P2P Video Streaming Strategies based on Scalable Video Coding

F. A. López-Fuentes

Departamento de Tecnologías de la Información
Universidad Autónoma Metropolitana – Unidad Cuajimalpa
México, D. F., México
flopez@correo.cua.uam.mx

ABSTRACT
Video streaming over the Internet has gained significant popularity during the last years, and the academy and industry have realized a great research effort in this direction. In this scenario, scalable video coding (SVC) has emerged as an important video standard to provide more functionality to video transmission and storage applications. This paper proposes and evaluates two strategies based on scalable video coding for P2P video streaming services. In the first strategy, SVC is used to offer differentiated quality video to peers with heterogeneous capacities. The second strategy uses SVC to reach a homogeneous video quality between different videos from different sources. The obtained results show that our proposed strategies enable a system to improve its performance and introduce benefits such as differentiated quality of video for clients with heterogeneous capacities and variable network conditions.

Keywords: Codificación de video escalable, sistemas multi-fuente, redes peer-to-peer.

1. Introduction

Video streaming over the Internet has gained significant popularity during the last years. This fact has generated a dramatic technological and social revolution in video distribution and consumption. People download videos from several online video portals or from community networks to share their common interest. Thus, video playback from on-line video or news site has become part of the daily life of most Internet users. Streaming video applications have a strong impact in different scenarios such as videoconferencing, P2P content distribution, or event broadcast (IPTV). Different video streaming applications for live streaming, video on demand services or mobile television [18] have emerged as valuable tools to improve communication. Subsequently, many P2P media streaming systems such as ZigZag [1], CoolStreaming [2] or Mutualcast [3], have been developed. The P2P paradigm has become a promising solution for video streaming, because it offers characteristics which cannot be provided by the client-server model. P2P networks do not have a single point of failure, the upload capacity is shared among all peers, the bottlenecks are avoided, the contents can beshared by all participating peers, and they provide scalability. In addition, scalable Video Coding (SVC) can provide encoding of a high quality video bit stream, which contains some subset bit streams that can themselves be decoded with a complexity and reconstruction quality similar to that achieved using the existing H.264/AVC (Advanced Video Coding) design with the same quantity of data as in the subset bitstream [4], [5]. Using Scalable Video Coding, parts of a video stream can be removed and the resulting substream constitutes another valid video stream for some target decoder. In this way, we distribute video with different quality to the requesting peers with different bandwidth characteristics or when the network characteristics are time-varying.

In this work we present two strategies for video streaming in P2P networksbased on scalable video coding. The first strategy integrates scalable video coding (SVC) techniques into a meshed-P2P streaming system in order to offer differentiated quality video to peers with heterogeneous capacities. The second strategy uses SVC to adapt the flow rate from multiple sources in order to effectively use the available upload capacity from each source to deliver a homogeneous video quality for all streams. We consider as participating peers to the source peers, the requesting peers and the helper peers. We assume that all requesting peers need to receive all videos. The helper peers are not interested in receiving the videos and just contribute their resources during distribution. In this work, the helper peers are arbitrarily selected. The rest of this paper is organized as follows. We first present the scalable video coding modes used to encode video information into layers in Section 2. Video streaming strategies based on scalable video coding (SVC) are discussed in Section 3. The first method aims to provide differentiated video quality according to the capacity of each node, while the second method aims to provide homogeneous quality to different resources of the network. Finally, Section 4 describes the manner in which these strategies are implemented and evaluated. Conclusions are given in Section 5 where we also suggest further work.

2. Related Work

Scalable Video Coding (SVC) is a technique that encodes video information into layers. SVC is a well-established concept in the video topic (field) community, and incorporates the following scalability modes:

• Temporal scalability: subset bitstream represents lower temporal resolution. With subset bitstream a part of frames in one GOP (Group of pictures) can be decoded.

• Spatial scalability: lower subset bitstream can only playback a video with short frame size.

• Fidelity scalability SNR (Signal-to-noise ratio): the base layer bitstream can only playback a video of very low quality. The more enhancement layers the client receives, the better the quality of the playback video.

• Combined scalability: is a combination of two or more modalities above.

Scalable Video Coding (SVC) starts with the base layer, which contains the lowest level of spatial, temporal and quality perspective detail. Additional layers called enhancement layers can increase the quality of the video stream. An enhancement layer is called a spatial enhancement layer when the spatial resolution changes with respect to its reference layer, and it is called a fidelity enhancement layer when the spatial resolution is identical to that of its reference layer [6]. SVC introduces new video coding techniques which provides the following features, reduced bit-rate, reduced spatial temporal resolution and coding efficiency, comparable to non-scalable video systems. SVC extends the H.264/AVC (Advanced Video Coding) standard to enable video transmission to heterogeneous clients. If there are not a prior knowledge of the client capabilities, resources and variable network conditions, SVC uses the available resources.

Several P2P video streaming systems using SVC are reported in the literature. The authors in [12] propose an adaptive video streaming mechanism, which is based on SVC. In this work, scalable video coding is used to guarantee smooth delivery of video to peers in a P2P media streaming system. To reach this goal the authors perform their test in an unstructured P2P network. Their results show that video throughput is improved and the quality of the received videos is enhanced.

A peer-to-peer video streaming system to stream video with SVC is described in [13]. In this work the authors realize a theoretical analysis in order to quantify the expected distortion of H.264/AVC simulcast and SVC. In this model, the authors assume that the uplink capacity is the same as the downlink capacity for all nodes, the network is error-free and the client's throughput is known by the source. The peak signal-to-noise ratio (PSNR) is used as a quality metric. This model was tested on top of a real-time protocol called Stanford Peer-to-Peer Multicast (SPPM) [14]. Especially in cases of network congestion, results report that SVC improves the performance with respect to single layer coding. This is because if the video is not completely received, SVC plays at least the most important video layers. PSNR (peak signal-to-noise ratio) is also improved and the frame loss rate is significantly decreased by using SVC.

In [15] the authors show how the visual quality is influenced by combining scalability parameters in SVC schemes in order to provide guideline for a bit-rate adaptation strategy of SVC. In this work the authors use high definition (HD) contents with high bit-rates and high frame rates as test sequences.

The scalable video coding sequences are produced by two different scalable video coding. The results are analyzed with respect to several factors that affect the visual quality such as content type and scalable video coding. This work aims to improve the visual quality in the on-line video content distribution systems.

A peer-to-peer streaming application for distribution of live and on-demand video file called Pulsar is proposed in [16]. In this architecture, the peers are connected within the network via a special routing protocol in order to offer continuous stream of data to all members. In this case, SVC is used to deal with the problem of heterogeneity of resources and peers in the network. The results revels that SVC realizes a more efficient use of the restricted bandwidth than H.264/AVC. A similar problem to support video streams with different qualities in a P2P system is studied in [17]. To this end, the authors propose to create a different video file for each quality level and different overlays. This SVC strategy is implemented over a mesh-based streaming architecture which can be used to both live streaming and video-on-demand. Mainly, SVC is used in this architecture to allow the adaptation of the following resources in the receiving peers: heterogeneous screen size and resolutions, different connections with variable capacities, and heterogeneous processing capabilities.

The systems previously described show that scalable video coding (SVC) has the potential to enhance the video streaming functionality of peer-to-peer networks. SVC is often used to gain resilience against transmission failure, guarantee smooth delivery of video content and to offer adaptive bit-rate.

In this paper we propose two strategies to guarantee scalable video coding in P2P video streaming networks. The first method aims to provide differentiated video quality according to the capacity of each node. The idea is to encode the video into one base layer (BL) and multiple enhancement layers (ELs). The BL is delivered with highest priority and ELs are delivered with best effort. The second method aims to provide homogeneous quality to different resources of the network. The idea is to encode the videos with different rates, and source peers collaborate to ensure that the requesting peers will receive the videos with the same quality. The JSVM (Joint Scalable Video Model) software [7] is used as the codec to provide SNR scalable bit streams. Our proposed strategies also can be useful in scenario where the clients have different display resolutions, systems with different intermediate storage resources, and networks with varying bandwidths and loss rates. Main contribution of this paper is implementation of scalable video coding in a P2P infrastructure same to Mutualcast [3], which does not consider video issues. Second contribution is implementation of SVC in a mesh P2P infrastructure, which is fully collaborative [10].

3. Video Streaming Strategies based on Scalable Video Coding

In this section we introduce the main contribution of this work, which are two strategies for video streaming on peer-to-peer networks supported by scalable video coding techniques. The first strategy is focused to offering differentiated video quality to peers with heterogeneous capacities, while the second strategy is focused to distributing homogeneous video quality for all stream generated from multiple sources.The upload and download capacity provided by ISPs(Internet Services Providers) are not the real IP network capacity. However, in this work a study about ISP interconnection and its impact on consumer Internet performance are not considered.

3.1 Differentiated Video Quality to Peers with Heterogeneous Capacities

The general scenario of our first strategy is presented in Figure 1[11]. In this case scenario, the communication structure contains one source node and four client nodes (three requesting peers R_1, R_2, R_3 and one helper peer H) which are able to receive and forward the streams. In this scheme, we use UDP (User Data Protocol) links to send the video blocks from the source to each requesting peer, and TCP (Transport Control Protocol) links to receive the feedback messages from the requesting peers. The feedback information required is related to initialization ofthe network structure and the flow control of the UDPdata transfer. Similar to Mutualcast [3], each peer receives a single block from the source for redistribution. As is shown in Figure 1, block X_1,

X_2, and X_3, have been sent to the peers R_1, R_2 and R_3 respectively. Block X_4 is sent to the helper peer H for redistribution to the requesting peers R_1, R_2 and R_3. The helper peer H is not interested in receiving the video and just contributes with its upload capacity during distribution. If the source site has abundant upload capacity, it sends one copy of block X5 directly to each requesting peer R_i. Our approach assumes that the upload capacity of the peers is constrained, while the download capacity is infinite and all participating peers are present during a streaming session.

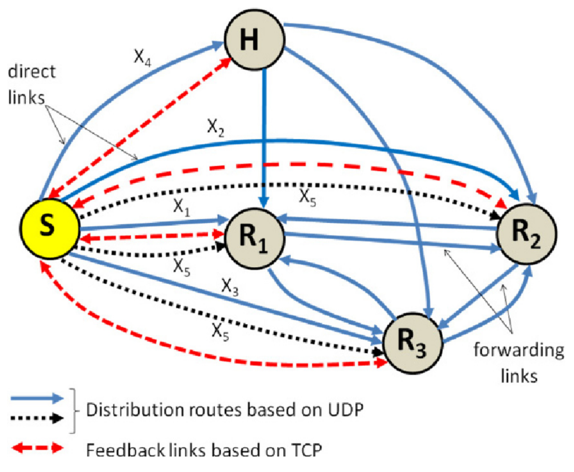


Figure 1. A meshed-P2P framework
for multicast applications.

SVC encodes the video into one base layer (BL) and one or more enhancement layers (EL). The BL provides a basic level of quality, while the enhancement layers refine the base layer quality. The base layer can be decoded independently of the enhancement layers. However, enhancement layers alone are not useful. SVC provides flexibility, because when the network capacity is not sufficient, only a subset of the layers is distributed to the requesting peers and they can still display the video, although at a reduced quality. When the network capacity increases, the requesting peers can receive more layers and thus, the reconstruction quality can be improved. Because peers with higher bandwidth request more layers and achieve a higher reconstruction quality than the requesting peers with less bandwidth.

To adapt the scalable video coding procedure to our distribution scheme (see Figure 1), we assume that the base layer (BL) or enhancement layer (EL) can be represented by block X_i. The source node sends the BL to all requesting peers. However, notice that not all of them receive the EL. This depends on the upload capacity of each requesting peer. The BL is directly delivered by the source to each peer, because of the BL highest priority. The priorities of the enhancement layers can be various (e.g., EL1 has a higher priority than EL2). If the source has abundant upload capacity after the BL has been sent, then the source sends the EL according to the respective layer priority. If the upload capacity of all clients is large enough,it is used by the source by sending different EL to each requesting peer for redistribution. Following this principle, one EL forwarded from another peer is considered with a higher priority than one EL sent directly from the source. In our approach, the source uses a swapping strategy to select a competent peer for each EL that has to be redistributed. However, if the upload capacity of a selected peer is insufficient to forward a copy of a given EL to all peers, the source sends it to the rest of the requesting peers. Peers with similar upload capacity receive from the source the same number of ELs for the redistribution. Peers with heterogeneous upload capacity receive different numbers of ELs. The network dynamics causes the dynamic changes in the transfer rules for the enhancement layers applying.Pseudocode for this strategy is shown in Figure 2.

```
begin
    $X_i$ = $BL$ or $EL$        /* Block $X_i$ contains Base Layer ($BL$)
                                  or Enhancement Layer ($EL$) */
    $i \neq j$
    video is encoded in a $BL$ and different $EL_j$

    for each requesting peer $R_1$ to $R_n$, source $S$ sends $BL$
    endfor
    if there are a helper peer $H$, then
        source $S$ sends a $EL_j$ to redistribution and
            peer $H$ redistributes $EL_j$ to peers $p_i$
    endif
    if upload capacity of a requesting peer $p_i$ is enough then
        source $S$ sends $EL_{j+1}$ to $p_i$ and
            peer $p_i$ redistributes $EL_j$ to peers $p_{i+1}$
    endif
    if source $S$ has abundant upload capacity then
            source $S$ sends $EL_{j+2}$ to each requesting peer $p_i$
    endif
end
```

Figure 2. Pseudocode for differentiated video quality to peers with heterogeneous capacities.

3.2 Homogeneous Video Quality for all Streams from Multiples Source

The second strategy uses scalable video coding to effectively exploit the available upload capacity from each source to deliver a homogeneous video quality for all streams in a multi-source structure [10]. We use the PSNR as a measure of video quality. The framework used by this strategy is illustrated in Figure 3 for two source peers, two requesting peers and one helper peer.In this example, the peers S_1 and S_2 are the sources, which contain the video sequences X and Y to be distributed. Peers R_1, R_2 are the requesting peers, and peer H_1 is a helper peer.The peer H_1 does not request the videos, but contributes its upload capacity to help distributing the videos to the other peers. Here, we can see that all the peers are in fact receivers and senders at the same time, as it is for instance the case in a multipoint video conferencing scenario. Each source splits the original content into small blocks and one unique peer is selected to distribute a block to the rest of the peers. The requesting peers and helper peers forward the received video block to the other requesting peers and the other source peers. For our example, the source S_1 divides the video X into the blocks X_1 to X_4, while the source S_2 divides the video Y into the blocks Y_1 to Y_4.

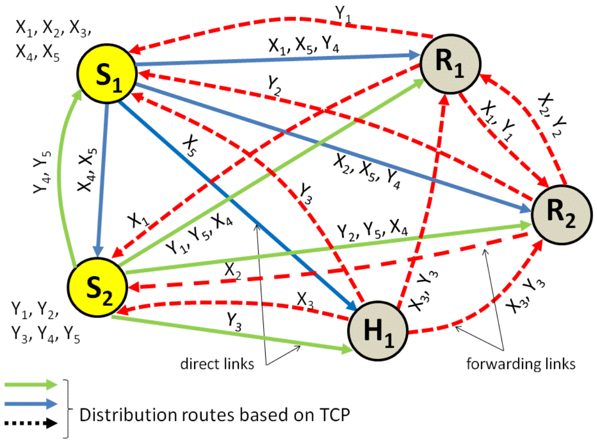


Figure 3. A meshed-P2P framework for multicast applications.

Because our approach is based on collaboration among sources, each source distributes its own video while additionally forwarding the block of video received from the other source to the rest of the requesting peers. At the same time, each requesting peer forwards the blocks directly received from a source to the rest of the participating peers. Thus, the blocks (X_1, Y_1), and (X_2, Y_2) are assigned to the requesting peers R_1 and R_2, respectively, while the block (X_3, Y_3) is assigned to the helper peer H_1.The block X_4 is assigned to the source peer S_2 and block Y_4 is assigned to the source peer S_1 for distribution. Peers with different upload capacity distribute a different amount of content. The block size assigned to each requesting peer is proportional to its upload capacity. When the source peers have abundant upload resources, each source additionally sends one block directly to the video receiving peers. To illustrate such a case, Figure 3 shows that source S_1 directly sends block X_5 to each video requesting peer and source S_2 directly sends block Y_5. Thus, source S1 sends one block to each participating peer for redistribution, one block in parallel to all requesting peers, and forwards one block of the video Y received from the source S_2 to each requesting peer R_i. The source S_2 behaves similar as source S_1, but in a complementary way. It sends the video Y and forwards the video block X_4. Each requesting peer forwards the blocks received from the sources S_1 and S_2 to the other requesting peers and the other sources, e.g., peer R_1 receives the blocks X_1 and X_5 from source S_1 and the block Y_1 and Y_5 from the source S_2. After this, peer R_1 forwards the block X_1 and Y_1 to the rest of the participating peers except to the source where the block was originally generated and the helper peer H_1. The blocks X_3 and Y_3 are sent by the sources S_1 and S_2, respectively to the helper peer H_1, which forwards the blocks to all participating peers except to the source where the block was originally generated.

In this strategy, the sources jointly decide the rate allocation for all participating peers, but additionally enforce the same video quality for all video streams by using scalable video coding techniques. We assume that all participating peers are fully connected and all of themneed to receive all videos. We describe a scenario with two sources (S_1 and S_2) distributing two different video sequences with same PSNR. Initially, the rates R_i generated by both sources are assumed to be identical. The upload requirement for source S_1 is then given by

$$C_{S1} = \sum_{i=1}^{N1} B_{Ri}^1 + B_{S2}^1 + (N_1+1)B_D^1 + N_1 B_{S1}^2 \qquad (1)$$

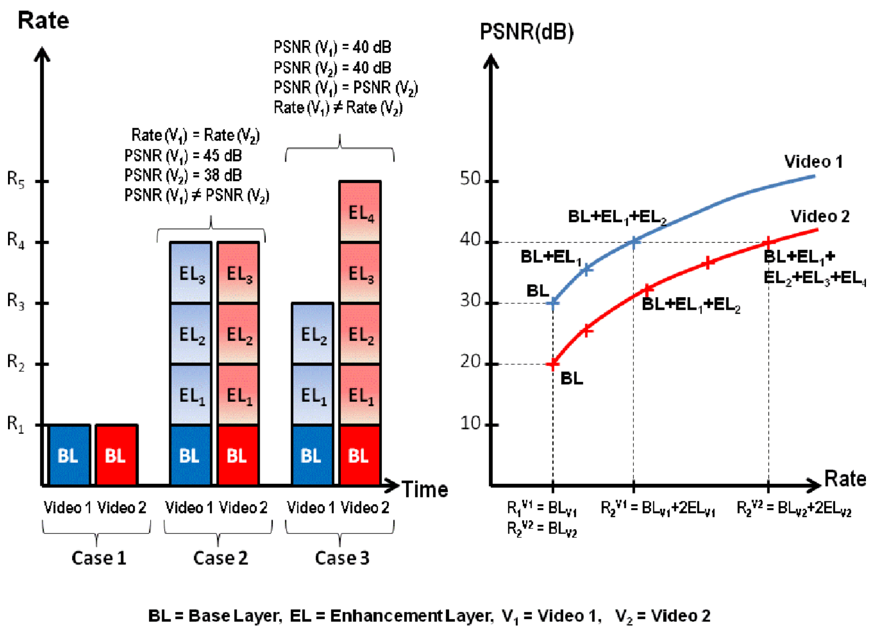
The first term in (1) represents the amount of data being sent from the source S$_1$ to the requesting peers for redistribution. B_{S2}^1 is the amount of data being sent from source S$_1$ to source S$_2$ for redistribution. $(N_1+1)B_D^1$ stands for the upload capacity needed to directly send a video block from source S$_1$ to the N$_1$ requesting peers and the other source S$_2$. $N_1 B_{S1}^2$ is the upload capacity that is needed at source S$_1$ to redistribute a block received from S$_2$. Similarly, the exhaustion of the upload capacity of source S$_2$ is achieved for

$$C_{S2} = \sum_{i=1}^{N1} B_{Ri}^2 + B_{S1}^2 + (N_1+1)B_D^2 + N_1 B_{S2}^1 \qquad (2)$$

The upload contribution from the requesting peers is given by

$$B_R = \sum_{i=1}^{N1} B_{Ri}^1 + \sum_{i=1}^{N1} B_{Ri}^2 = \sum_{i=1}^{N1}(B_{Ri}^1 + B_{Ri}^2) = \sum_{i=1}^{N1}\frac{C_{Ri}}{N1} = \frac{1}{N_1}\sum_{i=1}^{N1} C_{Ri} = \overline{C_R} \qquad (3)$$

The same quality for all videos is obtained if both videos have the same Rate-Distortion function. However, when the same rate for all video sequences is not enough to obtain a similar video quality among them, we need to enforce a same PSNR. The PSNR enforcement is possible, when the sources have abundant upload capacity. To this end, the broadcast links in each source are manipulated, and the rate of the sequence with the largest rate is reduced. We effectively use the available upload capacity from each source to deliver a homogeneous video quality for all streams. To this end, each source schedules the distribution according to the ratio of the video bit rates based on scalable video coding techniques. We assume that the quality requirements are known. Then, a number of layers to reach this quality level are determined for each video in each source using scalable video coding. The number of layers and the coding rate for two different video sequences is determined (see Figure 4).



(a) (b)

Figure 4. Enforcement of the same video quality for two different videos using scalable video coding. a). Redistribution of layers, b). PSNR comparison.

In Figure 4a), the sources send the base layer of their videos in Case 1. In Case 2, both sources send three enhancement layers of their videos, and the rate R_4 for both sequences is the same. The example assumes that video 1 and video 2 are different and they have been encoded with different bit rates. Thus, using the rate R_4 for both videos, a PSNR of 45 dB and 38 dB for video 1 and video 2, are obtained respectively. In order to enforce the same PSNR for both sequences scalable video coding is used in Case 3. Then, the source 1 sends two enhancement layer of video 1, while the source 2 sends four enhancement layers of video 2. Figure 4b) shows how both videos sequences can reach the same PSNR using different number of enhancement layers and different rate.

Once the number of required layers and the coding rate are known in each source and before starting the distribution, the sources exchange the coding rate of their videos. Each source computes a local distribution ratio k using these coding rates. The case considers M=2. This ratio is used in each source to determine the number of required packets for each video. In an ideal situation it is desirable that the throughput equals the playback bit rate of the videos in order to obtain a short initial waiting time and a minimal size of buffers. In contrast, if the upload capacity of the sources is not enough to satisfy the requested throughput, theinitial time and the buffer usage is increased. Additionally, when different videos are distributed from different sources, the sources need to synchronize the playback of the videos and adapt their upload allocation for the distribution, so that all videos can be received with adaptive throughput and have similar initial waiting time or video quality. The sources use the distribution ratio to adapt the distribution throughput of streams and each source can schedule the number of packets to distribute from itself and the number of packet to distribute from the other sources. The number of distributed packets for each video is proportional to its coding rate.

4. Implementation and Evaluation

We have implemented different prototypes of our proposed strategies in order to evaluate its performance in real world scenarios. The implementations run on Linux and consist of different programs written in the C/C++ language. The implementation includes a server module run by the source peers and a receiver module run by each requesting peer. Both modules have been enabled with a sender/receiver mode. All links among the participating peers are established using TCP connections. Reliable data delivery, flow-control and handling of node leave events are automatically supported by the TCP protocol.Packet loss is also supported by the TCP protocol via retransmission, which is not an ideal solution for systems with nodes distant together.

Each requesting peer runs a receiver module which receives the video blocks from the sources for its playback and forwards these blocks to the rest of the requesting peers that need to receive this content. In this collaborative way, all participating peers are sending and receiving at the same time. The distribution of blocks among the participating peers is implemented using threads. Multithreaded applications have several advantages such as faster execution and parallelization. A multithreaded program operates faster on computer systems that have multiple CPUs, because the threads of the program lend themselves to concurrent execution. .In this work, threads are used in order to ensure distribution and storage with minimum delay.

Each source peer sets up the following threads:

• A sending thread to distribute own content to the requesting peers,

• A receiving thread to receive content from the sources and forward the data to the other peers,

• A second receiving thread to receive the forwarded blocks from the other peers,

• A storing thread to read the blocks from storage,

• A measuring thread to realize different measurements during the streaming session.

Each requesting peer has the following threads:

• A receiving thread to receive data from the sources and forward the data to the other peers,

• A second receiving thread to receive the forwarded blocks from the other peers,

• A storing thread to read the blocks from storage,

• A measuring thread to realize different measurements during the streaming session.

During a streaming session, the sending thread, receiving thread and measuring thread are concurrently running in each peer. To run the prototype, initially, the source starts listening on a predefined port and waits for the socket connection request. Then, each requesting peer establishes a connection with the source. The source copies the IP address and listening port of every requesting peer and sends this information to all requesting peers. After this, each peer starts a new thread, which listens and waits to establish the forward link for transferring content between two peers. All peers maintain a list of all peers in the multicast group. So the forwarding connection could be established very quickly. After the initialization of the forward link, all links are ready for content distribution. Then, the sources and the requesting peers begin the data transfer. After the blocks are received at each requesting peer or in each source, the blocks are forwarded to the rest of the participating peers. During the implementation, the block size is set to 1 kB, so each block can be sent using a single TCP/IP packet. The distribution of the different videos is concurrently realized in the source by blocking and unblocking its distribution and redistribution queues until the videos have been distributed. We have used the PlanetLab infrastructure [8] to evaluate the performance of our prototypes. Our experiments on PlanetLab were realized using a small group of PlanetLab nodes mainly localized in the United States of America:

node2.planetlab.uprr.pr, planet1.scs.stanford.edu, planetlab2.cs.uregon.edu,planetlab7.csail.mit.edu,righthand.eecs.harvard.edu, planet1.cs.ucsb.edu, planetlab3.cs.uregon.edu, and kupl1.ittc.ku.edu. We use the PSNR as the quality metric in our experiments.

4.1 Differentiated Video Quality to Peers with Heterogeneous Capacities

To evaluate our first strategy, we select the BUS sequence for a CIF (Common International Format) size as our test sequence. The BUS YUV sequence is encoded using the JSVM (Joint Scalable Video Model) software [9]. JSVM is a Scalable Video Coding codec used to encode and decode a video. JSVM can provide a bitstream that contains one or more subset bitstreams, which are derived by dropping packets from larger bitstream. A base layer is a subset bitstream, and can playback a video with very low frame per second (fps), small size of resolution or low quality (PSNR). One or more enhancement layers can be encoded in order to obtain refinement. Our implementation uses Joint Scalable Video Model (JSVM) as codec to provide SNR scalable bitstreams. We use Coarse Grain Scalability (CGS) in this work. JSVC is still under development and changes frequently. The reference software for the JSVM can be found on the Internet [7]. The SNR (Signal-to-Noise Ratio) scalable bitstreams are used as source video streams.

For our experiments, we encode 60 frames with 1 BL (Base Layer) and 3ELs (Enhancement Layers). The BL is encoded at 631.62 kbps, while EL1, EL2 and EL3 are encoded at 864.58 kbps, 1165.24 kbps and 1594.42 kbps, respectively [11]. Using these rates, BL, EL1, EL2 and EL3 achieve a video quality (PSNR) of 31.13 dB, 32.07 dB, 32.97 dB and 34.0 dB, respectively. Figure 5 shows that the reconstructions quality for the BUS sequence can be different in each requesting peer. The video quality depends on the number of layers received by each peer.

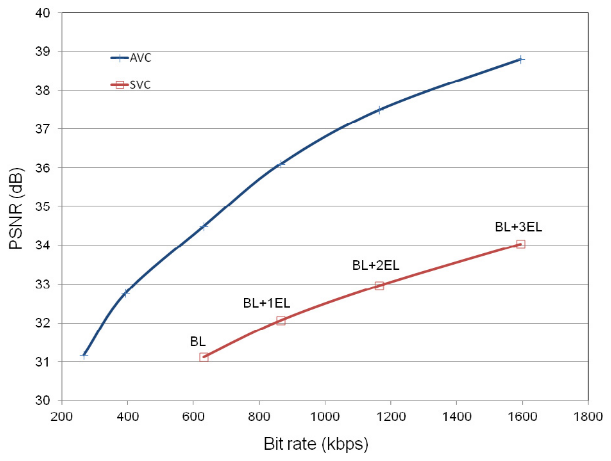


Figure 5. PSNR using SVC with base layer (BL) and different enhancement layers (ELs).

Figure 6 shows reconstructions of base layer and more enhancement layers of one video frame for the BUS sequence. The first picture shows the reconstruction of the video frame with only the base layer. The following pictures are reconstructed with the base layer and 1, 2 or 3 enhancement layers. In this case, we can see that video quality is acceptable if only the base layer is reconstructed in the end-peers.

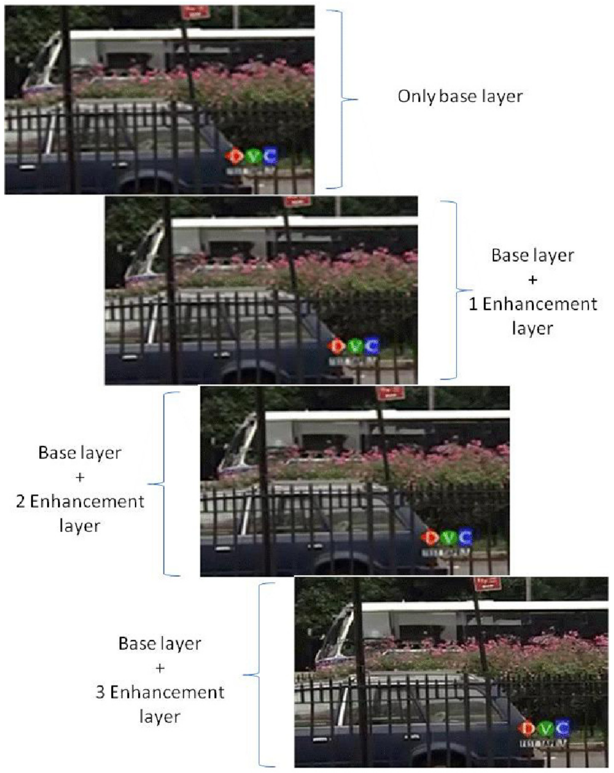

Only base layer
Base layer + 1 Enhancement layer
Base layer + 2 Enhancement layer
Base layer + 3 Enhancement layer

Figure 6. Sequence of reconstruction of a video frame using SVC.

In contrast, using refinement, we can see that the reconstruction quality gets better (e.g. picture with base layer and three enhancement layers). However, all layers have identical picture spatial resolution. This scenario provides enough flexibility when the network capacity is not enough to deliver a video with high reconstruction quality. Then, the source can only distribute the base layer to the client, and itcan still represent the video fluently. When the network capacity is better, the client can receive additional layer, and the base layer reconstruction quality can be improved.

The adaptability of the links to the bandwidth variations is realized by using SVC. To this end, a flow control mechanism is responsible to adjust the transfer plan in the source. Thus, the source node can send a specific enhancement layer (EL) to a specific requesting peer directly. Under this scenario, the throughput on the direct link is increased, while the throughput of the forward link is reduced. When the forward link is again available, the transfer plan is readjusted, and the forward link throughput is increased, while the direct link throughput is reduced. A peer achieves a better video quality by receiving an EL from the source than if it is not received from another peer R.

4.2 Homogeneous Video Quality for all Streams from Multiples Sources

Our second strategy is evaluated in terms of throughput, PSNR, and delay for all video streams. Our experiments are based on a joint rate allocation decision and we concurrently control the bit rate for both sources. The videos sequences used to evaluate this strategy are: Mother and Daughter (M&D), and Foreman. The short Foreman sequence is concatenated to a long test sequence with 3000 frames. The same is done with the M&D sequence. Both video streams are encoded with the JSVM software [9] with the same video quality PSNR around 42 dB, but using different encoding rate and different number of layers for each sequence. To achieve this video quality the Foreman sequence needs a bit rate of 1600 kbps, which is obtained by using one base layer and two enhancement layers. The Mother and Daughter sequence is encoded at 230 kbps using one base layer and one enhancement layer. Both videos have the same duration (60 seconds), but the size of the Foreman and Mother and Daughter files are 10 MB and 1.5 MB, respectively.

For these experiments, we considered that all the participating nodes are fully interconnected, including the sources nodes. The adaptive multicast approach is evaluated in terms of delay for the two different videos. To this end, the two test videos with the same PSNR are allocated as video 1 and video 2 at the sources S_1 and S_2, respectively. After this, the initial rates in the prototype multi-source multicast are fixed to 230 kbps and 1600 kbps for sources S1 and S_2, respectively. M&D video is located at source S1, while Foreman video is located at source S2. Then, both sources deliver their video files to all participating peers, including the sources.

First, we evaluate our second strategy using a local area network (LAN). Figure 7 shows the distribution throughput on source peers. We can see that without our strategy of distribution control at the source peers, the distribution throughput of Foreman video on source peer S_2 is larger than the distribution throughput of M&D video on source peer S1. The M&D video is distributed quickly. Contrary, we can see that, with scheduling the distribution, each source peer can regulate the distribution throughput of its own videoand the other video in proportion.

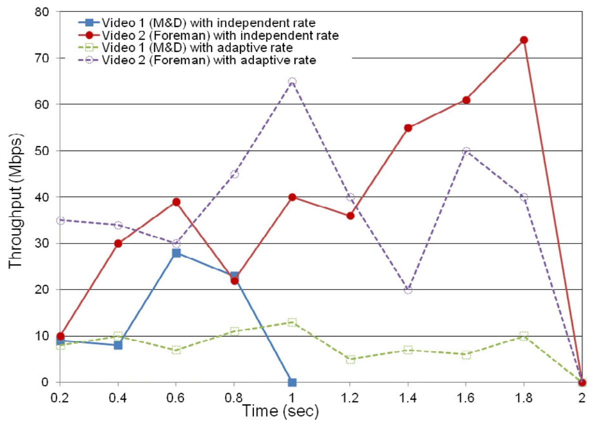


Figure 7. Distribution throughput on a source for two different video streams over a LAN.

In addition, we can see in Figure 7 that curves of distribution throughput of both video streams are almost the same.

Figure 8 displays the receiving throughput of M&D and Foreman videos on a requesting peer. We can easily see that without regulating the distribution throughput at the source peers, M&D video is received very quickly on the requesting peer. Most probably, the initial delay of playback for this stream is shorter. In contrast, for Foreman video, because it is slowly received, its initial delay may be longer than D&Mvideo. In order to be able to synchronously play out these two streams the buffer demands on the nodes is very high. However, if both source peers schedule distribution, the delay of receiving both video streams on a requesting peer takes almost the same time. Therefore, the playback during receiving can be synchronously with low buffering demands. As the Figure 8shown, the curves of receiving throughput of these two streams are almost the same.

Figures 9 and 10 show the resulting throughput for both videos obtained from our experiments on PlanetLab. Video with independent rate means that the source distributes the videos with different PSNR. Video with adaptive rate means that the source peer integrates scalable video coding to enforce the same video quality (PSNR) for both videos.

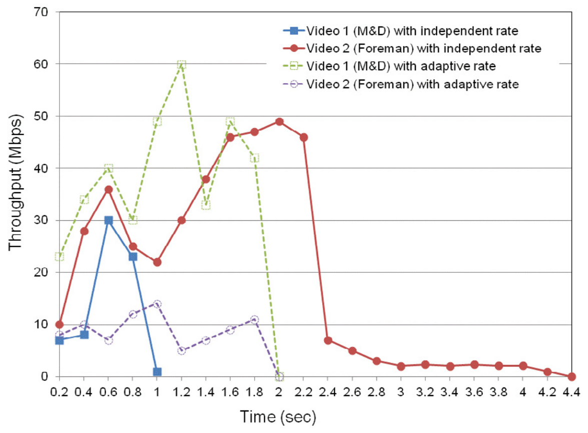


Figure 8. Receiving throughput on a requesting peer for two different video streams over a LAN.

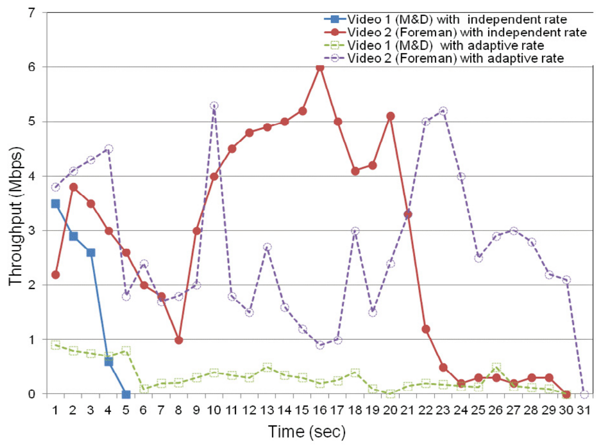


Figure 9. Distribution throughput on a source peer for two different video streams over PlanetLab.

Similar to our experiments realized in our local area network, Figure 9 shows that the delay to distribute both videos without using scalable video coding (SVC) is different, while through our SVC strategy, both videos reach a similar delivery time, because the distribution throughput adapts their bit rate and both videos can be played back synchronously without high buffering demands. In addition, Figure 10 shows that when the approach based on SVC is used, the requesting peer receives both videos in a more similar duration than when it is not used.

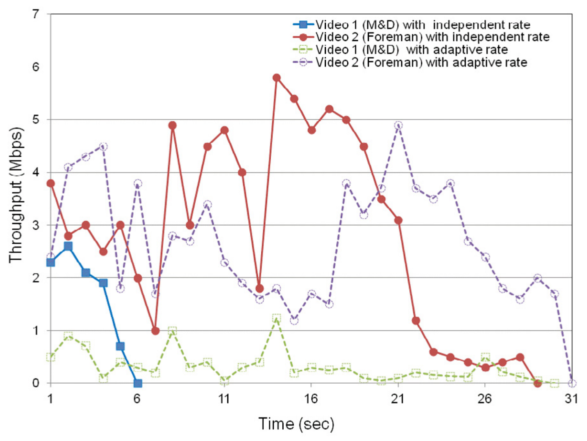


Figure 10. Receiving throughput on a requesting peer for two different video streams over PlanetLab.

SVC allows an adaptive rate control to integrate more collaboration between the sources in order to achieve a similar PSNR quality for all video sequences. Thus, the average throughput can be used to recalculate the PSNR with JSVM software, obtaining PSNR values of 43.4 dB and 43.9 dB for Foreman and Mother and Daughter, respectively.

5. Conclusions

We have presented and evaluated two strategies to P2P video streaming transmission based on scalable video coding (SVC). Our first strategy helps to distribute differentiated quality video to peers with heterogeneous capacity. It allocates different number of layers to peers with different capacities or different levels of redundancy to thebase layer. Using SVC, we can not only ensure subscribers that we will provide the minimum QoS, but we can also increase this level substantially depending on the available bandwidth of the links.

Our second strategy helps to reaching similar video quality for all streams from multiples sources. To this goal, we have formulated this strategy as an optimization problem with an objective function to maximize the aggregate video quality. Both proposed strategies were evaluated in the PlanetLab infrastructure using four nodes. Our systems with scalable video coding (SVC) have demonstrated its effectiveness compared to systems without SVC. Our proposed strategies could be ideally used for P2P video streaming scenarios with few participants such as video-conference or surveillance systems.

As future work, some important properties of P2P networks, such as scalability and churn can be incorporated in our proposed strategies. We believe that both strategies can be considered for very large P2P video streaming systems. Currently, our solution for multiples sourcesis implemented using TCP protocol, which introduce some limitations associated to the packet loss and retransmission. A UDP/TCP hybrid mechanism with a protection scheme to alleviate the packet loss problem could be considered for this case.

References

[1] D. A. Tran et al., "A Peer-to-Peer Architecture for Media Streaming," IEEE Journal on Selected Areas in Communication, Special Issue on Advances in Overlay Networks, vol. 22, no.1, pp.121-133, 2004.

[2] X. Zhang et al., "CoolStreaming/DONet: A Data-driven Overlay Network for Efficient Live Media Streaming," in 24th IEEE INFOCOM, Miami, FL, USA, 2005, Vol.3, pp. 2102-2111.

[3] J. Li et al., "Mutualcast: An Efficient Mechanism for One-To-Many Content Distribution," in ACM SIGCOMM ASIA Workshop, Beijing, China, 2005.

[4] H. Schwarz et al., "Overview of the Scalable Video Coding Extension of the H.264/AVC Standard," IEEE Transactions on Circuits and Systems for Video Technology, vol. 17, no. 9, pp. 1103-1120, 2007.

[5] M. Wien et al., "Performance Analysis of SVC," IEEE Transactions on Circuits and Systems for Video Technology, vol. 17, no.9, pp. 1194-1203, 2007.

[6] H. Schwarz and M. Wien,"The Scalable Video Coding Extension of the H.264/AVC Standard," IEEE Signal Processing Magazine, vol. 25, no.2, pp. 135-141, 2008.

[7] J. Reichelet al., "Joint Scalable Video Model (JSVM) 11," Doc. JVT-X202, Joint Video Team, Video Coding Experts Group, 2007.

[8] L. Peterson et al.,"A Blueprint for Introducing Disruptive Technology into the Internet,"ACM SIGCOMM Computer Communication Review," vol. 3, no. 1, pp. 59-54, 2003.

[9] Heinrich-Hertz-Institute, Joint Video Team (JVT); JSVM software manual, version 9.12.2. Available from: http://ip.hhi.de/imagecom/G1/savce/downloads/SVCReferenceSoftware.htm.

[10] F. A. López-Fuentes and E. Steinbach, "Adaptive Multi-source Video Multicast," in IEEE International Conference on Multimedia & Expo, Hannover, Germany, 2008, pp. 457-460.

[11] F. A. López-Fuentes,"Adaptive Mechanism for P2P Video Streaming using SVC and MDC,"in International Conference on Complex, Intelligent, and Software Intensive Systems,Krakow, Poland, 2010, pp. 457-462.

[12] M. Mushtaq and T. Ahmed, "Smooth Video Delivery for SVC Based Media Streaming Over P2PNetworks," in 5thIEEE Consumer Communications and Networking Conference, Las Vegas Nevada, USA,2008, pp. 447-451.

[13] P. Baccichet et al., "Low-delay peer-to-peer streaming using scalable video coding," in Packet Video Workshop, Lausanne, Switzerland, 2007, pp. 173-181.

[14] E. Setton et al., "Peer-to-Peer Live Multicast: A Video Perspective," in IEEE INFOCOM, vol. 96, no.1, Phoenix, AZ, USA, 2008, pp. 25-38.

[15] J. Lee et al., "Subjective Evaluation of Scalable Video Coding for Content Distribution,"in ACM International Conference on Multimedia, Florence, Italy,2010, pp. 65-72.

[16] J. Rieckh, "Scalable Video for Peer-to-Peer Streaming," Master's Thesis, Technical University of Vienna, Summer 2008.

[17] O. Abboud, "Quality Adaptive Peer-to-Peer Streaming Using Scalable Video Coding,"in 12thIFIP/IEEE International Conference on Management of Multimedia and Mobile Networks and Services, Venice, Italy, 2009, pp. 41-54.

[18] R. Tamayo-Fernández et al. "An architecture for design and planning of mobile television networks," Journal of Applied Research and Technology, vol.9, no.3, pp. 277-290., 2011.