Homogeneous Quality Video in Multi-Sources Systems

Francisco de Asís López-Fuentes[1]

Networking and Distributed Systems Group
[1]Departament of Information Technology
Universidad Autónoma Metropolitana - Cuajimalpa
Av. Vasco de Quiroga 4871, Santa Fe, Cuajimalpa de Morelos
053000 México, D. F.
{flopez}@correo.cua.uam.mx
Paper received on 11/20/13, Accepted on 01/19/14.

Abstract. Scalable video coding (SVC) has emerged as an important video standard to provide more functionality to video transmission and storage applications. This paper evaluates a strategy based on scalable video coding for peer-to-peer (P2P) video streaming services. This strategy uses SVC to reach a homogeneous video quality between different videos from different sources. Our scheme is implemented under the H.264/MPEG-4 SVC compression standard. Our evaluations were realized over a local area network (LAN). We evaluate our implementation in terms of overall throughput, delivery time (delay) and video quality. The obtained results show that our proposed solution strategies achieve good performance and introduce benefits in the peer-to-peer video streaming systems.

Keywords: scalable video coding, peer-to-peer systems, video streaming architectures.

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 form 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, Peer-to-peer (P2P) content distribution, or event broadcast (e.g. Internet Protocol Television (IPTV)). Different video streaming applications for live streaming or video on demand services 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. P2P paradigm has become a promising solution for video streaming, because it offers characteristics which cannot be provided by the client-server model. In contrast to client-server model, P2P networks do not have a single point of failure, the upload capacity is shared

Mireya García-Vázquez, Grigori Sidorov, Sunil Kumar (Eds.).
Advances in computing science, control and communications
Research in Computing Science 69, 2014, pp. 197- 206

among all peers, the bottlenecks are avoided, the contents can be shared by all participating peers, and they provide scalability. On the other hand, 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 (SVC), parts of a video stream can be removed and the resulting sub-stream constitutes another valid video stream for some target decoder. In this way, we distribute video with different quality to requesting peers with different bandwidth characteristics or when the network characteristics are time-varying.

In this paper we use scalable video coding 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. Our SVC scheme uses a peer-to-peer (P2P) structure described in previous work [9]. P2P structure used in this work is inspired from a scheme called Mutualcast [3], which is a scheme that reaches the maximum possible throughput. Our proposal scheme considers 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. We evaluate a particular case for a multi-source system using SVC for a reduced number of peers. Scalability of our scheme is limited, but it can be used in systems with a reduced number of participating nodes, such as video conferencing services or on-line games. The main contribution of this paper is to show how scalable video techniques can help to obtain a homogeneous video quality in LAN (local area network) applications with multi-sources and videos encoded with different bit-rates.

The rest of this paper is organized as follows. We first present the scalable video coding modes used to encode a video 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 Scalable Video Coding Background

Scalable video coding (SVC) is a technique that encodes the video into layers. SVC is well established concept in the video area, and incorporates the following scalability modes [5], [6] and [7]:

- *Temporal scalability:* subset bitstream represents lower temporal resolution. With subset bitstream a part of frames in one group of pictures (GOP) can be decoded.
- *Spatial scalability:* lower subset bitstream can only playback a video with lower frames size.
- *SNR (Signal-to-noise ratio)/fidelity scalability:* the base layer bit stream can only playback a video of very low quality. And the more enhancement layers the client receives, the better quality the video has.

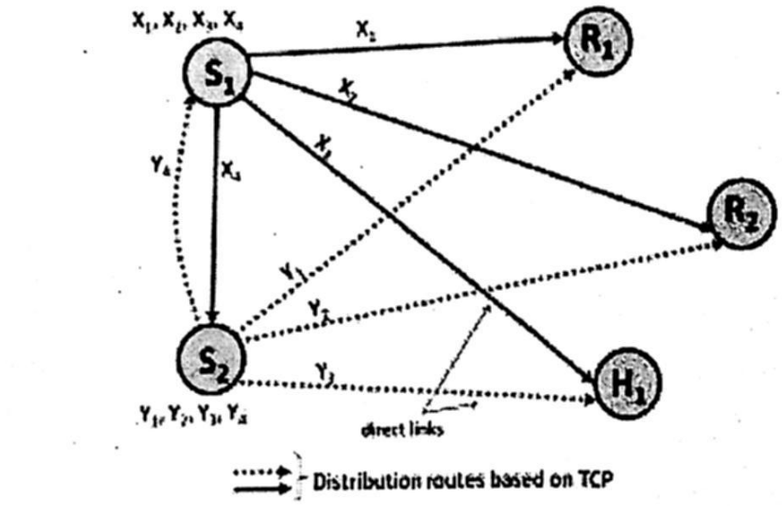
- *Combined scalability:* is a combination of all three or two modalities above. Scalable video coding (SVC) starts with the base layer, which contains the lowest level of spatial, temporal and quality perspective detail. Additional layers called enhancement layers can increase the quality of the video stream.

An enhancement layer is called a spatial enhancement layer when the spatial resolution changes with respect to its references 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 provide 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. To achieve it, SVC uses available system resources [6], in the case of the lack of a prior knowledge of the downstream client capabilities, resources, and variable network conditions. SVC provides a higher degree of error resiliency and video quality with no significant need for higher bandwidth. Also, scalable video coding can support a broad a range of devices and networks. Thus, scalable video coding is a suitable solution for adaptive content delivery to end-users having various preferences, terminal capabilities, and network conditions [10]. Scalable video coding has been used for different scenarios and applications and several works can be found in the literature [11], [12], [13]. In this paper we propose to apply scalable video coding in P2P video streaming networks. We aim to provide homogeneous quality video 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 (Signal-to-noise ratio) scalable bit streams. We do not compare our proposed model with other methods, because we implement our SVC strategy in a specific P2P multi-source infrastructure. Our proposed strategies also can be useful in scenario where networks with varying bandwidths and loss rates.

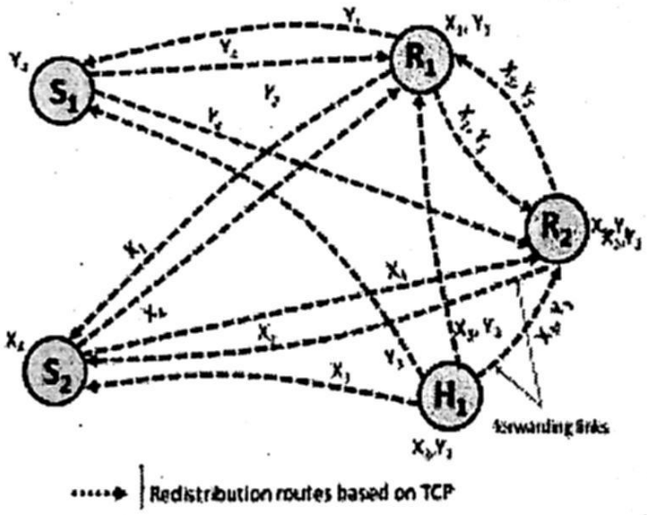
3 P2P Video Streaming Model

The proposed 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 [9]. We use the peak signal-to-noise ratio (PSNR) as a measure of video quality. This P2P scheme uses TCP, which has several limitations for video on wide area network (WAN), such as retransmission delay, packet loss, etc. However, for video applications in LAN (local area network), retransmission delay introduced by TCP (Transport Control Protocol) could be not significant [11]. In fact, retransmission has been used very successfully for non real-time data transmission. The framework used by this strategy is illustrated in Figure 1 for two source peers, two requesting peers and one helper peer. In this example, the peers S1 and S2 are the sources, which contain the video sequences X and Y to be distributed. Peers R1, R2 are the requesting peers, and peer H1 is a helper peer. The peer H1 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.



a)



b)

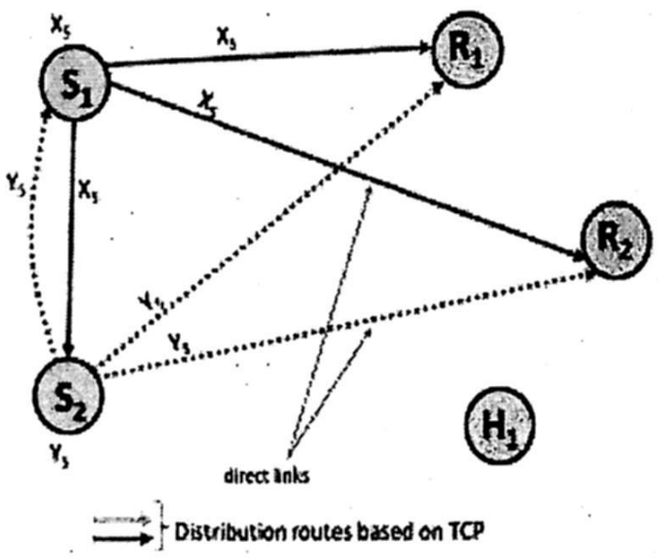


Fig. 1. Multi-source scheme based on a mesh topology. a) initial distribution, b) redistribution from all peers, c) one block is directly sent to each receiving peers

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 S1 divides the video X into the blocks X1 to X4, while the source S2 divides the video Y into the blocks Y1 to Y4 (Figure 1a). 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 (X1, Y1), and (X2, Y2) are assigned to the requesting peers R1 and R2, respectively, while the block (X3, Y3) is assigned to the helper peer H1, the block X4 is assigned to the source peer S2 and block Y4 is assigned to the source peer S1 for distribution. This scenario is shown in Figure 1a. 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. 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 S2 to each requesting peer Ri. The source S2 behaves similar as source S1, but in a complementary way. It sends the video Y and forwards the video block X4. Figure 1b shows a redistribution scenario. In this case, each requesting peer forwards the blocks received from the sources S1 and S2 to the other requesting peers and the other sources. Peer R1 receives the blocks X1 from source S1 and the block Y1 from the source S2. After this, peer R1 forwards the block X1 and Y1 to the rest of the participating peers except to the source where the block was originally generated and the helper peer H1. The blocks X3 and Y3 are sent by the sources S1 and S2, respectively to the helper peer H1, which forwards the blocks to all participating peers except to the source where the block was originally generated. When the source peers have abundant upload resources, each source additionally sends one block directly to the video receiving peers. Figure 1c shows this case. Here, source S1 directly sends block X5 to each video requesting peer and source S2 directly sends block Y5. 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 them need to receive all videos. Our scenario is described for two sources (S1 and S2) distributing two different video sequences with same PSNR (Peak Signal-to-Noise Ratio) to all participating peers.

The same quality for all videos is obtained if both videos have the same RD-function. However, when the same rate for all video sequences is not enough to obtain a similar video quality among them, a same PSNR need to be enforced. 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. Determining the number of layers and the coding rate for two different video sequences is illustrated in Figure 2.

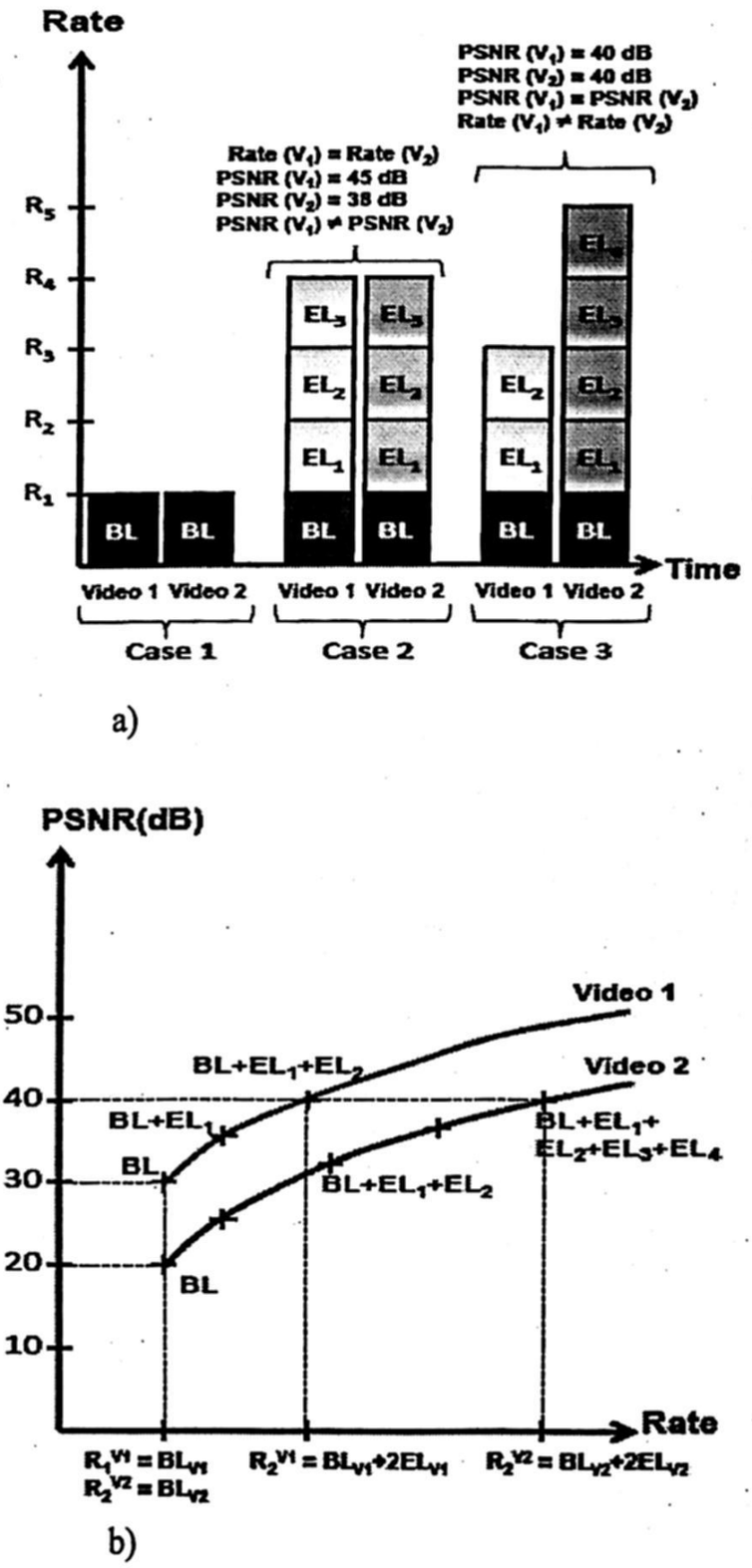


Fig. 2. Enforcement of the same video quality for two different videos using scalable video coding. a). Redistribution of layers, b). PSNR comparison

In Figure 2a), 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 R4 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 R4 for both videos,

a PSNR (Peak Signal-to-Noise Ratio) 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 2b) shows how both videos sequences can reach the same PSNR using different number of enhancement layers and different rate. In this paper, the PSNR's measurements are obtained experimentally through the JSVM (Joint Scalable Video Model) software [8].

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 by using these coding rates. 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 is the same as 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, the initial 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 Evaluation

In order to evaluate the performance of our proposed solution, we implemented a prototype of this in Linux (Fedora distribution). Our implementation consists of different programs written in the C/C++ language. This 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 (Transport Control Protocol) connections. Reliable data delivery, flow-control and handling of node leave events are automatically supported by the TCP protocol. 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. We have used a LAN (Local Area Network) infrastructure to evaluate the performance of our scheme implementation. Our proposed scheme is evaluated in terms of throughput, PSNR, and delay for all video streams. We use the peak-signal-to-noise-ratio (PSNR) as the quality metric. 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. Foreman sequences shows a man talking in a construction site, while M&D shows two persons in the foreground. A person is talking, while the other persona is fixed. 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 [8] 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, while Mother and Daughter sequence is encoded at 230 kbps. Our Foreman sequence uses one base layer and two enhancement layers, while M&D sequence uses the base layer and one enhancement layer. Both videos have the same duration (60 seconds), but different size. Foreman file is 10 MB, while M& D file is 1.5 MB.

In these experiments, we considered that all the participating nodes are fully interconnected, including the sources nodes. We compare the time required by both videos to be received in a requesting peer. To this end, two test videos with the same PSNR are allocated as video 1 and video 2 at the sources S1 and S2, respectively. The initial rates are fixed to 230 kbps and 1600 kbps for sources S1 and S2, respectively. M&D video is located at source S1, while Foreman video is located at source S2. Each source delivers its video files to all participating peers, including the sources. Figure 3 shows the distribution throughput on source peers, while Figure 4 displays the receiving throughput of M&D and Foreman videos on a requesting peer.

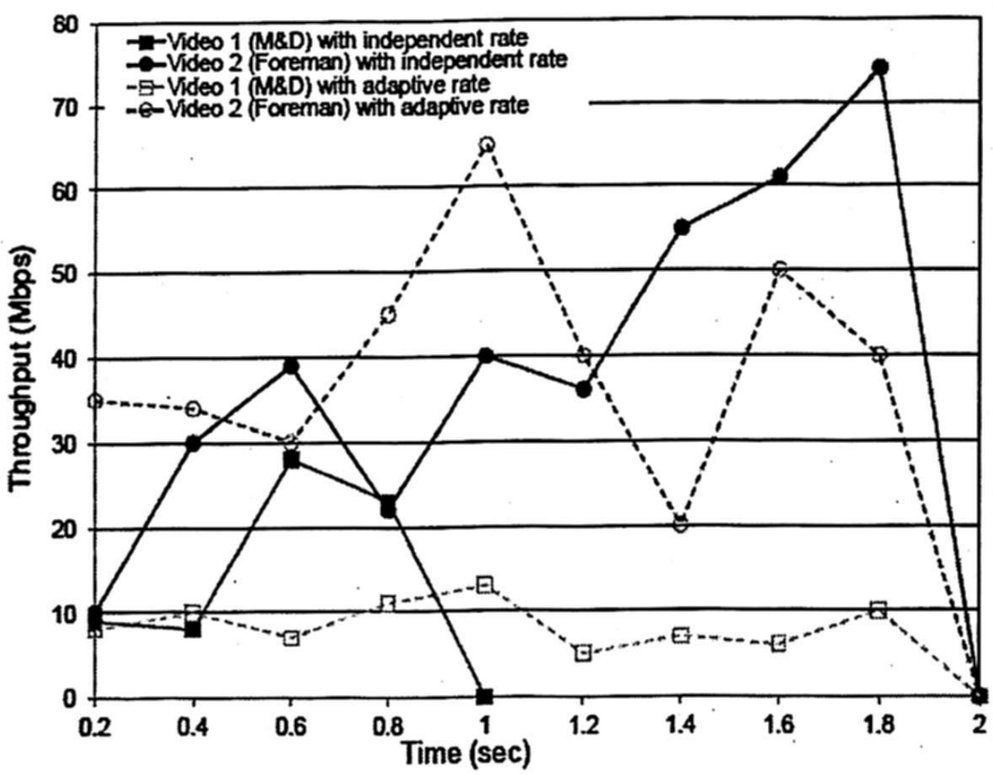


Fig.3. Distribution throughput on a source peer for two different video streams

We can see in Figure 3 that without our strategy of distribution control at the source peers, the distribution throughput of Foreman video on source peer S2 is larger than the distribution throughput of M&D video on source peer S1. The M&D video is distributed very quickly. Contrary, we can see that, with scheduling the distribution, each source peer can regulate the distribution throughput of its own video and the other video in proportion. Thus, the source ends to provide both videos at same time, but Foreman video requires greater source's throughput than M&D video. In Figure 4 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&M video. 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 same time. Therefore, the playback during receiving can be synchronously with low buffering demands.

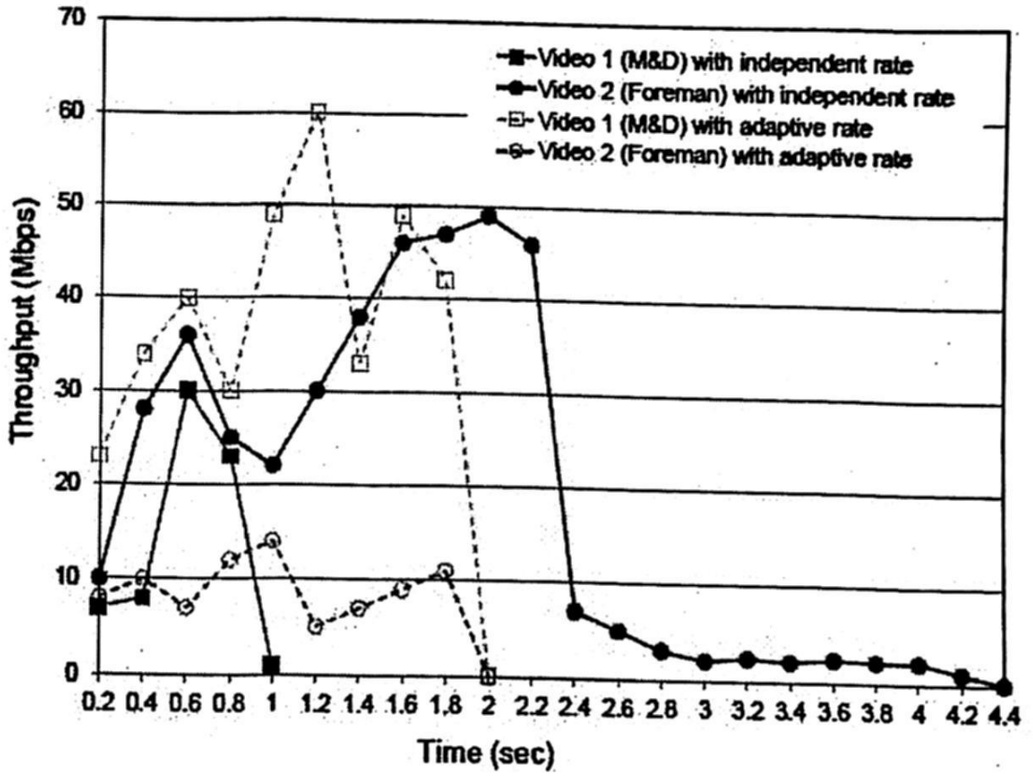


Fig.4. Receiving throughput on a requesting peer for two different video streams

Based on these results, we can see that an adaptive rate control and more collaboration between the sources allow that all video sequences achieve a similar PSNR (Peak Signal-to-Noise Ratio) quality. Thus, the average throughput can be used to recalculate the PSNR using JSVM (Joint Scalable Video Model) software, and we can obtain PSNR values of 43.4 dB and 43.9 dB for Foreman and Mother and Daughter, respectively.

5 Conclusions

Video applications are characterized by different resolutions designed for devices with different computational capabilities. In this paper, we have presented and evaluated a multi-source scheme to video streaming based on scalable video coding (SVC). Our proposed solution helps to reach similar video quality for all streams from multiples sources. Our proposed strategy was evaluated in a LAN infrastructure using simple experiments with four nodes. Our experiments show that we can reach a similar

video quality from multiple peers by using a strategy based on scalable video coding. Therefore, different videos can be received in similar times by a requesting peer independently of their size. Thus, our proposed scheme with SVC shows more effectively our scheme where SVC is not used. We believe that our proposed solution with SVC could be ideally used for peer-to-peer (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 model.

References

1. Tran, D. A., Hua K. A Do, T. T.: A Peer-to-Peer Architecture for Media Streaming. In: IEEE Journal on Selected Areas in Communication, Special Issue on Advances in Overlay Networks, (2003).
2. Zhang, X., Liu, J., Li, B., Yum, P.T-S.: CoolStreaming/DONet: A Data-driven Overlay Network for Efficient Live Media Streaming. In: 24th IEEE INFOCOM, Miami, FL, USA, pp.2102--2111, (2005).
3. Li, J.. Chou, P.A., Zhang, C.: Mutualcast: An Effcient Mechanism for One-To-Many Content Distribution. In: ACM SIGCOMM ASIA Workshop, (2005).
4. Schwarz, H., Marpe, D., Wiegand, T.: Overview of the Scalable Video Coding Extension of the H.264/AVC Standard. In: IEEE Transactions on Circuits and Systems for Video Technology, 17(9), pp.1103--1120, (2007).
5. Wien, M., Schwarz, H.,Oelbaum, T.: Performance Analysis of SVC. In:IEEE Transactions on Circuits and Systems for Video Technology, 17(9), pp. 1194--1203, (2007).
6. Schwarz, H., Wien, M.: The Scalable Video Coding Extension of the H.264/AVC Standard. In: IEEE Signal Processing Magazine, 25(2), pp. 135--141, (2008).
7. Reichel, J. Schwarz, H., Wien, M.: Joint Scalable Video Model (JSVM) 11, Doc. JVT-X202, Joint Video Team, Video Coding Experts Group, (2007).
8. Joint Video Team (JVT); JSVM software manual, version 9.12.2, Heinrich-Hertz-Institute, http://ip.hhi.de/imagecomG1/savce/ downloads/ SVCReferenceSoftware.htm.
9. López-Fuentes, F. A., Steinbach, E.: Multi-source video multicast in peer-to-peer networks. In: 22nd IEEE International Symposium on Parallel and Distributed Processing, Miami, Florida USA, (2008).
10. Lee, J. S., De Simone, F., Ebrahimi, T.: Subjective Quality Evaluation via Paired Comparison: Application to Scalable Video Coding. In: IEEE Transactions on Multimedia, Vol. 13, No. 5, pp. 882--893, (2011).
11. Wang, Y., Ostermann, J., Zhang, Y.- Q. In: Video Processing and Communications, Prentice-Hall, (2002).
12. Mirshokraie, S., Hefeeda, M.: Live Peer-to-Peer Streaming with Scalable Video Coding and Networking Coding. In: ACM SIGMM conference on Multimedia systems, MMSys '10, ACM, New York, NY, USA, 2010, pp. 123--132 (2010).
13. Abboud O., Zinner, T., Pussep, K., Oechsner, S., Steinmetz, R. and Tran-Gia, P. In: A QoE-Aware P2P Streaming System Using Scalable Video Coding. In: IEEE International Conference on Peer-to-Peer Computing (P2P), Delft, Netherlands. (2010).