Multicast Streaming of Hierarchical MPEG-4 Presentations

Christoph Neumann
INRIA Rhône-Alpes, Planète project, France
christoph.neumann@inrialpes.fr

Vincent Roca
INRIA Rhône-Alpes, Planète project, France
vincent.roca@inrialpes.fr

ABSTRACT

This work introduces a novel approach for the streaming distribution of hierarchically encoded MPEG-4 presentations using IP-multicast. The main achievements of this approach are: (1) it is massively scalable in terms of number of users, (2) it ensures the reception of a minimum quality of the video to everybody, (3) bursts of packet losses do not automatically lead to a sudden change of quality unlike in most other video streaming solutions, (4) it is naturally TCP friendly, (5) it is immediately deployable and does not rely on any QoS mechanism in the network. This solution is well suited to a large scale television program distribution over the Internet. Yet it is not recommended for video-conferencing and applications with user interaction because it introduces a large playing delay.

1. INTRODUCTION

This work deals with the streaming transmission of MPEG-4 presentations, where a client receives and plays information on the fly. Streamed MPEG-4 presentations are suited to a transmission in cumulative quality layers, where each layer gives a higher quality-level. A traditional solution consists in mapping these video layers onto several multicast groups. In order to perform congestion control, each receiver dynamically adapts the number of layers received according to the experienced losses [5]. In order to behave correctly, this solution requires that the video layer granularity be fine enough. Depending on the video codec used, this may not always be the case.

Another streaming solution consists in having a single video stream, mapped onto a single multicast group [6] [7]. In that case the source adapts the transmission rate (e.g. by changing video coding) according to RTCP feedback messages that give an indication on the experienced packet loss rate at receivers. Because of the presence of these control messages, this solution is not massively scalable.

Our proposal completely departs from these two approaches, and is in fact mid-way between reliable multicast file transfer and streaming. It largely relies on ALC, a layered reliable multicast protocol that enables a full asynchronism and independence between receivers.

The rest of the paper is organized as follows: we give a fast overview of the ALC reliable multicast protocol in section 2; then we introduce MPEG4 encoding in section 3; our proposal is described in section 4; we give experimental results in section 5; we summarize the results in section 6 and we conclude.

2. QUICK INTRODUCTION TO ALC

The Asynchronous Layered Coding (ALC) protocol [2] of the IETF RMT working group is a layered reliable multicast protocol. Information is sent on several (usually cumulative) layers at a fixed predefined rate. Then each receiver chooses how many layers he wants and can receive. This receiver initiated decision is taken by an associated TCP-friendly layered congestion control module (e.g. RLC [11], FLID [1] or WEBRC [3]). ALC does not use any feedback information to the sender. It is therefore massively scalable in terms of number of receivers.

ALC is well suited to the transmission of popular content in an "on-demand" mode, where clients join an ALC session, recover data, and leave the session at their own discretion. This is made possible by the use of FEC (Forward Error Correction) encoding [4], and by the transmission of all the packets, data of FEC, in a random order and continuously on the various ALC layers. This "on-demand" mode is very specific to ALC and other reliable multicast approaches (e.g. NORM and TRACK) are limited to a "push" synchronous transmission model where all clients are more or less supposed to be ready before the transmission starts. As we will see in section 4.1, our approach relies on this "on-demand" transmission model and is therefore intrinsically linked to ALC. More information on ALC and its use for multicast file transfer can be found in [2] and [10]. Our MCL library [9] offers a full featured ALC and RLC/FLID-SL implementation and forms the basis of this work.

3. MPEG-4 SCALABILITY

MPEG-4 scalability [8] offers the possibility to see a video at the desired spatio-temporal resolution by parsing appropriate portions of the bit-stream. One approach of scalability is the hierarchical layered video compression. It consists in having a base layer for low quality video encoding. On top of it one or more enhancement layers are added. Each layer enables to render the video with a higher quality. Yet in practice the number of enhancement layers is limited (e.g.

the MPEG-4 ISO Reference codec [12] only only one base and one enhancement layer).

4. GENERAL IDEAS

4.1 Sender behavior

The server partitions a media stream into *segments* of nearly the same duration, *delta*. This duration is by default one minute but other values are possible (section 4.5). Because the video is hierarchically encoding, a segment is composed of several video layers, that we call *blocks*. Each block is transmitted separately on a *distinct ALC session*, and thus on a different set of multicast groups (figure 1). All blocks of a segment are sent continuously during *delta* seconds, and after this period, the server automatically switches to the next segment (e.g. at time $t0+delta$ on figure 1). During each period, the packets of a block are not sent chronologically but in a random order. FEC packets included by ALC in the data stream enable receivers to efficiently reconstruct missed or not yet received packets. Transmissions follow an on-demand delivery mode (section 2) because receivers do not necessarily join at the beginning of a block transmission, especially for enhancement video layers. This is why ALC is absolutely required by our approach. More information on how ALC transmissions can occur can be found in [10].

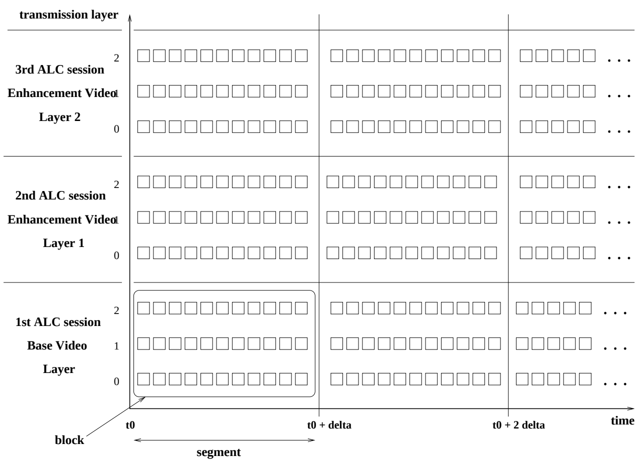


Figure 1: Sender behavior.

4.2 Receiver behavior

A receiver first subscribes to the ALC session corresponding to the Base Video Layer of the current segment, n. When the block of the Base Layer is successfully received, the receiver subscribes to the ALC session of the next enhancement video layer. This process stops when all enhancement layers have been successfully received (if ever) or when the transmission of the next segment, $n+1$, begins. When transmissions for segment $n+1$ start, the receiver plays the video of segment n and switches to the first ALC session to receive segment $n+1$, and so on. Therefore a receiver always plays the previous video segment while receiving the next one. This mechanism is illustrated in figure 2:

- From t0 to t3: the receiver tries to get the whole segment n and displays the segment $n-1$ already received.
- From t0 to t1: he receives the base video layer block of segment n. At t1 he has received enough data for this block and switches to the first enhancement layer.
- From t1 to t2: he receives the first enhancement video layer block. At t2 he has received enough data for this block and switches to the second enhancement layer.
- From t2 to t3: he receives the second enhancement video layer block. At $t3 = t0+delta$, end of the period, he stops without having enough data for this block. This block can either be entirely dropped (and won't be used at all during decoding), or be kept (to let the receiver try to decode it partially), depending on how the user perception of the video is managed.

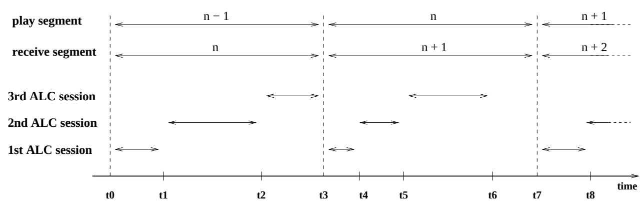


Figure 2: Receiver behavior.

4.3 A Solution Immediately Deployable Everywhere

With classical approaches, a client can receive several video layers simultaneously, and packets of a higher level can overhaul packets of lower levels. Such packets are useless since layers are cumulative, and the enhancement layers always need the data of the underlying layers. To avoid this problem, the presence of a differentiation mechanism (DiffServ or IntServ) is assumed in the core network to protect lower layers. This is a major requirement, which adds much complexity to the solution, and restricts its use to QoS-capable networks.

On the opposite our approach does not rely on privileged transmission schemes in the network and can immediately be deployed everywhere. A client receives only one video layer at a time, starting by the most important ones, to maximize the probability of receiving them correctly.

4.4 Benefits of Intra-ALC Congestion Control

Our approach takes advantage of the layered congestion control protocol used within each ALC session (we use RLC but a more elaborated protocol like WEBRC is possible). This is a major advantage over a congestion control based on adding/removing a video layer, as it is often suggested, since the granularity of MPEG-4 scalability is in practice very low (section 3). On the opposite our scheme benefits from fine grained congestion control protocols, currently being standardized and that can even take into account IGMP leave latency issues (section 4.5.2).

4.5 Segmentation

4.5.1 *The Need for Video Block Control Objects*

A control object is sent by the sender, for each video block, with the following information: the number of frames in the current video layer block, the first and the last timestamp, the last RTP sequence number, and the duration, *delta*, of the current segment. With this information a receiver knows when he completely received one layer of the current segment. He can also detect the end of the current segment and discard late packets of a previous segment.

4.5.2 Duration of a Segment Period

Several aspects must be considered to choose an appropriate *delta*. The first idea is to have a very short *delta* in order to reduce join latency and playing offset. But several considerations are against short segment durations.

The first one is the *congestion control timing limitations*. Congestion control protocols using static layering like RLC and FLID-SL (section 2) are limited by the IGMP leave latency problem (i.e. the delay between when the last receiver of a LAN leaves a multicast group and its effect). Therefore RLC imposes a deaf period after a loss during which everything is frozen. Besides, even without losses, because of the presence of congestion control a receiver cannot immediately receive at full speed, ALC layers being only added progressively (e.g. RLC imposes an exponential distribution of layer addition signals). These two facts require that each segment period be long enough. Using a dynamic layering approach for congestion control like FLID-DL or WEBRC would largely improve the situation since they solve this IGMP leave latency problem. Yet they also add a significant amount of complexity and induce other problems (in particular a high signaling load).

The second one is the *packet loss compensation capability*. Losses usually occur in bursts, because of congestion control problems or routing instability. In order to give a chance to receivers to recover losses (just by listening to an ALC session longer), the period duration must be large enough in front of the loss burst length.

Taking into account these aspects, we intuitively determined a trade-off between transmission efficiency and user interactivity, and set *delta* to one minute. Future work will refine this aspect using a more theoretical approach.

5. EXPERIMENTAL RESULTS

5.1 The Streaming Approaches Compared

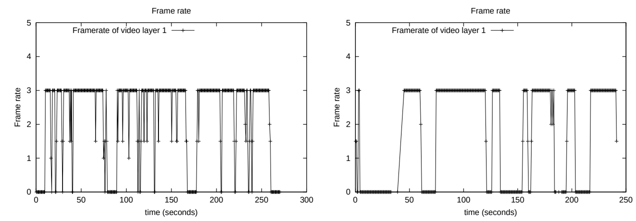
We implemented our proposal using the MCL library that implements the ALC/RLC/FLID-SL protocols [9]. We compare its performances with a classical streaming scheme using a single ALC session. In that case, each video layer is mapped onto an ALC layer. The sender sends each frame at the time it should be displayed. Therefore the transmission rate is not constant but reflects the instantaneous MPEG4 coding rate. No FEC protection or QoS mechanism is used for this classical approach.

5.2 Simulation Conditions

The video is encoded using spatial scalability, with two video layers, a base layer plus a single enhancement layer. The bit-rate of these layers is 43 kbps and 50 kbps respectively. The frame-rate is 3 frames per second, which is not realistic but sufficient for testing purposes.

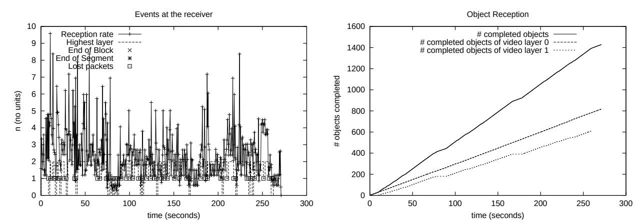
The sender simulates losses by dropping bursts of packets following a two-state Gilbert model. We chose a 1% loss probability in the lossless state, and a 75% loss probability in the lossy state, which leads to an average number of four consecutive packets lost.

We use a real testbed running the PIM-SM (Sparse Mode) multicast routing protocol, and two Linux PCs, a server and a client, attached to two different Ethernet subnets.



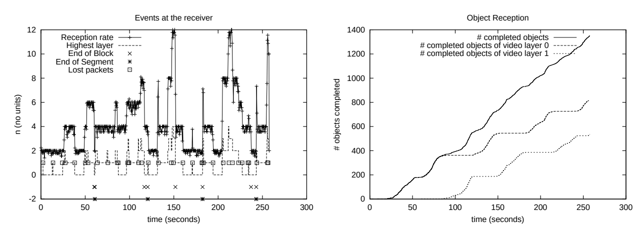
(a) classical approach (b) our approach

Figure 3: Frame-rate of the enhancement layer.



(a) reception rate (b) # received objects

Figure 4: Classical approach, PIM-SM network.



(a) reception rate (b) # received objects

Figure 5: Our approach, PIM-SM network.

5.3 Simulation Results

The overall performances in terms of number of received objects is about the same in both cases (figures 4 (b) and 5 (b)). Since an object corresponds exactly to a frame, it gives an idea on the quality of the displayed video.

With the classical approach the transmission rate for each video layer depends on the instantaneous video coding rate (the corresponding receiving rate is shown in figure 4 (a)). This is not the case with our approach where the sending rate is constant for a given period, and defined by the transmission rate of the various ALC sessions (not shown). Transmissions are smoothed over the time which is an advantage from a networking point of view.

Another difference is the distribution of lost frames. Figures 3 (a) and (b) show that the curve of the classical approach is much more irregular than with our approach. The classical approach, in presence of several losses, may not be able to reconstruct one more consecutive objects. As each object contains a frame, it leads to bursts of frame losses that largely degrade the users quality perception. Our approach is less sensitive to this problem.

Note that our loss model does not fully simulate a congested network (the limited duration of simulated loss bursts assumes the presence of a congestion control adaptation). Since the classical approach cannot react to such a situation, the network remains congested and performances decreases considerably, as well as the user perception. This would be particularly visible with several parallel sessions. On the opposite, since our approach includes congestion control, several parallel sessions would adapt around an equilibrium. This experiment (and others) is left for future work.

6. SUMMARY OF THE RESULTS

At the light of previous sections our approach offers the following benefits:

Pseudo-realtime streaming: The content can be produced, encoded and transmitted in real-time, and a client can play the video rapidly, after a small initial join latency. Yet the approach is not fully real-time since the delay between receiving a packet and playing its contents is of the order of one minute.

Unlimited scalability: The number of clients in a single streaming session is unlimited since there is no feedback flowing to the source in both the ALC and streaming components. Having one client or several millions makes no difference from the server point of view. Of course multicast routing protocols or additional application-level protocols may limit the global scalability by introducing mechanisms that require bi-directional transmissions, but this is out of the scope of the present work. Our approach fits well with the Source Specific Multicast (SSM) model, which simplifies multicast routing and offers a full class D addressing space to the source. Therefore multicast address allocation is not a problem and many concurrent streaming sessions are possible on a given server.

Ensures to each client the reception of the minimum video quality made possible by its access network: It is the result of the reception of video layers in the appropriate order, from the base layer first, up to the highest enhancement layer. The quality experienced by each client only reflects its own reception quality and has no impact on others.

A smoothed video quality: Because of the natural buffering capability and file transfer mode of the approach, packet losses, even in case of long bursts, do not automatically lead to frame losses and sudden video quality changes. They only increase the time spent receiving a given layer.

Congestion Control support: Our approach is TCP friendly, no matter how many layers the video codec produces. Recent and efficient congestion control protocols (e.g. WEBRC) can transparently be used in the ALC module.

Does not rely on any QoS mechanism in the Internet: Finally our approach does not need any QoS mechanism like DiffServ to define the relative importance of each layer. No assumption is made on the underlying network which makes our solution immediately deployable.

These properties make it well suited to the large scale pseudo-realtime distribution of television programs or videos over the Internet for instance.

7. CONCLUSIONS

We have introduced a multicast streaming solution for hierarchically encoded MPEG-4 presentations. We have shown its benefits compared to a classical multicast streaming solution where each video layer (base or enhancement) is directly mapped on a different multicast group. Our solution is in fact mid-way between reliable multicast file transfer and streaming. It is based on the ALC protocol to reliably transmit video segments. Therefore it benefits from ALC strength in terms of congestion control (TCP-friendly behavior), error recovery (packet bursts are easily recovered) and unlimited scalability (no feedback at all). It also takes full advantage of the “on-demand” delivery model offered by ALC. Our approach supports client heterogeneity and lets each of them experience the best possible display made possible by its access network. Finally our approach limits (up to a certain point) video quality instability by smoothing the effects of packet losses.

We implemented our approach and carried out a first set of experiments on a PIM-SM enabled testbed. Future work will improve these tests to better assess its benefits and limitations compared to other streaming solutions. The optimal value to assign to the segment duration parameter, at the center of this solution, will also be analyzed more formally.

8. REFERENCES

[1] J. Byers, M. Frumin, G. Horn, M. Luby, M. Mitzenmacher, A. Roetter, and W. Shaver. Flid-dl: Congestion control for layered multicast. In *2nd Workshop on Networked Group Communication (NGC2000)*, Nov. 2000.

[2] M. Luby, J. Gemmell, L. Vicisano, L. Rizzo, and J. Crowcroft. *Asynchronous Layered Coding (ALC) protocol instantiation*, Apr. 2002. Work in Progress: <draft-ietf-rmt-pi-alc-08.txt>.

[3] M. Luby, V. Goyal, S. Skaria, and G. Horn. Wave and equation based rate control using multicast round trip time. In *ACM SIGCOMM*, Aug. 2002.

[4] M. Luby, L. Vicisano, J. Gemmell, L. Rizzo, M. Handley, and J. Crowcroft. *The use of Forward Error Correction (FEC) in reliable multicast*, Feb. 2002. Work in Progress: <draft-ietf-rmt-info-fec-02.txt>.

[5] J. O. M. Nilsson, D. Dalby. Layered audiovisual coding for multicast distribution on ip networks. In *6th IEEE, European Workshop on Distributed Imaging*, Nov. 1999.

[6] H. N. Feamster, D. Bansal. On the interactions between layered quality adaption and congestion control for streaming video. In *12th International Packet Video Workshop, Pittsburgh, PA*, Apr. 2002.

[7] D. E. R. Rejaie, M. Handley. Layered quality adaption for internet video streaming. In *IEEE Journal on selected areas of communications (JSAC)*, 2000.

[8] R.Koenen. *Overview of the MPEG-4 Standard*, Mar. 2001. ISO/IEC JTC1/SC29/WG11 N4030.

[9] V. Roca. *MCL project home page.* http://www.inrialpes.fr/planete/people/roca/mcl/.

[10] V. Roca and B. Mordelet. Improving the efficiency of a multicast file transfer tool based on alc. In *IEEE International Symposium on Computer Communications (ISCC'02)*, July 2002.

[11] L. Vicisano, L. Rizzo, and J. Crowcroft. Tcp-like congestion control for layered multicast data transfer. In *IEEE INFOCOM'98*, Feb. 1998.

[12] S. A. J. Winder. *Version: Microsoft-FDAM-1-2.3-001213 User Manual*, Dec. 2000.