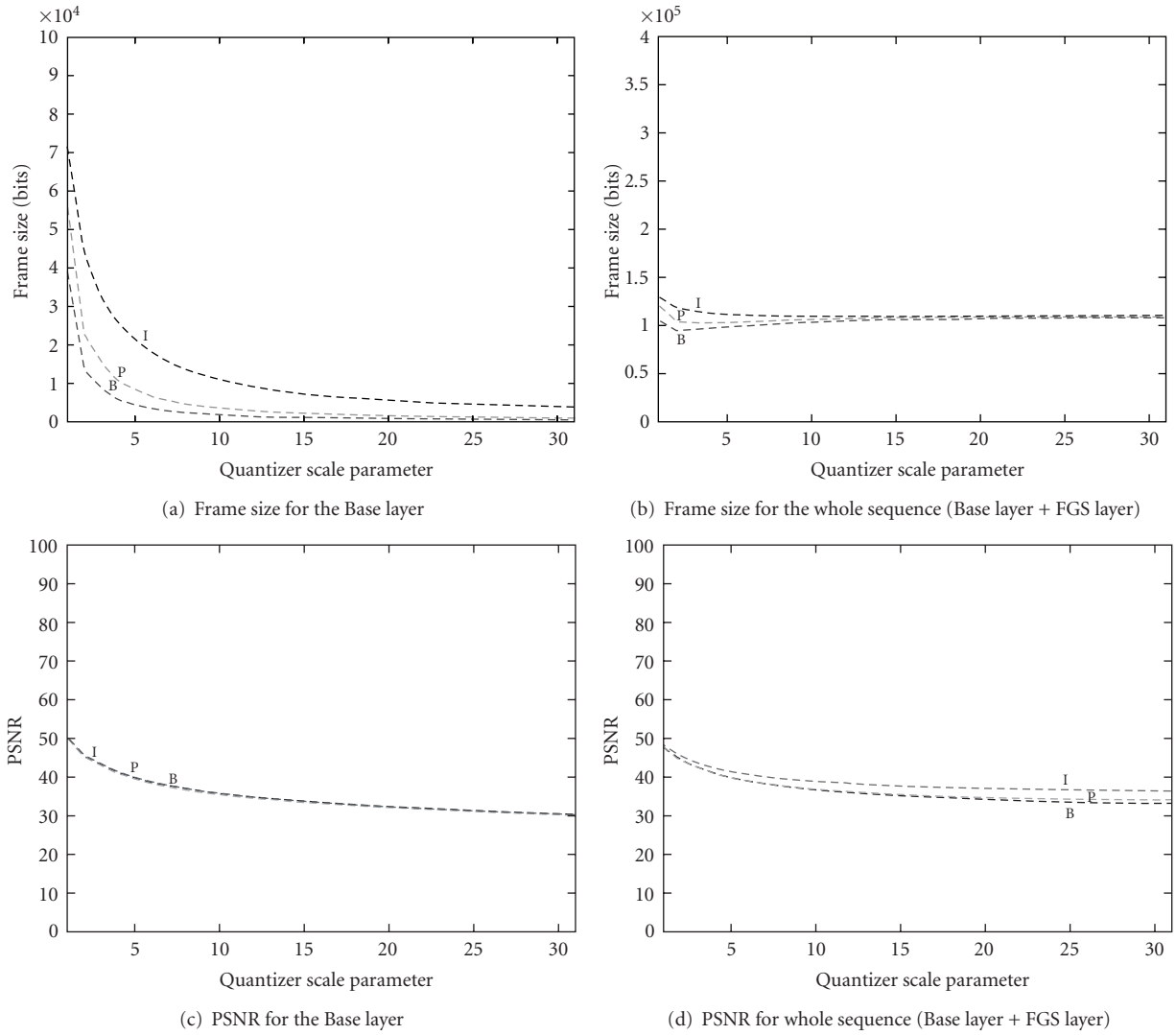


(b) Probability density function (pdf)



(c) Normalized autocorrelation function

Figure 5: Time behavior and first-and-second order statistics of the activity process of the sequence "BBC Planet Earth documentary".



(a) Frame size for the Base layer

(b) Frame size for the whole sequence (Base layer + FGS layer)

(c) PSNR for the Base layer

(d) PSNR for whole sequence (Base layer + FGS layer)

Figure 6: Rate-Distortion curves of the video sequence "BBC Planet Earth documentary".

position in the tree can cause sudden changes in delays, thus increasing the delay jitter as well. In addition, let us observe that this does not happen only when a peer changes level in the tree, but also if either it changes it parent or one of it predecessors changes level or predecessor. However, two important observations can be made in favor of such a strategy:

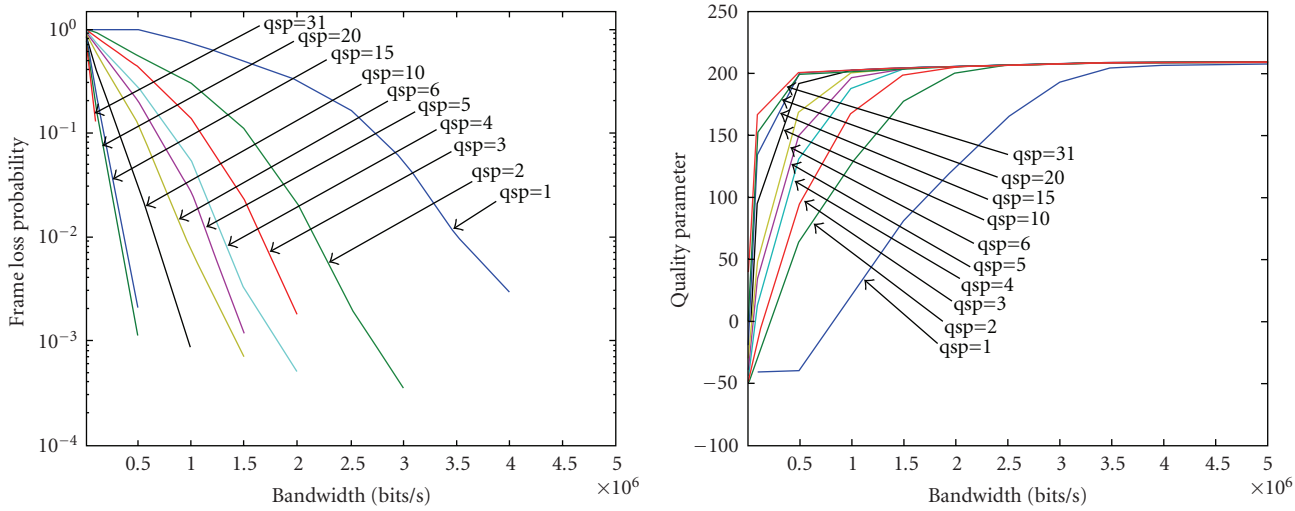
(1) since the bandwidth values are passed under the Topology Manager EWMA ($EWMA_{TM}$) filter (see Section 4), this strategy is not affected by occasional bandwidth changes. Changes in the tree topology occur in the event of serious and lasting bandwidth changes; in this case changes are likely to optimize performance;

(2) the delay jitter caused by topology changes can be recovered at destination through the application of intelligent compensation buffers and adaptive media play-out techniques (see e.g., [25, 26]).

Another observation is that both the interval durations T_P and T_T, and the parameters for all the EWMA filters used in our implementation were chosen empirically after a large number of experiments in order to optimize the tradeoff between system responsiveness to network bandwidth variations and the amount of signaling traffic.

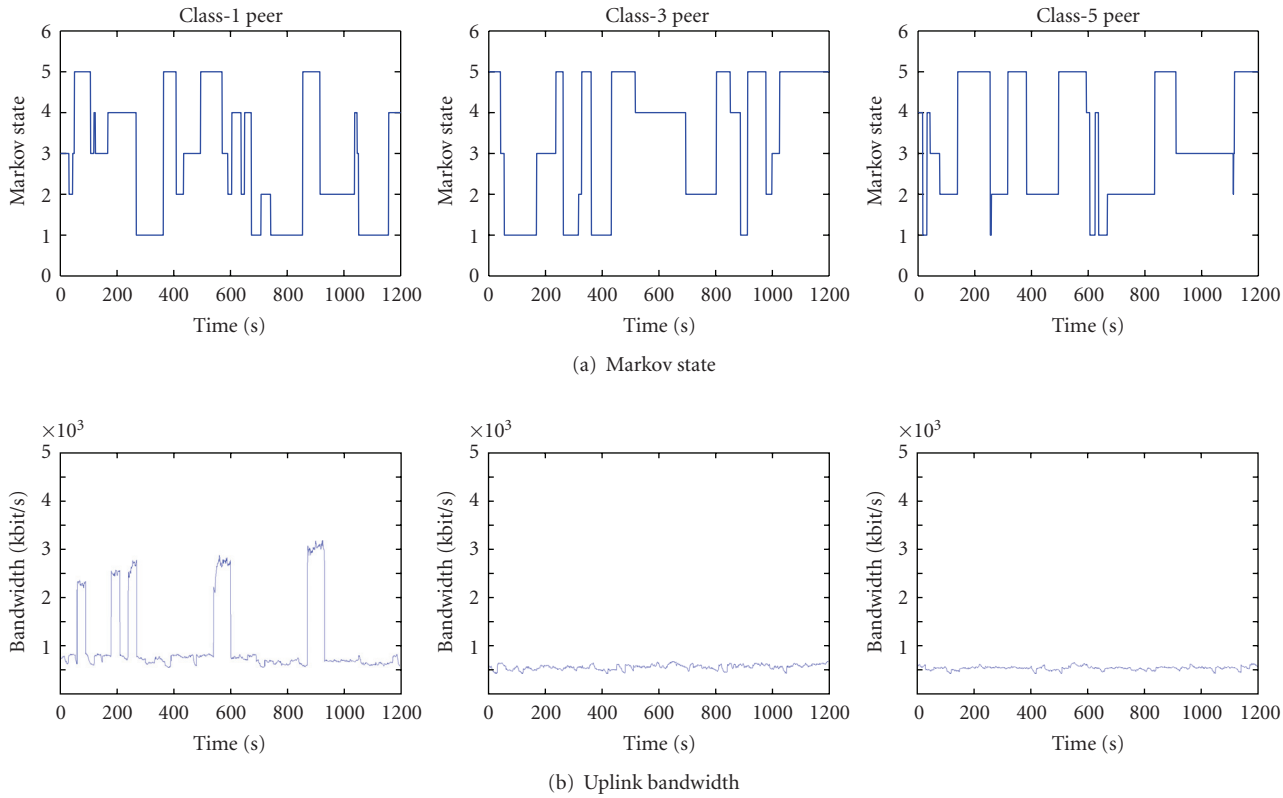
4. Case Study

In this section we define a case study to analyze the performance of the proposed system. The target is to demonstrate the gain achieved by applying FGS encoding against classical video streaming over P2P, in terms of the peak signal-to-noise ratio (PSNR) evaluated on the video flow received at destination from each peer. More specifically, Section 4.1 introduces our case study; Section 4.2 describes the bandwidth generation simulator we developed to generate the processes of the uplink bandwidth at the overlay network level; Section 4.3 analyzes some statistics of the



(a) Frame loss (b) Quality parameter

Figure 7: Frame loss percentage and quality parameter for the video sequence "BBC Planet Earth documentary.



(a) Markov state

(b) Uplink bandwidth

Figure 8: Markov chain state and the generated uplink bandwidth for three representative peers of the tree.

considered sequence; Section 4.4 contains both the results of the overlay network behavior analysis and the performance analysis at the application level. Finally, Section 4.5 shows the comparison of the platform we propose in this paper against other approaches currently used.

4.1. Case Study Description. According to what we said in the previous section, we consider a video distribution platform with centralized control by the Topology Manager.

We will assume that all the peers, included the source, have set the same fan-out parameter, F, representing the

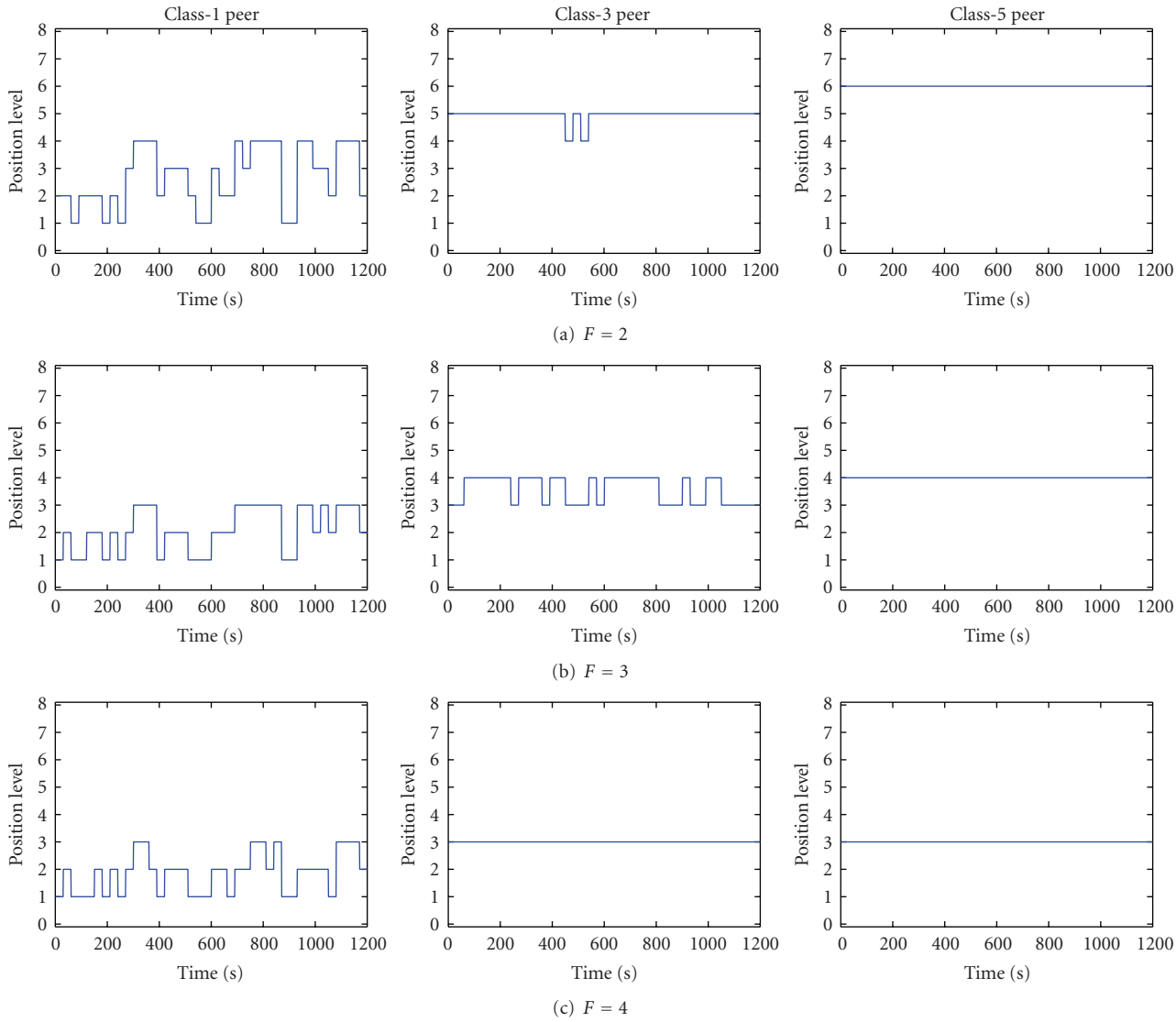


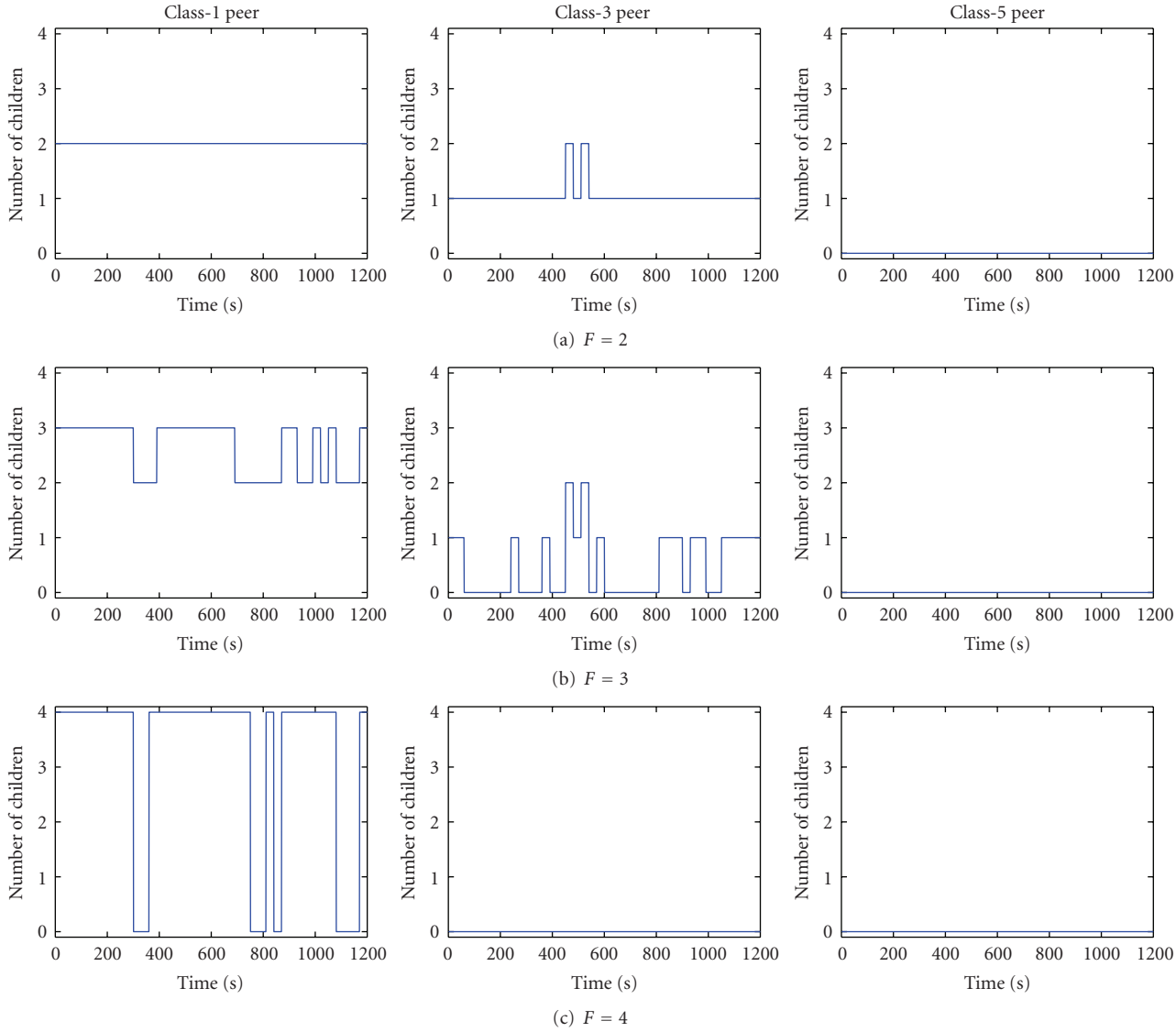
Figure 9: Time behavior of peer position in the tree of three representative peers.

maximum number of peers that can be attached as children in the distribution tree.

We will carry out a steady-state analysis, assuming that the number of peers in the network, hereafter referred to as N, remains constant for the whole duration of the simulation. In this way our study does not depend on the particular algorithm used to manage the topology structure when peer arrivals or departures occur. Therefore, the only job of the Topology Manager is to rearrange the tree according to bandwidth variations. The management of transitory peers will be discussed in the Future Work section.

Peers are grouped in $C = 5$ different classes, each characterized by different Internet access link performances, and different average values of the uplink bandwidth. As previously explained, the downlink bandwidth of each peer is assumed to be much higher than the corresponding uplink bandwidth. Therefore, along this section we will refer to the uplink bandwidth as the *bandwidth*, unless explicitly mentioned. Classes are organized with decreasing average bandwidth values: for example, peers in the third class have higher bandwidth values than peers in the fourth class. We have assumed that the source is a high-bandwidth server with an uplink of 5 Mbit/s.

4.2. Bandwidth Generation Simulator. In real scenarios bandwidth apparently available to a peer (e.g., the ADSL access of it domestic connection) might be not true, and the peer actually can make use only of a small fraction of it, for example due to the intensive use of file sharing programs or bandwidth sharing with other users in the same LAN. For this reason, in order to model the bandwidth variations in a realistic manner, we realized a bandwidth generation simulator at the overlay network level. It creates the desired bandwidth sequences first generating intermediate sequences, by using a modified version of the Switched Batch Bernoulli Process (SBBP) (see [20, 27]), the most general Markov



(a) $F = 2$

(b) $F = 3$

(c) $F = 4$

Figure 10: Time behavior of the number of children in the tree of three representative peers.

modulated process in the discrete-time domain. Then it creates the final sequences by applying a EWMA filter, in the following indicated as the *Bandwidth Variations EWMA (EWMA*$_{\mathrm{BV}}$*)*, with parameter $\beta^{(\mathrm{BV})}$ ($\beta^{(\mathrm{BV})} = 0.8$ in our simulations (see Section 3.1 for details on the EWMA filter). This EWMA filter is applied to the temporary sequences generated at the first step, to achieve the final smoothed traces. Therefore, the uplink of each peer is simulated with two blocks in cascade: a bandwidth generator block and a EWMA$_{\mathrm{BV}}$ filter block for the smoothing needed to obtain the desired dynamics. The bandwidth generation tool and the uplink bandwidth processes used for all the simulations are available at [28].

The first step of the generation algorithm works as follows. Let $i \in \{1, \ldots, C\}$ be the generic class, grouping peers with similar Internet accesses, and let $P^{(i)}$ be the generic peer belonging to the class i.

Let $BW_{P^{(i)}}(n)$ be the uplink bandwidth process of the peer $P^{(i)}$.

The process $BW_{P^{(i)}}(n)$ will be considered as modulated by a L-state underlying Markov chain, where the mean permanence is a geometricall-distributed random variable with mean value M_μ, assumed equally for all the classes. Once the underlying Markov chain leaves a state, it moves to one of the other states with the same probability. To this end we define the generic element of the transition probability matrix of the underlying Markov chain as follows:

$$Q^{(i)}_{[h,k]} = \begin{cases} \dfrac{1}{(L-1)}, & \text{if } h \neq k \text{ with } h,k \in \{1,\ldots,L\}, \\ 1 - \displaystyle\sum_{j \neq k} Q^{(i)}_{[h,j]}, & \text{if } h = k \text{ with } h,k \in \{1,\ldots,L\}. \end{cases} \tag{2}$$

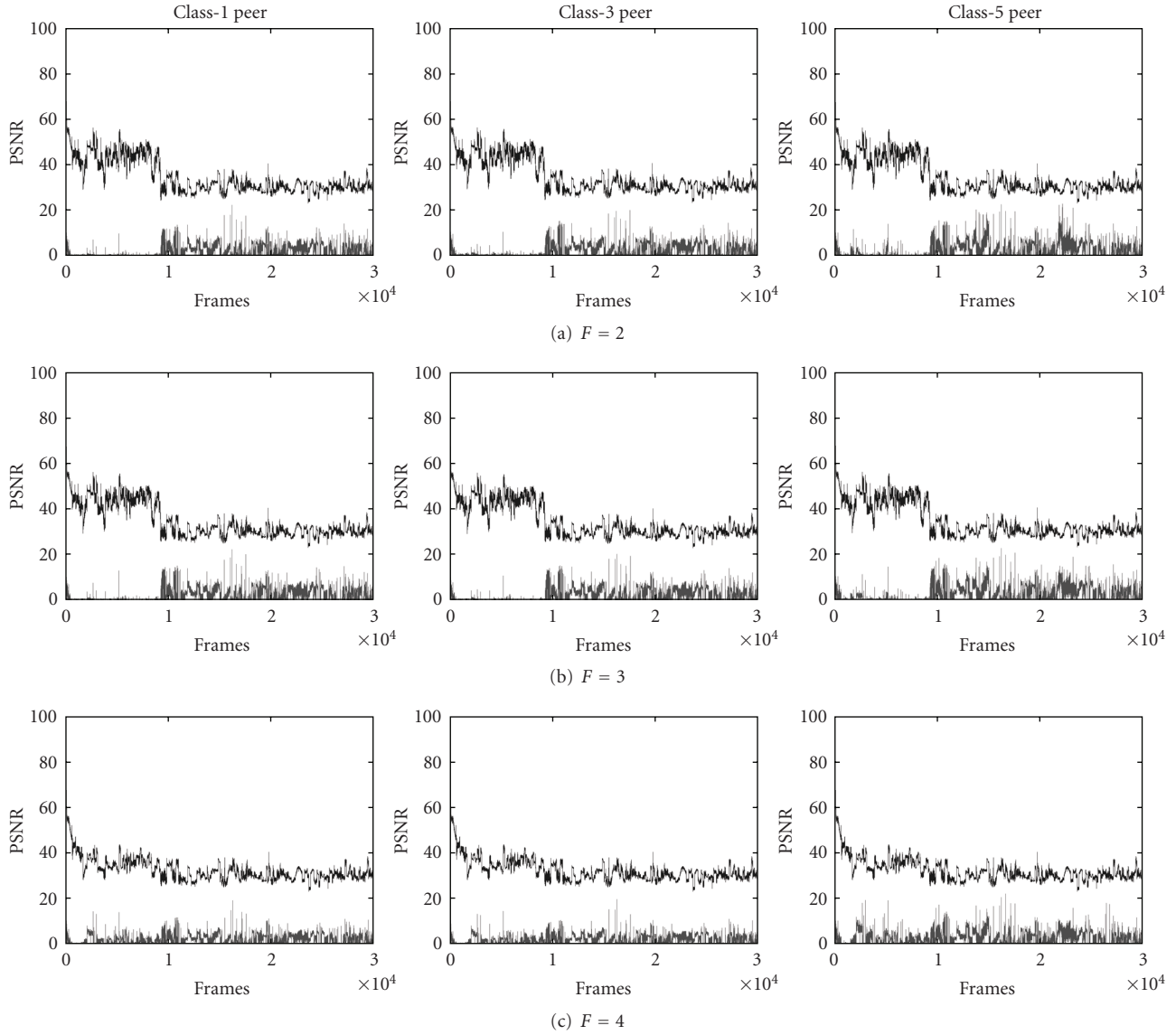


Figure 11: PSNR for the Base layer (black) and gain achieved using FGS (gray).

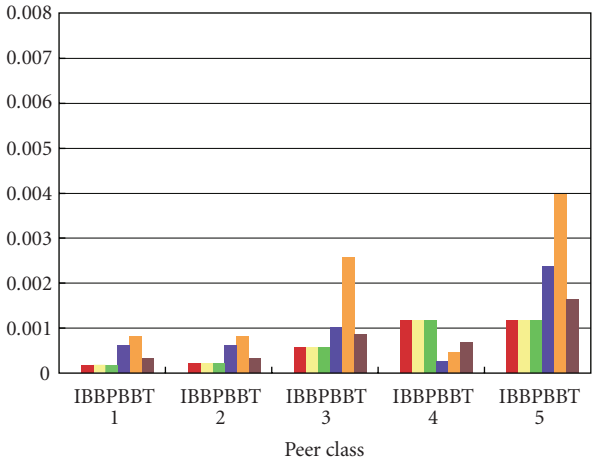
During the permanence of the Markov chain in the generic state h, with $h \in \{1, \ldots, L\}$, the value of the uplink bandwidth at the slot n, $BW_{P^{(i)}}(n)$, is randomly chosen according to a Gaussian distribution with a state-dependent mean value $\mu^{(i)}(h) = [(h - (i-1)) \cdot 10\% + 1] \cdot \widehat{W}$, with $\widehat{W}$ equal to the average uplink value of the intermediate state $h = \lceil L/2 \rceil$, and a standard deviation $\sigma^{(i)}(h) = 0.08 \cdot \mu^{(i)}(h)$.

In our case study we have used a slot duration of 1 seconds, the number of states of the Markov chain $L = 5$, a mean duration of the permanence in each state of the underlying Markov chain of $M_\mu = 60$ slots, and $\widehat{W} = 370$ kbit/s.

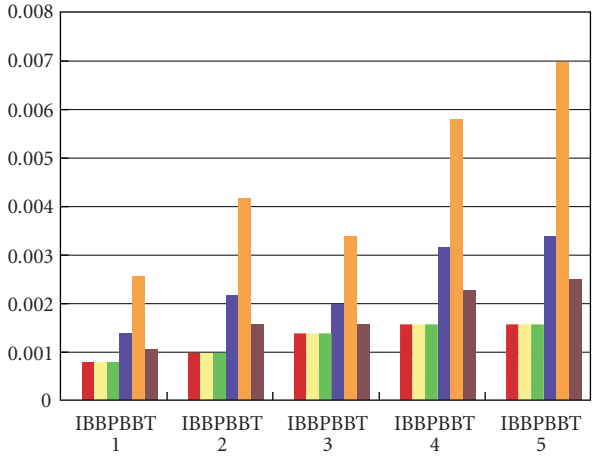
Let us stress that the bandwidth generation technique and values described above are only provided for the reproducibility of the results achieved: any other bandwidth generation process can be used. In fact, other bandwidth processes have been applied by the authors, giving equally significant results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

4.3. Statistical Analysis of the Considered Sequence. For all the experiments along the paper we have considered a sequence of the video "BBC Planet Earth documentary". The sequence has a duration of 20 minutes and is encoded with a 176 × 144 QCIF format, at a frame rate of 25 frame/seconds, and using as Group of Pictures (GoP) structure with the pattern IBBPBB. For the sake of completeness, in this section we show a statistical analysis of this sequence.

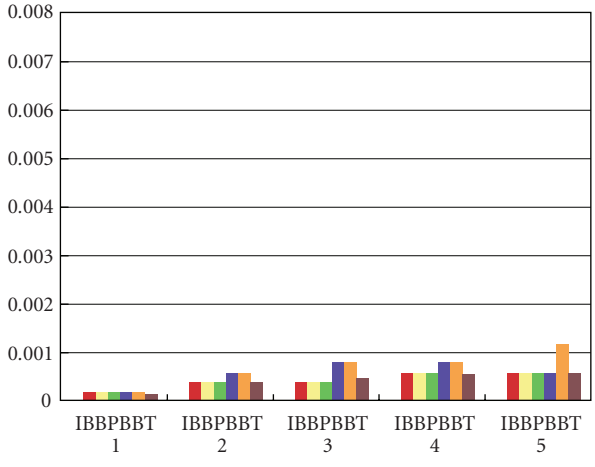
The first parameter we analyze is the activity process of the sequence, which is a spatial property: the greater the spatial frequency range, the greater the activity. Activity has a strong impact on the behavior of the encoding results, in both frame size and PSNR: frames with a higher activity have a higher size; on the other hand, if a rate controller



(a) $F = 2$



(b) $F = 3$



(c) $F = 4$

Figure 12: Frame loss percentage for the five representative peers.

Table 1: Total number of levels in the tree and number of peers in the last level.

F	Number of levels	Peers in the last level
2	6	22
3	4	45
4	3	64

is applied to control the frame size, frames with a higher activity present a lower PSNR. Activity is determined by the scene of the movie only, and does not depend on the encoding technique and parameters used in the encoding process. Its time behavior for the considered sequence is shown in Figure 5, together with it first- and second-order statistics in terms of probability density function (pdf) and normalized autocorrelation function. Figure 6 presents the Rate-Distortion curves, calculated as in [20–22] for both the Base layer and the whole sequence (the latter is considered as the aggregation of both the Base and the FGS layers). The curves represent the frame size (Figures 6(a) and 6(b)) and the PSNR (Figures 6(c) and 6(d)) for the three encoding modes (I, P and B) as a function of the quantizer scale parameter, qsp. The strongly decreasing behavior of the curves for low values of qsp in Figure 6(b) is due to the corresponding behavior of the Base layer in the same range, since the Base layer is predominant over the FGS layer within this range. In Figure 6(d) we can observe that, as expected, the presence of the FGS layer allows the quality of the overall sequence to reach a high value even when the Base layer is encoded with high values of qsp, and therefore the overall PSNR is almost independent on the used quantizer scale. It follows that a low value for qsp would increase the frame size of the base layer too much, causing potential frame losses in case of bandwidth oscillations; on the other hand, a high value for qsp would increase the size of the aggregation of Base and FGS layers to be transmitted and decrease the PSNR. However, in this latter case, the encoded stream is more robust to network losses, thanks to the higher percentage of FGS bits, that can be truncated in any point of the stream. Therefore, in real scenarios, that is in presence of bandwidth oscillations, it is obvious the importance of the Rate Controller: according to bandwidth, it has to determine the size of the Base, and consequently the size of the FGS layer, in order to maximize the encoding quality, but getting frame losses as fewer as possible. For this reason, in our analysis we will consider the following quality of service (QoS) aspects simultanously:

(1) the encoding quality at the source side;

(2) the loss percentage in the network.

In fact, it is misleading considering the peak signal-to-noise ratio (PSNR) parameter at the source only, that is, for all the frames encoded by the source, since the quality perceived at destination is strongly degraded if network losses occur. On the other hand, it is misleading as well averaging PSNR over only the noncorrupted received frames because in this case the estimation would result not fair (e.g., because low-quality frames are small, and therefore more likely to be received).

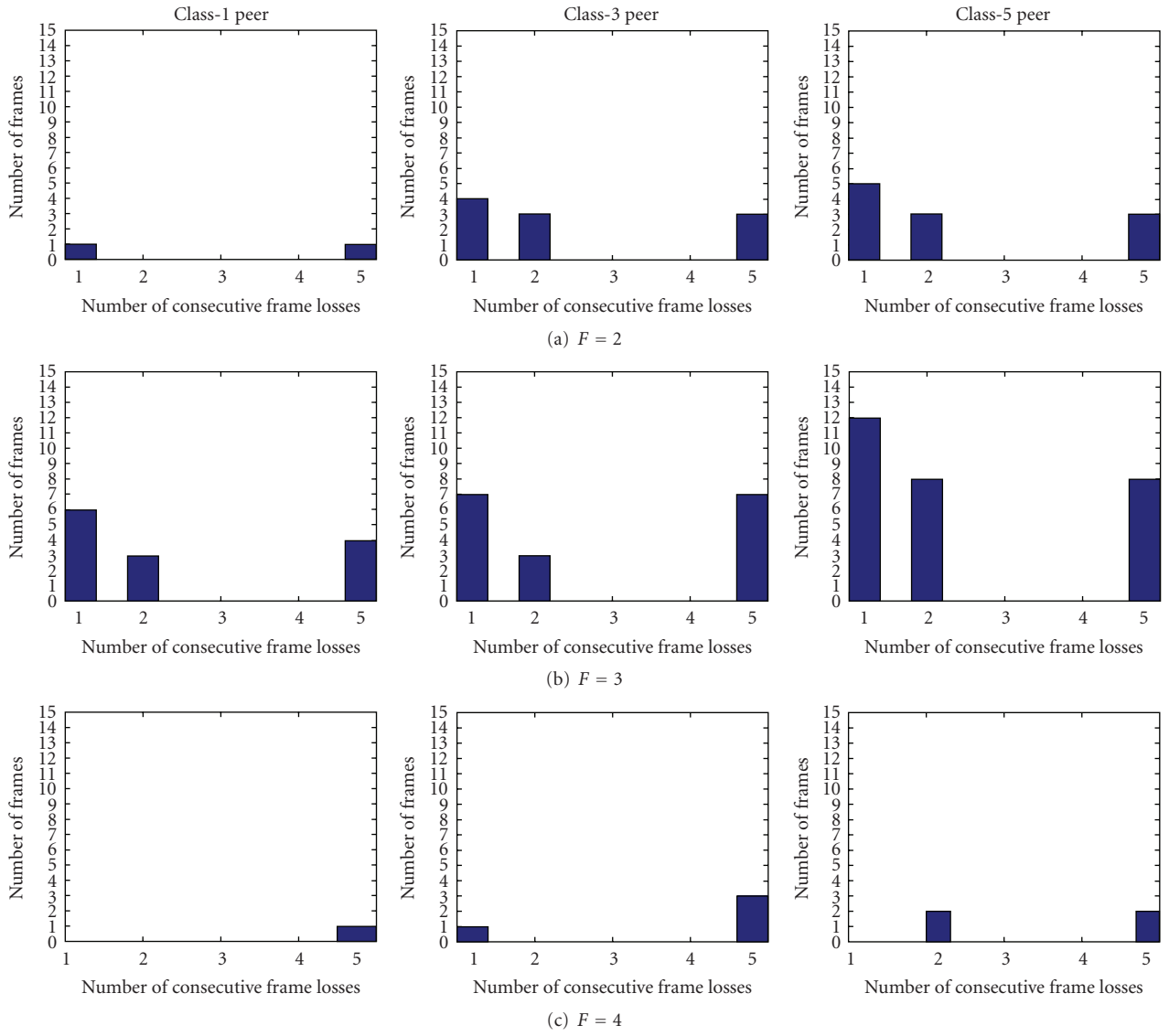


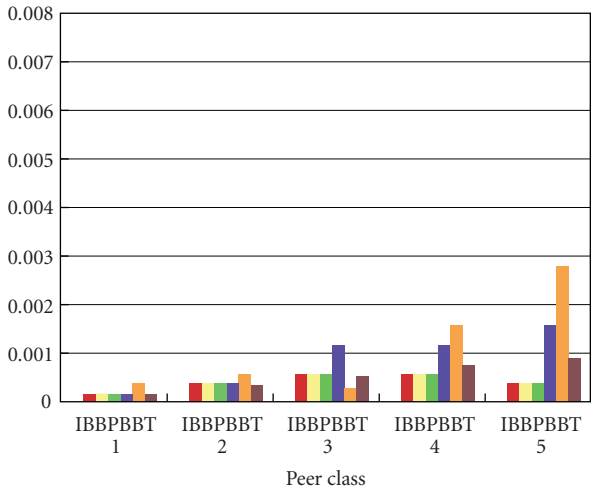
Figure 13: Histograms of consecutive frame losses.

Table 2: Average PSNR and Q for representative peers in each class.

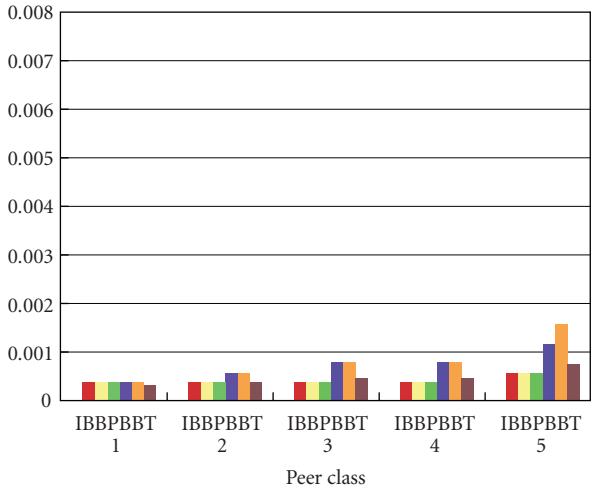
	PSNR			Q		
Class	$F = 2$	$F = 3$	$F = 4$	$F = 2$	$F = 3$	$F = 4$
1	36.2427	36.3263	34.5218	0.409232	0.446816	−0.365208
2	35.9045	36.0584	34.1065	0.257003	0.326285	−0.552064
3	35.8657	36.0403	33.9856	0.239575	0.318134	−0.60648
4	35.8563	36.0343	33.9228	0.235342	0.315454	−0.634722
5	34.4134	34.4915	32.3582	−0.413992	−0.378809	−1.33883

Table 3: Average PSNR and Q for representative peers in each class (*Dynamic* + *Base* method).

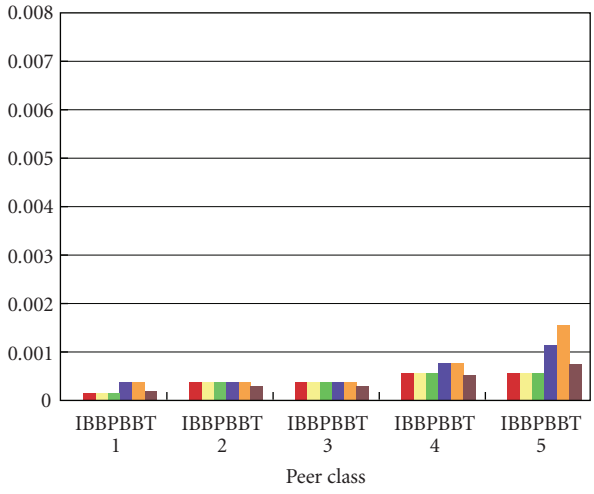
	PSNR			Q		
Class	$F = 2$	$F = 3$	$F = 4$	$F = 2$	$F = 3$	$F = 4$
1–5	34.186	34.108	32.3634	−0.411642	−0.371378	−1.33648



(a) $F = 2$



(b) $F = 3$



(c) $F = 4$

Figure 14: Frame loss percentage for the five representative peers (*Static* + *Base/FGS* and *Static* + *Base* methods).

For this reason, in order to take into account the encoding quality and the frame rate reduction simultaneously, we have used a *Quality parameter*, defined in [29] through a heuristic formula, which 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:

$$Q = 0.45 \cdot psnr + \frac{(fr - 5)}{10} - 16.9, \tag{3}$$

where *psnr* is the PSNR value measured at the destination averaged only on the frames received at destination, *fr* is the frame rate of the video sequence received at destination, excluding frames corrupted due to damages of the Base layer for network losses. The constant coefficients in (3) were calculated in [29] by evaluating the data set obtained in a survey, and assuming a minimum acceptable frame rate of 5 frame/seconds. According to the above definition, the greater the PSNR and the frame rate at destination, the greater the Q parameter. Figure 7 shows the frame loss percentage and the Q value against the mean available network bandwidth for different values of *qsp*. Negative values of Q represent a decoded frame rate at destination of less than 5 fps, which is the minimum threshold assumed in [29] for an acceptable quality.

4.4. Numerical Results. This section contains numerical results concerning the overlay network behavior analysis (Section 4.4.1), and the performance analysis at the application level (Section 4.4.2).

4.4.1. Overlay Network Behavior Analysis. Here we will show results on some peers randomly chosen as representative of their corresponding class. Their uplink bandwidth has been generated using the tool described in Section 4.2.

We have fixed the number of peers to 85, including the source, and analyzed the performance for three different values of the fan-out parameter, $F \in \{2, 3, 4\}$. The 84 peers (the source has been considered as a high-bandwidth server with 5 Mbit/s uplink) have been randomly chosen from five different classes of peers (peers in different classes have different ranges of bandwidth values, as discussed in Section 4) according to a uniform distribution. The resulting peers were distributed as follows:

(i) 12 peers of class 1;

(ii) 23 peers of class 2;

(iii) 17 peers of class 3;

(iv) 19 peers of class 4;

(v) 13 peers of class 5.

We have generated traces for a time interval of 20 minutes, equal to the length of the considered movie trace. The last level of the obtained trees is partially filled for $F \in \{2, 3\}$, whereas it is completely filled for $F = 4$. Table 1 shows the number of tree levels and the number of peers lying in the last level for the three considered values of F.

Table 4: Average PSNR and Q for peers in each class for (*Static* + *Base/FGS* method) with $F \in \{2, 3, 4\}$.

	PSNR			Q		
Class	$F = 2$	$F = 3$	$F = 4$	$F = 2$	$F = 3$	$F = 4$
1	35.3316	35.4134	34.1984	0.3764	0.4189	−0.3744
2	35.2096	35.3491	34.1605	0.21363	0.30123	−0.4546
3	34.8912	34.773	33.6131	0.2111	0.2918	−0.5101
4	34.1513	34.0378	33.0019	0.2075	0.2744	−0.5321
5	33.6781	33.7644	32.5137	0.1012	0.1791	−0.559

Table 5: Average PSNR and Q for representative peers in each class (*Static* + *Base* method).

	PSNR			Q		
Class	$F = 2$	$F = 3$	$F = 4$	$F = 2$	$F = 3$	$F = 4$
1–5	30.6046	30.675	30.2024	−2.12792	−2.09624	−2.23894

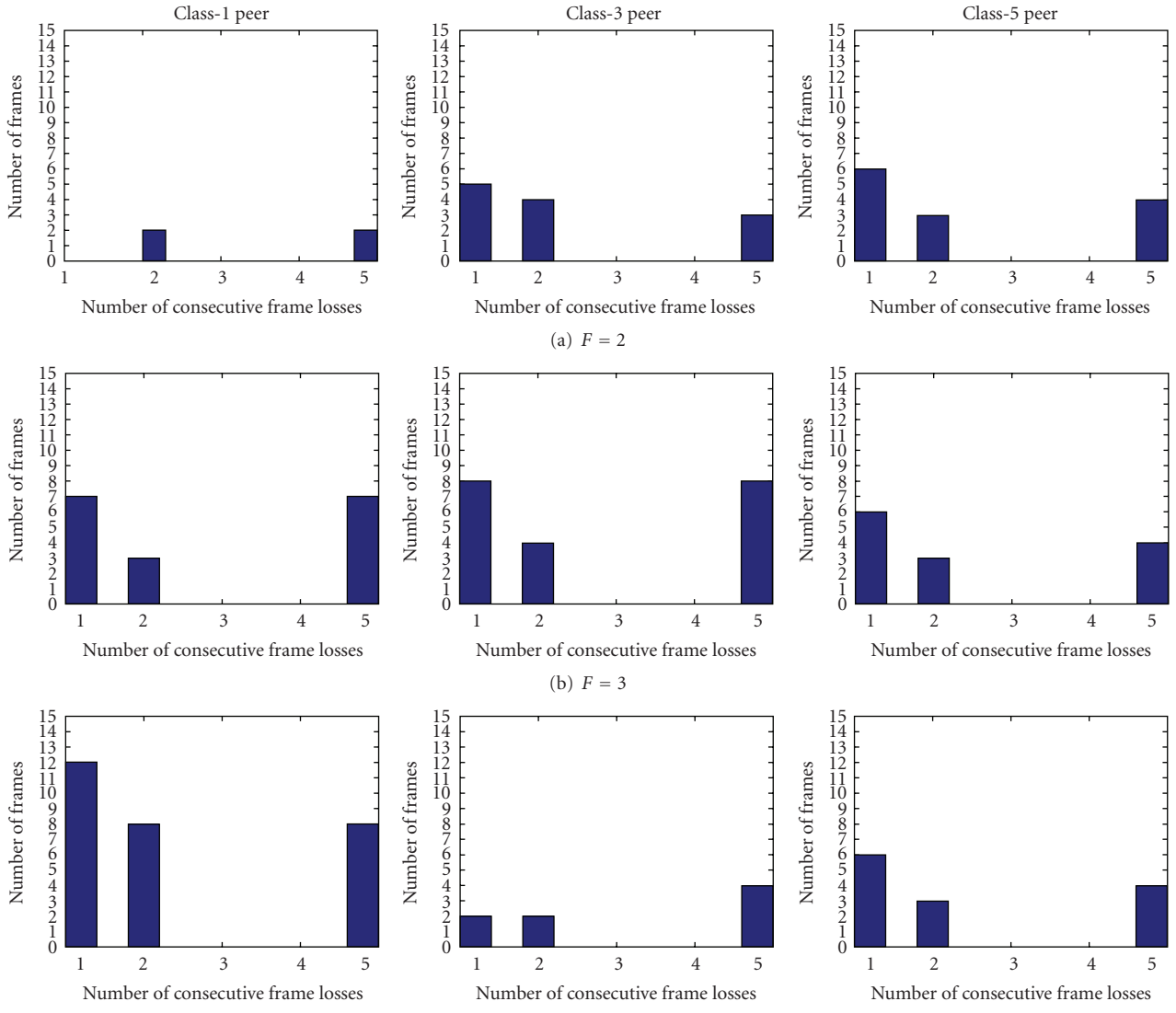


Figure 15: Histograms of consecutive frame losses (*Static* + *Base/FGS* and *Static* + *Base* methods).

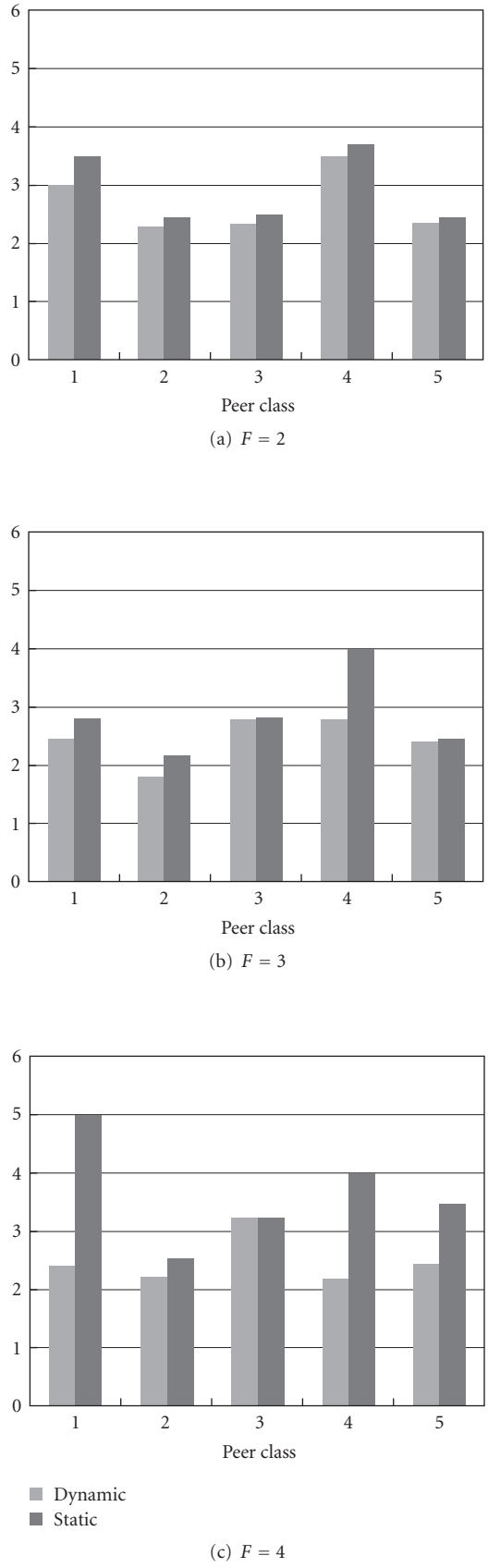


Figure 16: Period of average length of consecutive losses for the five representative peers - *Static* and *Dynamic* methods.

In order to work with a more efficient scheme, as already introduced in Section 3.3, the leaf nodes have been assigned to the peers in the last but one level in a round-robin fashion.

We first show in Figures 8, 9, and 10 how the system works, analyzing the behavior of three representative peers (peer of class 1 on the left column, peer of class 3 in the middle one, and peer of class 5 on the right one) taken as representatives of their corresponding classes.

More specifically, for each considered peer, Figures 8(a) and 8(b) present the state of the underlying Markov chain of it uplink bandwidth, and the generated uplink bandwidth, respectively, and show how the Markov model is able to impress the desired temporal correlation on the bandwidth process. Then in Figures 9(a), 9(b) and 9(c) we observe how peers change level during their lifetime according to the behavior of the instantaneous uplink bandwidth from the source, and therefore are strongly influenced by the class they belong to.

Figures 10(a), 10(b), and 10(c) show the time behavior of the number of children for the considered peers. A peer of class 5 (third column) is usually a leaf of the tree, that is, it has no children, and it does not frequently become an internal node. Different observations hold for the considered peer of class 1 (first column), which has always a positive number of children for all the values of $F \in \{2, 3, 4\}$. For $F = 3$, the peer of class 1 changes level quite often, and, consequently, it gets either 2 or 3 children according to the level where it is situated. The peer of class 3 (second column) changes it level quite often for $F = 3$ and, for this reason, it number of children varies from 0 to 2.

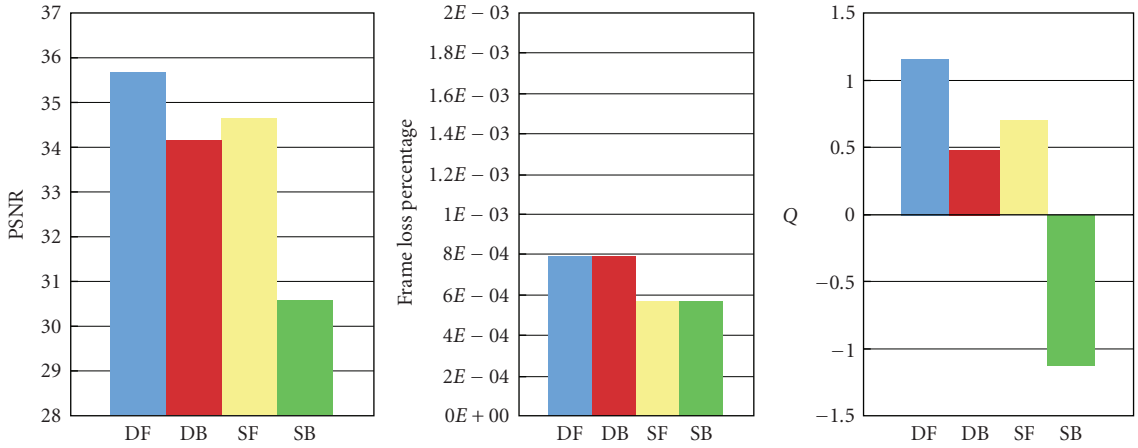
The reader notes that Figures 8, 9, and 10 are strictly correlated. Peers with low bandwidth values get lower positions in the tree, whereas peers with higher bandwidth values lie in the upper levels.

For example, around the time instant 200 Seconds, the peer of class 1 (first column) has a bandwidth which rapidly increases (see Figure 8(b)) and, consequently, this peer is immediately moved from level 2 to level 1 for $F \in \{2, 3, 4\}$. Consequently, the number of children increases and it is exactly equal to F, for each $F \in \{2, 3, 4\}$.

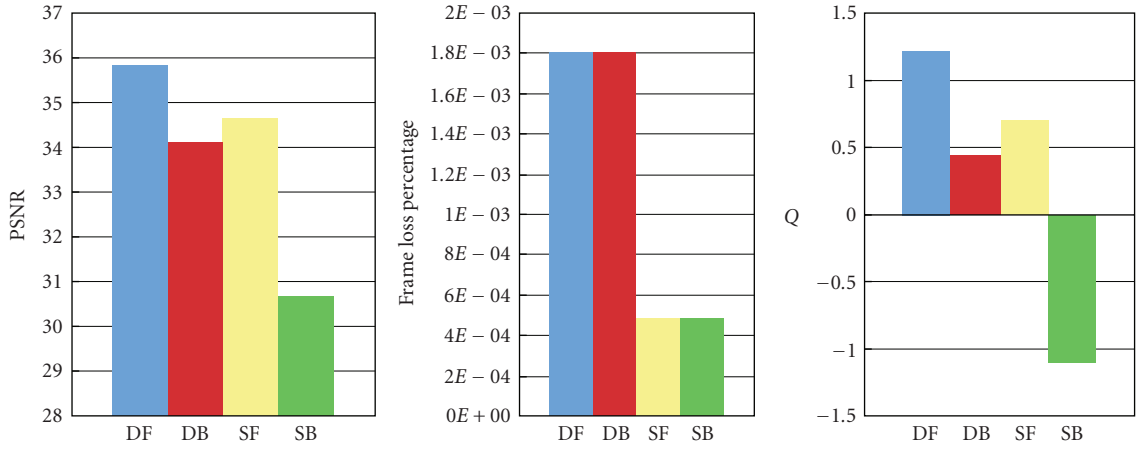
4.4.2. Performance Analysis at the Application Level. In this section we will show some statistics concerning the quality of the video received from the source by the peers we are considering as representative of each class.

Table 2 shows the average PSNR and the average Quality parameter Q defined in Section 4.3 (3), measured at destination on the overall sequence by peers of each class for different values of F.

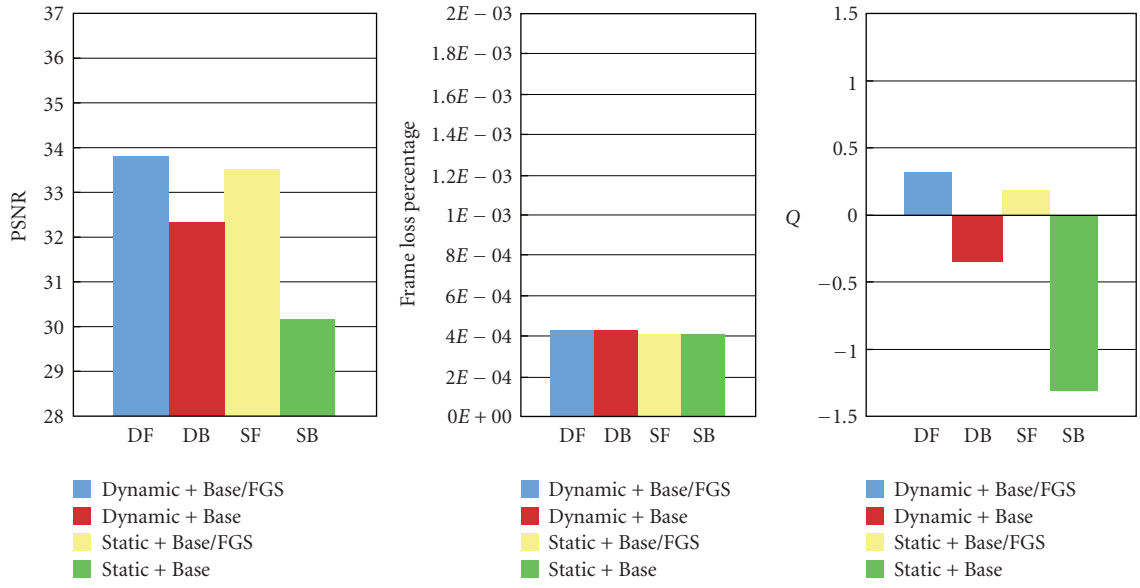
Figure 11 shows the PSNR per frame for the videos received by the three representative peers (one of class 1, one of class 3 and one of class 5) when both Base and FGS layers are transmitted (in black), and the PSNR difference between the video transmitted with both Base and FGS layers and the video transmitted with just the Base layer (in gray). For each value of F, when both Base and FGS layers are used, the PSNR is always higher, or, at the most, equal (difference equal to 0) to each other.



(a) $F = 2$



(b) $F = 3$



(c) $F = 4$

Figure 17: Global results for each method.

As discussed in Section 2, we have considered the video in QCIF format using the pattern IBBPBB as Group of Picture (GoP) structure. As a consequence of the application of this pattern, we remark that when a frame is lost, an error propagation action occurs as follow.

(i) If an I-frame gets lost then all the other frames of the same GoP and the two ending B frames of the previous GoP are lost as well.

(ii) If a P-frame gets lost then the two starting and the two ending B frames of the same GoP are also lost.

(iii) If a B-frame gets lost then nothing happens since B frames are not needed for the reconstruction of the other frames.

For each frame of the GoP, we have also computed it loss percentage (for $F \in \{2, 3, 4\}$) shown in Figure 12. The reader notes that, as introduced in Section 3.1, I-frames have the priority to be transmitted with respect to the other GOP frames because an I-frame loss causes 7 more frame losses. It is for this reason that, in our system, it is more difficult to incur in I-frame losses (i.e., charts in Figure 12 do not show any loss of I-frames). Each chart in the table has the peer class as x-axis, and the frame loss percentage as y-axis. For each value in the x-axis there are five columns related to the six GOP frames (since the one related to I-frame is always 0).

Figure 13 shows a histogram of the number of consecutive frames which are lost, calculated for the three representative peers and for $F \in \{2, 3, 4\}$. Let us note that there are no more than 5 consecutive frame loss: the reason is no I-frame has been lost (an I frame should have caused a number of eight consecutive lost frames: BBIBBPBB). Instead, in some cases, there are 5 consecutive lost frames, due to the corruption of the Base layer of a P-frame (in this case both the two B-frames before and the two B-frames after the lost P-frame cannot be decoded, and therefore the following pattern is lost: BBPBB). Let us note that a combination of losses that causes three or four consecutive lost frames does not exist.

4.5. FGS Performance Assessment. In this section we show how our video transmission system greatly improves on scenarios where either the tree is created but is not updated according to peer uplink bandwidth variations (and therefore it is static), or the enhancement layer for video transmission is not sent at all (i.e., FGS encoding is not applied). More specifically, in the following we will consider for comparison the four combinations of the above situations that can be classified and called as follows:

(1) *Dynamic + Base/FGS* (DF): a peer-to-peer network (with $F \in \{2, 3, 4\}$) where peers are organized with a tree topology which changes dynamically using both Base and FGS layers for video transmission; this is the novel method we propose in this paper;

(2) *Dynamic + Base* (DB):a peer-to-peer network (with $F \in \{2, 3, 4\}$) where peers are organized with a tree topology which changes dynamically; in such a case each peer receives the Base layer only of the whole movie;

(3) *Static + Base/FGS* (SF): a peer-to-peer network where peers are organized in a static tree (peer position is fixed in the tree for the whole transmission), and the source uses both Base and FGS layers for video transmission;

(4) *Static + Base* (SB): a peer-to-peer network where peers are organized in a static tree (peer position is fixed in the tree for the whole transmission); in such a case each peer receives the Base layer only of the whole movie.

For such scenarios, for each class of peers and values of $F \in \{2, 3, 4\}$ we show the PSNR, the Q value, the frame loss percentage and the distribution of consecutive frames losses.

We have already shown the above parameters for the novel method we propose in this paper (i.e., *Dynamic + Base/FGS*) in Table 2 and Figures 12 and 13. As far as the frame loss percentage and consecutive frame losses are concerned, let us notice that only the Base-layer frames contribute to their computation; therefore the methods *Dynamic + Base/FGS* and *Dynamic + Base* produce the same results (shown in Figures 12 and 13) for them. However, the PSNR and Q values for *Dynamic + Base* method are different since the FGS layer gives more details in terms of video quality. Table 3 shows such values. As expected, the PSNR and Q values for each combination of F and peer class are worse than the corresponding PSNR and Q values achieved by applying the *Dynamic + Base/FGS* method, and shown in Table 2.

The PSNR and Q values produced by *Static + Base/FGS* and *Static + Base* methods are shown in Tables 4 and 5, respectively. Similar considerations taken for the Dynamics methods hold: as shown in Figures 14 and 15 they produce the same results for the frame losses and the consecutive frame losses. In addition, from Figure 15 we can note that *Static* methods incur in a number of consecutive frame losses much higher than the ones of *Dynamic* methods.

As seen above for the Dynamics methods, the PSNR and the Q values for the *Static + Base/FGS* method are better than those produced by the *Static + Base* method.

In order to better analyze the impact of the applied method over the number of consecutive frames, we have calculated the average length of the periods of consecutive losses for both *Static* and *Dynamic* methods, for $F \in \{2, 3, 4\}$. The results are shown in Figure 16.

Finally, we have produced Figure 17 showing global results of each of the four methods above discussed for each of the variables of interest (PSNR, frame losses and Q value), and for each value of $F \in \{2, 3, 4\}$. Each value is calculated as the average of each performance parameter over all the peers in the network. It is evident how the PSNR and the Q values are much better for the proposed method (i.e., *Dynamic + Base/FGS*). *Dynamic* methods, for $F \in \{2, 3\}$, present a slightly higher value of frame losses: this is due to the size of the Base layer which is bigger, and therefore more vulnerable, than the one obtained using *Static* methods.

5. Conclusions and Future Work

This paper proposes a multipoint video transmission framework over a heterogeneous content distribution P2P network. The source generates the video flow by using an MPEG-4/FGS encoder, in such a way that the number of losses occurring at the Base-layer stream are minimized, even in the presence of short-term bandwidth fluctuations.

The FGS layer is sent together with the Base layer, but with a lower priority. The source uses a rate controller to regulate the encoding rate of the Base layer, according to an estimation of the bandwidth of the overlay network bottlenck link. A protocol is defined in order to provide the source with this information.

A case study is introduced to evaluate the performance of the proposed framework and the improvements obtained with respect to some reference cases in which FGS is not applied and/or the overlay network topology is not dynamically reorganized.The problem of organizing the tree in realtime is discussed and some techniques are proposed and compared.

As future work we plan to investigate the possibility to apply the proposed scheme for 3D-video streaming on several application scenarios: *three-dimensional television (3D-TV)*, *free viewpoint television (FTV)*, and *multi-view video coding (MVC)*.

Another task to take into account is to perform a statistical analysis of the delay, and in particular the delay jitter that, as discussed in Section 3.3, may result critical for frequent topology changes, particularly for realtime applications.

A feasible solution would be to design an adaptive playout buffer to reduce packet discarding at destination, and a topology management strategy which is able to wisely decide the position of each peer in the tree in order to avoid delay jitter that cannot be compensated even with an adaptive playout buffer.

Last but not least, in presence of high churn rate, the management of transitory peers will be investigated in order to maintain the architecture as more robust as possible.

Acknowledgment

This work is partially supported by the Italian MIUR PRIN 2007 project "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.

References

[1] S. D. Sevetto and K. Nahrstedt, "Broadcast quality video over IP," *IEEE Transactions on Multimedia*, vol. 3, no. 1, pp. 162–173, 2001.

[2] D. Xu, M. Hefeeda, S. Hambrusch, and B. Bhargava, "On peer-to-peer media streaming," in *Proceedings of the 22nd International Conference on Distributed Computing Systems*, pp. 363–371, Vienna, Austria, July 2002.

[3] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, "Scalable application layer multicast," in *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pp. 205–217, Pittsburgh, Pa, USA, 2002.

[4] Y. H. Chu, S. G. Rao, and H. Zhang, "A case for end system multicast," in *Proceedings of the ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, pp. 1–12, Santa Clara, Calif, USA, June 2000.

[5] V. N. Padmanaban, H. J. Wang, and P. A. Chou, "Resilient peer-to-peer streaming," in *Proceedings of the 11th IEEE International Conference on Network Protocols*, pp. 16–27, Atlanta, Ga, USA, November 2003.

[6] M. Castro, P. Druschel, A. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, "SplitStream: high-bandwidth multicast in cooperative environments," in *Proceedings of the 19th ACM Symposium on Operating Systems Principles*, pp. 298–313, Bolton Landing, NY, USA, 2003.

[7] T. T. Do, K. A. Hua, and M. A. Tantaoui, "P2VoD: providing fault tolerant video-on-demand streaming in peer-to-peer environment," in *IEEE International Conference on Communications*, vol. 3, pp. 1467–1472, June 2004.

[8] D. A. Tran, K. A. Hua, and T. T. Do, "A peer-to-peer architecture for media streaming," *IEEE Journal on Selected Areas in Communications*, vol. 22, no. 1, pp. 121–133, 2004.

[9] D. Jurca, J. Chakareski, J.-P. Wagner, and P. Frossard, "Enabling adaptive video streaming in P2P systems," *IEEE Communications Magazine*, vol. 6, pp. 108–114, 45.

[10] T. Nguyen, K. Kolazhi, R. Kamath, S. Cheung, and D. A. Tran, "Efficient multimedia distribution in source constraint networks," *IEEE Transactions on Multimedia*, vol. 10, no. 3, pp. 523–537, 2008.

[11] E. Setton, J. Noh, and B. Girod, "Rate-distortion optimized video peer-to-peer multicast streaming," in *Proceedings of the ACM Workshop on Advances in Peer-to-Peer Multimedia Streaming*, pp. 39–48, Singapore, November 2005.

[12] H. Radha and Y. Chen, "Fine-granular-scalable video for packet networks," in *Proceedings of Packet Video Workshop (PV '99)*, New York, NY, USA, April 1999.

[13] W. Li, "Overview of fine granularity scalability in MPEG-4 video standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 3, pp. 301–317, 2001.

[14] H. Radha, M. van der Schaar, and Y. Chen, "The MPEG-4 fine-grained scalable video coding method for multimedia streaming over IP," *IEEE Transactions on Multimedia*, vol. 3, no. 1, pp. 53–68, 2001.

[15] K. Choi, K. Kim, and M. Pedram, "Energy-aware MPEG-4 FGS streaming," in *Proceedings of the 40th Annual Design Automation Conference (DAC '03)*, pp. 912–915, ACM, Anaheim, Calif, USA, 2003.

[16] J. Zhou, H.-R. Shao, C. Shen, and M.-T. Sun, "Multi-path transport of fgs video," in *Proceedings of Packet Video Workshop (PV '03)*, Nantes, France, April 2003.

[17] F. Wu, S. Li, and Y.-Q. Zhang, "A framework for efficient progressive fine granularity scalable video coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 3, pp. 332–344, 2001.

[18] Y.-K. Wang, "Method, device and system for effective fine granularity scalability (FGS) coding and decoding of video data," International Application N. PCT/IB2006/000631, International Filing Data 22.03.2006, 2006.

[19] "Intel(r) integrated performance primitives (intel(r) ipp)," http://www.intel.com/cd/ software / products / asmo-na / eng / 302910.htm.

[20] A. Lombardo and G. Schembra, "Performance evaluation of an adaptive-rate MPEG encoder matching intserv traffic constraints," *IEEE/ACM Transactions on Networking*, vol. 11, no. 1, pp. 47–65, 2003.

[21] G. M. Schuster and A. K. Katsaggelos, *Rate-Distortion Based Video Compression: Optimal Video Frame Compression and Object Boundary Encoding*, Springer, New York, NY, USA, 1997.

[22] C. F. Chang and J.-S. Wang, "A stable buffer control strategy for MPEG coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, no. 6, pp. 920–924, 1997.

[23] P. J. Brockwell and R. A. Davis, *Introduction to Time Series and Forecasting*, Springer, New York, NY, USA, 1996.

[24] B. Krithikaivasan, Y. Zeng, K. Deka, and D. Medhi, "Arch-based traffic forecasting and dynamic bandwidth provisioning for periodically measured nonstationary traffic," *IEEE/ACM Transactions on Networking*, vol. 15, no. 3, pp. 683–696, 2007.

[25] G. Platt and J. Y. Khan, "Adaptive jitter compensation on the downlink of future mobile multimedia communications systems," in *IEEE International Conference on Communications*, vol. 3, pp. 1887–1891, May 2003.

[26] M. Kalman, E. Steinbach, and B. Girod, "Adaptive media playout for low-delay video streaming over error-prone channels," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, no. 6, pp. 841–851, 2004.

[27] O. Hashida, Y. Takahashi, and S. Shimogawa, "Switched batch bernoulli process (SBBP) and the discrete-timeSBBP/G/1 queue with application to statistical multiplexer performance," *IEEE Journal on Selected Areas in Communications*, vol. 9, no. 3, pp. 394–401, 1991.

[28] "Bandwidth generator tool," http://www.diit.unict.it/arti/ Tools/BandwidthSim.zip.

[29] G. Hauske, R. Hofmeier, and T. Stockhammer, "Subjective image quality of low-rate and low-resolution video sequences," in *Proceedings of the 8th International Workshop on Mobile Multimedia Communications*, Munich, Germany, October 2003.