Streaming with Causality : A Practical Approach

Cezar Pleşca, Romulus Grigoraş
IRIT UMR CNRS 5505, ENSEEIHT
2 rue Camichel, Toulouse, France
{plesca, grig}@enseeiht.fr

Philippe Quéinnec, Gérard Padiou
IRIT UMR CNRS 5505, ENSEEIHT
2 rue Camichel, Toulouse, France
{queinnec, padiou}@enseeiht.fr

ABSTRACT

Highly interactive collaborative streaming applications express the need for causality. Solutions exist but we argue that more work needs to be done especially from a perceptual point of view. The key question is: given the current state of the Internet and the perceptual tolerance of causal desynchronization, does causality make any difference ? This paper proposes a practical answer to this question by comparing different solutions. We support this comparison by producing video results for a live streaming scenario on an experimental platform. Further, this paper proposes a novel approach for handling causality in multimedia and shows that it can perform better than Δ*-causality, usually considered the best solution.*

Categories and Subject Descriptors: Information Interfaces and Presentation H.5.4 Hypertext/Hypermedia: Architectures

General Terms: Measurement, Design, Experimentation.

Keywords: live streaming, group synchronization, causality, jitter

1. INTRODUCTION

Distributed applications are becoming increasingly complex, offering rich and powerful services to their users. In order to offer satisfactory performance, these applications are also becoming increasingly demanding in terms of communication support. Applications such as virtual environments, distributed simulation and computer supported collaborative work (CSCW) require the simultaneous use of several communication channels.

Each of these channels has its own QoS and is implemented by a specific protocol stack. QoS parameters are specified in terms of media quality and media relations (intra-stream and inter-streams synchronization) [1]. Stream synchronization is concerned with latency and jitter issues with respect to a single stream, as well as maintaining temporal relationships between different streams, called inter-stream synchronization (e.g. lip sync).

Same-time/different-place applications are concerned with interactions between dispersed groups of users, and they need to support multiparty communications [2]. Therefore, we have to consider a new type of synchronization, namely group synchronization.

Group synchronization ensures the coherence among all participants of an interactive session. They must have the same view or a similar, consistent view at virtually the same time. For example, in a highly interactive session, the video/audio streams exchanged by the participants are causally related in such a way that each participant needs to hear the answer to a question *after* seeing the question itself.

On one hand, research in distributed systems considered the problem of causality for multimedia synchronization [3, 4] and has recently concentrated on distributed environments [6]. On the other hand, for the applications exchanging audio/video streams, although there is a general agreement that there is a need for causality, actually very few applications have implemented it. We argue that streaming with causality needs to be more thoroughly investigated, especially from a perceptual point of view. This takes us to the key question: **given the current state of the Internet and the perceptual tolerance of causal desynchronization, does the use of causality make any difference ?**

This paper proposes a practical answer to this question by comparing different existing solutions. Further, it proposes a novel approach for handling causality in multimedia and shows that this can perform better than Δ-causality, usually considered the best solution. The practical character of our solution comes from exploiting the user's perceptual tolerance regarding causal inconsistencies.

In the next section we describe a distributed scenario that is sensitive to causal inconsistencies. Section 2 discusses existing solutions. Section 4 presents our solution: MModChannel. In section 5, we propose an experimental platform that allows us to evaluate the performance of each approach. To support our argument, we present analytical results together with reconstructed videos for a perceptual evaluation.

2. A LIVE STREAMING SCENARIO

Let us consider the case of a distributed multimedia application where a video stream (e.g. representing a soccer match) is sent live from site A to a group of users (figure 1). One of the participants (i.e. user B) starts to comment the video stream, by broadcasting an audio stream. These two streams are in a causal relation: the cause of the audio stream is the visualization of the video stream.

Figure 1 shows a possible consistency problem perceived by participant C : during the transmission, he or she may hear a segment of the audio stream (representing the exclamation "GOOOAL!") before seeing the part of the video stream containing the actual goal. This is a causal inconsistency since the exclamation is nothing but a consequence of a segment of the video stream showing the ball into the net.

Disparity in the speed of communication links as well as network congestion can contribute to causal order violations. The main rea-

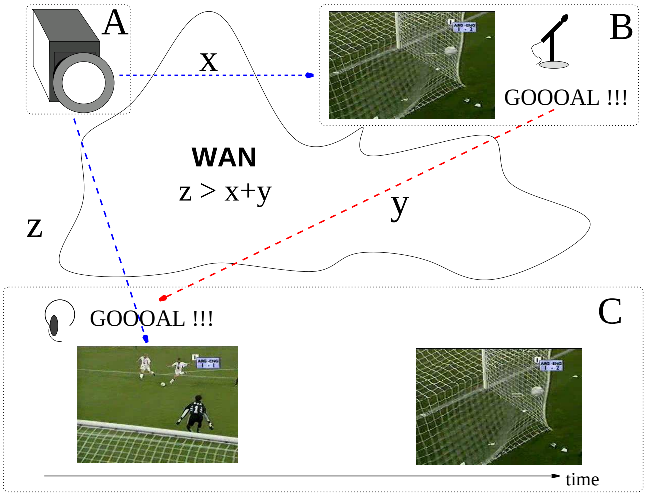


Figure 1. Consistency of Streams Perception

son for violation of causal ordering is the **triangle inequality** [7]. In our scenario, this occurred because a message sent from A to C passing through B takes less time to travel than one sent directly to C. In order to solve this causal inconsistency, the application should reorder the delivery of the streams according to the causal relations.

3. PREVIOUS WORK

Protocols widely used nowadays in multimedia applications (e.g. RTP-based, providing FIFO based on sequence numbers and inter-stream synchronization by means of timestamps) only provide point-to-multipoint ordering and synchronization, and therefore are unable to capture interactions specific to multiparty cooperative applications (e.g. causality). Multimedia research has put a lot of effort into synchronizing complex multimedia presentations composed of various inter-dependent streams [1]. However, a smaller effort has gone into group synchronization (consistency) according to relations that are not known beforehand, but occurring only when distributed users interact by exchanging streams multipoint-to-multipoint [2].

Using classic causality in distributed multimedia systems can affect in a negative way the quality of multimedia streams in terms of bandwidth, latency and jitter.

Bandwidth & Latency Classic causality requires complete reliability and therefore does not bound latency. It also generates an non-negligible bandwidth overhead that grows with the number of communicating sites. An approach to relax causality is Δ-causality that ignores some causalities in order to bound latency. Δ-CAUSAL protocols assure the causal consistency for messages arriving by Δ; beyond this limit messages are considered lost. Research on this topic [3, 4] has concentrated on adaptation issues (adjusting Δ) and optimizing the transmission (reduce the bandwidth overhead by minimizing piggybacking information).

Jitter Audio and video streams are very sensitive to jitter that can be reduced by buffering techniques. Research on this topic shows the tradeoff between intrastream synchronization (smooth playback at the receiver) and end-to-end delay [1]. The weakness of intrastream synchronization methods lies on their potentially poor delay performance that makes it inadequate for interactive applications. Heavily interactive applications (e.g. synchronous CSCW) require a small round trip time. Therefore, little or no buffer approaches are generally used to provide for the best interactivity.

As both CAUSAL and Δ-CAUSAL protocols assure consistency for every message (audio or video frame), they introduce jitter every time a triangle inequality arrises.

Considering the scenario from the figure 1, let's analyze the audio/video causal delivery on the site C. Video packets should be delivered before the corresponding audio packets. If the video is late (triangle inequality), then the audio is delayed. Therefore, on one hand, such situations would increase the jitter of the audio stream leading to interruptions in its presentation. From the user's point of view, an audio stream is more sensitive to such interruptions than a video stream. On the other hand, depending on the media content, short inconsistencies (e.g. 500ms) can be well tolerated by the user. Regarding final presentation quality, it would be more suitable to have a continuous audio stream and short inconsistencies rather than frequent interruptions to assure a complete consistency.

An interesting solution for consistency problems in replicated continuous interactive media have been studied in [6], where the author addresses the tradeoff between responsiveness and short-term inconsistencies by using the local lag concept. Instead of immediately executing an operation issued by a local user, the operation is delayed for a certain amount of time before it is executed. However, this is more appropriate for distributed environments or games rather than for interactive audio/video streaming.

In the next section, we propose an original approach: we bound the latency like Δ-CAUSAL **and** also tolerate inconsistencies up to a certain limit (e.g. inconsistencies no larger than 1s) in order to have reasonable jitter. To achieve this, we use a MModChannel communication service that offers the application a flexible degree of causal coherence trying to minimize the effects of consistency's respect on each individual stream's perception.

4. MMODCHANNEL: AN ORIGINAL SOLUTION

We propose a relaxation of the stream causality by assuring consistency only between some points in the streams. Obviously, inside each stream we have to assure the FIFO order. Concerning the group synchronization problem stated above, one can observe the need for combining weak (FIFO) and strong communication protocols (CAUSAL). We will show that causally sending audio and video streams can be refined in: sending some frames on a CAUSAL channel; sending the rest of the frames on a FIFO channel. Our approach is to combine partial order protocols by using our MMod-Channel implementation [8] of abstract channel communications [5].

In [8], we proposed a generic MModChannel that can properly handle the mix of any delivery (partial) orders. An application that needs to exchange messages using different coordinated channels can create a MModChannel by specifying the stack protocols for each of the elementary channels.

An elementary channel is an abstraction that represents a subset of the messages exchanged by the participants connected to this channel. These messages are delivered according to a specific order (e.g. FIFO, CAUSAL or TOTAL) of the channel. MModChannel offers the application a simple interface for exchanging messages through a subset of its elementary channels, w.r.t. each of them.

Sending When sending messages, the application may choose to transmit the *same message* on a set of elementary channels. Such messages act as a synchronization points for these channels.

Receiving The delivery process ensures that every message is delivered to the receiving application in the correct (partial) order of every channel it has been sent through.

In order to solve the causal problem stated above (figure 1), we use a MModChannel composed of two elementary channels : FIFO and CAUSAL. The causal consistency is captured on site B and we observe the correct delivery on site C. On site B, we need to express the causal dependency between the video stream (received from A) and the audio commentary (sent from B).

The figure 2 shows how the audio and video streams are causally synchronized on site B:

- emission of every frame of both streams on FIFO;
- emission of some frames (synchronization points) on CAUSAL.

This assures FIFO order inside each stream and CAUSAL consistency between synchronization points. But how to choose the synchronization points ? Naturally, we are tempted to express the causal relation exactly as it occurs (between V_i and A_i). On the site C, inconsistencies are tolerated only between 2 consecutive synchronization points. Unfortunately, video synchronization points arriving late postpone the corresponding audio points (and also the next packets), leading to jitter.

To eliminate this disadvantage, we take advantage of the user's perceptual tolerance and choose the audio synchronization point shortly after its corresponding video (after T_{causal}). This method relaxes the precise synchronization between audio and video points seen earlier ($V_i \leftrightarrow A_i$) and does not tolerate inconsistencies larger than $T_{video} + T_{causal}$ between streams. In fact, the worst inconsistency would occur when the video frame V_i is presented together with the audio frame $A_{m'-1}$. $A_{m'-1}$ is the furthest audio frame that could be presented before V_n because $A_{m'}$ depends causally on V_n. Moreover, according to the FIFO order, V_n follows after V_i.

Let us analyze what happens if A_i is received but the frame V_i is late. Causal protocols stop the audio stream until V_i arrives. Our MModChannel proposal continues to FIFO-deliver audio packets up to A_m because they do not depend on V_i. Obviously, we can't tolerate any amount of inconsistency: the audio stream is interrupted if, when A_m is scheduled for delivery, V_i is still missing. A proactive solution to this last problem is playout rate adjustment [9]: we can slow down the delivery of the audio packets and therefore give more time to V_i to arrive.

Playout rate adjustment can also be used as a reaction to causal interruptions. In such situations, a number of frames blocked by a causal violation need to be consumed faster than normal. Research on this topic has shown that changing the playout rate of video and audio up to 25% is often un-noticeable.

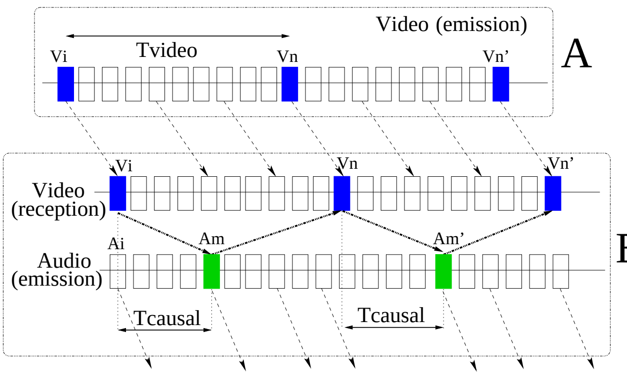
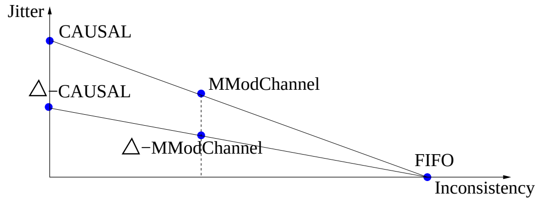


Figure 2. MModChannel Flexible Consistency

MModChannel can be easily used, in a relative fashion, for other distributed scenarios like videoconferencing or synchronous groupware applications.

5. COMPARING THE SOLUTIONS

In this section, we analyze the performance of our approach by comparing it with FIFO, CAUSAL and Δ-CAUSAL protocols. We go beyond simulations and produce real videos from the streams exchanged during the experiments. This allows us to perceptually evaluate the importance and the quality of causal consistency.



FIFO is a reliable point-to-point protocol that does not handle consistency between streams coming from different sources. Therefore, network conditions changes on the video path do not affect the delivery of the audio stream. CAUSAL assures reliability and tight consistency between the two streams. Consequently, any change in one stream delivery will affect the other stream's delivery. In other words, latency's variations of the video stream also disturb audio stream. Δ-CAUSAL assures causal consistency for messages arriving by Δ (i.e. the others messages being discarded). Meanwhile, like CAUSAL, latency's variations for video frames arrived by their causal deadline (Δ) still imply jitter on the audio stream. MModChannel (FIFO+CAUSAL) provides lighter consistency while trying to maintain a low jitter for the audio stream. It achieves this as it does not react immediately to late video frames by tolerating some inconsistencies. Δ-MModChannel (FIFO+Δ-CAUSAL) works in a similar way as MModChannel for messages arriving by Δ.

5.1 Experimental setup

Tools We emulate WAN conditions on our LAN using NIST Net. This tool allows us to experiment with various network parameters (packet delay, jitter, bandwidth limitations, congestion, packet loss and duplication) on a live network using a Linux router. We defined, for each pair of sites, some particular traffic conditions (e.g. loss and delay) compatible with statistics from [**?**].

Conditions We conducted experiments using our soccer streaming scenario and make it last 45 seconds. During the session, we considered a latency of 20±2 ms for the path A→B and a latency of 100±10 ms for the path B→C.

Time (sec)	0-15	15-30	30-45
Latency (ms)	100 ± 10	300 ± 50	900 ± 100
Loss	0%	2%	5%

Table 1. Network conditions for the path A→C

Streams' characteristics The video stream is encoded using MJPEG at a framerate of 20fps and requires a bandwidth of 746 Kbps. The audio stream is in PCM format and is streamed at 128Kbps.

Protocols' configuration The trickiest point of our experimental setup is the choose of the right parameters of the protocols. For example, the value of Δ should be sufficiently low in order to bound latency ($\Delta - Causality_{\Delta\to\infty} = Classic\ Causality$) and large enough to have acceptable loss. We have chosen Δ=1000ms. The parameters for the MModChannel are $T_{causal} = 400$ ms and $T_{video} = 800$ ms. This means sending every 16^{th} frame on CAUSAL, and placing audio synchronization points 400 ms after their corresponding video.

5.2 Experimental Results

We compare five protocols according to the jitter they induce on the audio stream, and the observed audio/video inconsistencies (figures 3 and 4). We use a playout rate adjustment of 25%. The average values are computed on a window of 2.5 seconds and the results for 3 periods are:

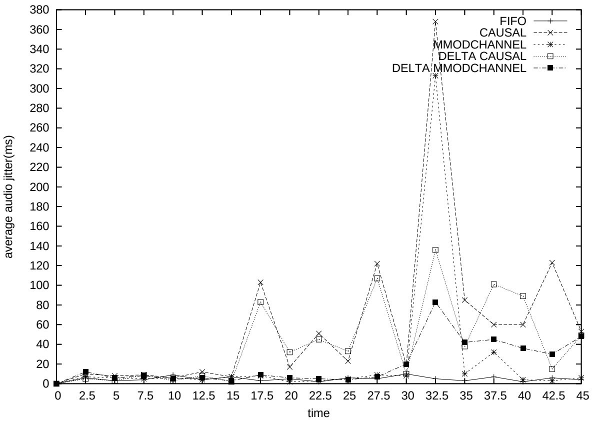


Figure 3. Average Audio Jitter

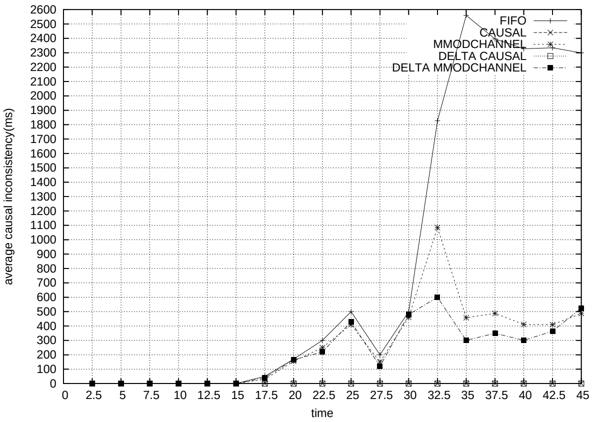


Figure 4. Average Causal Inconsistency

Interval 0-15s As the network does not present the triangle inequality, we can observe that all protocols work fine : low jitter and low causal inconsistencies.

Interval 15-30s CAUSAL and Δ-CAUSAL assure perfect consistency. Unfortunately, the consistency is obtained at the price of audio jitter. FIFO keeps the jitter acceptable but obviously does not handle inconsistencies. MModChannel and Δ-MModChannel cope very well with the audio jitter while observing acceptable inconsistencies (max. 500ms).

Interval 30-45s The dramatic change of network latency seriously affects the audio jitter for CAUSAL and MModChannel protocols whereas theirs Δ-correspondents introduce smaller jitter. The jitter decreases for the rest of the period for reliable protocols (CAUSAL and MModChannel) as audio packets are buffered until the video causally related ones are received. Unreliable protocols (Δ) buffer less and, as they wait up to Δ to receive lost messages, they will observe more jitter than reliable ones. FIFO observes a minimal jitter but does not protect from large inconsistencies.

While CAUSAL and Δ-CAUSAL protocols protect user from all inconsistencies, the two MModChannel maintain inconsistencies under 1200ms (i.e. $T_{causal} + T_{video}$).

To analyze the perceptual quality of different approaches, we decomposed the real audio and video streams at the frame level. Tracing the sending/receiving times of each frame/packet from the emulation process, we could reconstruct the streams as they would be presented by means of MJPEG Tools. The reader can "feel" the results of our experiments by watching the reconstructed videos available online at `http://www.1webspace.org/acm2005/index.html`.

6. CONCLUSION AND PERSPECTIVES

We are now able to give an answer to our hamletian question: *causality or no causality* for multimedia streaming ?

Most of the time, when there is no or little triangle inequality, we have showed that point-to-point protocols (RTP-based FIFO) are sufficient. Nevertheless, under certain conditions (mostly when there is important network jitter leading to a large triangle inequality), classic Δ-CAUSAL protocols provide complete consistency at the expense of important media jitter. Our MModChannel solution performs better by trading-off consistency and media jitter. However, our approach is a delicate mix of Δ-causality, playout rate adjustment and perceptual parameters that suggest several areas for future research. For example, we need to find an answer to the following question: given a network configuration and the characteristics of the streams, how to choose the optimal MModChannel parameters ? Moreover, would it be possible to finely adjust them with network changes ? We may also need to conduct a more thorough perceptual study with various contents, users and media types.

Regarding jitter, we think that better results could be achieved if we had to consider the integration of a small, reasonable playout buffer (100-250 ms) into our approach. With the availability of PlanetLab, we plan to validate our approach by evaluating MModChannel results, as well as its scalability, in real networks.

MModChannel is a powerfull tool for experimenting with various partial order protocols such as causality. It would be interesting to make use of its strength for studying others applications that can accommodate with flexible consistency (e.g. networked games). MModChannel can also be used for assuring consistency of a complex streaming presentation (e.g. written in SMIL), where different streams can be delivered in a partial-order relation.

7. REFERENCES

[1] N. Laoutaris and I. Stavrakakis, *Intrastream Synchronization for Continuous Media Streams: A Survey of Playout Schedulers, In* IEEE Network Magazine, *2002.*

[2] H. Miranda and L. Rodrigues, Flexible Communication Support for CSCW Applications, *In* SPIRE/CRIWG, *1999.*

[3] T. Tachikawa, H. Higaki and M. Takizawa, Δ-Causality and Epsilon-Delivery for Wide-Area Group Communications, *In* Computer Communications, *2000.*

[4] R. Baldoni, R. Prakash, M. Raynal and M. Singhal, Effi cient Δ-Causal Broadcasting, *In* Journal of Computer System Science and Engineering, *1998.*

[5] Saul P. Hernandez, J.Fanchon and K.Drira, Abstract Channels as Connectors for Software Components in Group Communication Services, *In* Fifth Mexican International Conference in Computer Science, *2004.*

[6] M. Mauve, J. Vogel, V. Hilt and W. Effelsberg, Local-lag and Timewarp: Providing Consistency for Replicated Continuous Applications, *In* IEEE Transactions on Multimedia 6, *2004.*

[7] F. Adelstein and M. Singhal, Real-Time Causal Message Ordering in Multimedia Systems, *In* International Conference on Distributed Computing Systems, *1995.*

[8] C. Pleşca, R. Grigoraş, G. Padiou and Ph. Quéinnec, A Flexible Communication Toolkit for Synchronous Groupware, *In* International Conference on Multimedia Communications Systems, *2005.*

[9] H. Liu and M. El Zarki, Adaptive delay and synchronization control for Wi-Fi based AV Conferencing, *In* IEEE International Conference on Quality of Service in Heterogeneous Wired/Wireless Networks, *2004.*