数理解析研究所講究録
996巻 1997年 181-194

Real-Time Synchronization Machines for Continuous Media Data

京大数研 川勝 則孝 (Noritaka Kawakatsu)
京大数研 中島 玲二 (Reiji Nakajima)
京大数研 田辺 誠 (Makoto Tanabe) *

Abstract

On multimedia systems, it is an important issue how to play continuous media data (CM data) with given temporal requirements (or QoS). In this paper, we introduce real-time synchronization machines for CM data (RSM), a class of mathematical machines, which are intended to input multimedia data consisting of a single or more streams of CM data and make temporal corrections on their outputs according to the given constraints, such as jittering corrections and inter-/intra-synchronization.

An RSM is represented by a directed graph, where each node corresponds to a process that inputs of the frames of a stream of CM data and controls their outputs. On the other hand each edge is labeled by a time interval and a frequency ratio between streams. A basic cycle of RSM behaviours consists of inter-process communications of the intervals which specify permissible ranges of frame outputs.

*{kawakatu,reiji,tanabe}@kaba.or.jp

In addition, we discuss various properties and applications of RSM, including a clock regarded as a CM stream, improvement of playout quality by buffering, and realization of intra-/inter-synchronization with the ratio of m to n.

1 Introduction

We define CM data to be media data, which change continuously as time advances. Digitalized CM data are regarded as a set of one or more media streams, where each stream is a sequence of data units, called frames. On multimedia systems, it is an important issue to send multimedia data through networks (which consists of a single or more streams of CM data) from the source site to the presentation site and play out them with reasonably acceptable condition.

The output time of each frame is expected to observe a certain temporal constraint with respect to its corresponding frames. In order that these constraints hold, so called inter-/intra-synchronizations are needed. In the example of multimedia data called a movie, which consists of two streams i.e. video and audio, the intra-synchronization includes jittering corrections of video(or audio) frames. On the other hand, the inter-synchronization will be concerned with lip-synching between video and audio. These temporal constraints

on the output of the CM data determine whether the quality of their outputs is good enough to see or hear comfortably (temporal part of QoS). This issue is particularly worth serious consideration when the site of playout is remote from the data source and the data are sent through networks.

Many of works in the area of the problem have been centered on the analysis of specification QoS and data representation issue of CM data. Little work, however, has been done on the concrete problem of how to realize temporal corrections.

We introduce ***real-time synchronization machines for continuous media data*** (RSM), a class of mathematical machines which are intended to input multimedia data consisting of a single or more streams of CM data and make temporal corrections on their outputs according to QoS. Here, we would like RSM to be a model of synchronization mechanism, which

- has as abstract representation as possible,
- works without assumption of existence of time-stamps,
- is constructed easily and systematically from corresponding QoS,
- and outputs CM data to meet the QoS if possible.

Related works on multimedia system's principles include Petri Net approaches, such as OCPN [3] [4] [8] and TSPN [5], which

Media	type of QoS	[min,max] of jitter
video	intra-jitter	$[p, q]$
audio	intra-jitter	$[p', q']$
video/audio	inter-jitter	$[r, s]$ (frequency ratio 1 : 3)

Figure 1: Required QoS for the multimedia data of A/V

specify playouts of CM data, and the Tempo Project [2](a dual language approach), where one language is for the abstract specification, and the other if for concrete specification. PDS [7] is a deduction system which analizes playout of media data with assumption of existence of some playing mechanism like RSM. The periodic threads of RT-Mach [6] may be useful to implement the mechanism of RSM.

2 Behaviour of RSM, an informal definition

Firstly, we give an example of RSM(figure 2) in order to illustrate how RSM work. This RSM is a machine which inputs audio and video streams, make temporal corrections(if possible) and outputs them in order to play out the multimedia data consisting of the two streams with the required QoS (figure 1). It consists of two

processes V and A, which input the video(V) and audio(A), make intra-jitter corrections and output them, respectively. In addition, the process V sends a constraint interval to the process A for the inter-synchronization. Their behaviour is as follows:

As for the intra-synchronization of video,

- the process V outputs the video frame within the interval $t + [p, q]$, if it is possible, where t is the output time of the previous frame and $[p, q]$ is labeled at the edge cycling on the node V.

If output specified as above is not possible, the frame is dropped and no temporal restriction for intra-synchronization is imposed on the output time of the next frame. (We will discuss more details later.)

As for the intra-synchronization of audio and the inter-synchronization,

- the process V sends the interval $\tilde{t} + [r, s]$, on outputting the current frame, to the process A through the interval queue, where $\tilde{t}$ is the output time of the frame, and
- the process A is supposed to output three audio frames for each single video frame with the inter-synchronization (as 'three', the frequency ratio, is labeled at the inter-node edge). Then, the permissible range of the audio frame becomes $(u + [p', q']) \cap (t + [r, s])$. Here (figure 3), u is the expected time of the current au-

dio frame and ought to be in the intersection, where t is the latest output time of video frame (inter-synchronization) and u is the previous output time of audio frame (intra-synchronization). For u and u', only intra-synchronization is imposed.

In case of dropping the video frame, no temporal restriction is imposed on the output time of the corresponding audio frame. (More precisely, there is some restriction as described later.)

Note that for the initial condition, the process A and V output their initial frames at the same time.

Here, we give a description of behaviour of general RSM. An RSM is a directed graph. There, each node corresponds to a process controlling a stream of input CM data. Each edge stands for the temporal inter-constraint between them if it is between two nodes (e.g. the $[p, q]$ edge of figure 2). If an edge starts from and returns to the same edge, it gives the intra-constraint. An edge is labeled by a time interval, where the time interval specifies the range of differences between corresponding frames of the start and destination nodes. If it is between two nodes, it is also labeled by a natural number, where the natural number is the frequency ratio of the synchronization between the two nodes (e.g. In figure 2, three audio frames correspond to a single node). Let us consider the

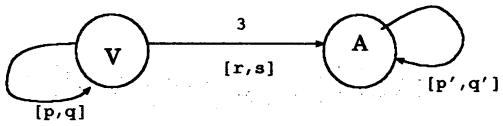

Figure 2: An example of RSM

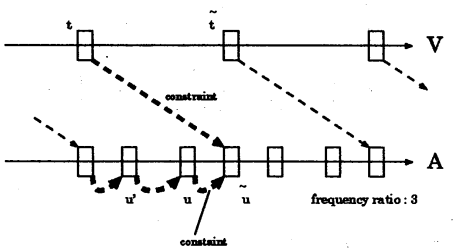

Figure 3: Correspondence by inter-/intra-synch.

process A (figure 4). The basic cycle of behaviour of the process is (i) receiving the current frame, (ii)receiving the constraint intervals from all edges that direct to A, (iii) computing the intersection of them (called the validity interval for the frame), and (iv) outputting the frame within the validity interval if it is possible. More precisely,

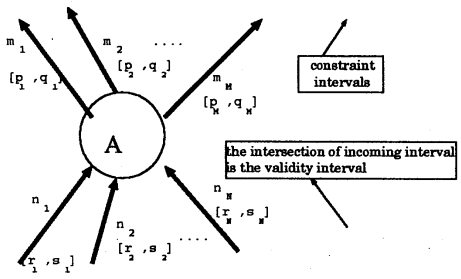

Figure 4: A general node

1. The process
   - picks out the intervals $[x_i, y_i]$ from all the edges directing to the node A, (These intervals are computed at all the nodes from which the edges directing to A start, in the way described below.)
   - assigns the interval $\cap_i([x_i, y_i])$ to the validity interval for the current frame, and
   - sends $\tilde{t}+[p_j, q_j]$ as the constraint intervals through each edge starting from A, where $\tilde{t}$ is the output time of the frame.
2. Drops the frame if outputting is impossible, and sends $[-\infty, \infty]$ as the constraint interval.
3. Puts each constraint interval into the queue of the corresponding edges together with $n_j - 1$ dummy intervals($[-\infty, \infty]$) (which are needed in order to realize the $1 : n_j$ inter-synchronization).

Note, here, that though $[-\infty, \infty]$ seems to be impose no temporal restriction, it actually specifies that the output by the destination node must not precede the corresponding one of the start node. For example, consider the above example (figure 2). The process A sends the constraint intervals to B as soon as it outputs (or drops) frames. On the other hand, the ith frame of B is allowed to be output (or dropped) only after the ith interval arrives.

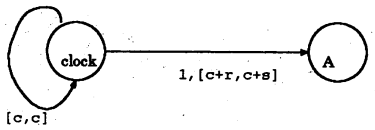

Figure 5: An RSM with a clock medium

Note also that at the initial stage, the process A synchronizes the initial output with those of its (directly or indirectly) connected nodes. (In other words, all the processes waits for its initial output until the last process gets ready.)

3 Discussion

We discuss some important cases of RSM, which deal with several typical kinds of playouts of CM data.

3.1 Playout of CM data with the synchronized clock

A playout of CM data is often specified as a sequence of almost periodic outputs of the frames, which can be restricted by the relative time interval from the periodic times of a clock. Figure 5 gives an RSM which tries to output CM data with this property.

In this RSM, the clock process inputs virtual frames and outputs them with the period of c. The clock process sends a constraint interval and the process A assigns it to the validity interval, where

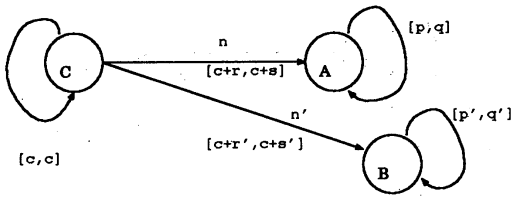

Figure 6: An RSM to realize $m : n$ inter-synchronization

After all, the process A is supposed to work periodically according to the clock.

3.2 $m : n$ inter-synchronization

The natural number n labeled on an edge stands for the frequency ratio of the synchronization is $1 : n$. A single edge, which connects two nodes directly, can not specify the more general frequency ratio $m : n$ between the two nodes. Let two CM streams, corresponding to the two nodes, be synchronized with a certain clock, it is possible to construct an RSM that try to synchronize the two streams with the frequency ratio $m : n$ as the figure 6 shows. (Note that the RSM satisfies the condition, $c \in [np, nq] \cap [p', q'] \neq \emptyset$.)

This RSM tries to (intra-)synchronize these two streams and (inter-)synchronize every n frames of the stream A and every n' frames of the stream B with every single frame of the clock stream C.

It makes intra-synchronization of each of the two streams A and B so that time differences between consecutive frames of the stream A may not exceed the range $[p, q]$,and those of the stream B may not exceed the range $[p', q']$. Also, it makes inter-synchronization between the streams A and B with the frequency ratio $A : B = n : n'$ so that time differences from each frame of the stream A to its corresponding frame of B may be within the range $[p' - q, q' - q]$.

3.3 Buffering (the time advance model)

Any RSM is supposed to have a buffer of frames of the corresponding stream. There, however, the concept of prefetch of frames has not yet been introduced. Prefetch of frames is a technique of inputting frames early enough to make temporal corrections and buffering them until they should be output, which enable RSM to output CM data with better quality of the playout than what they have had on arriving if the arriving CM data have a certain type of QoS.

It is possible to introduce the concept of the time advance model [1] to RSM in order to enable them to prefetch frames. The time advance model is used in order to guarantee a certain type of QoS, namely that the average period is fixed. Intra-/inter-jitter of out-

puts of frames are bounded, when delay or prefetch is permitted. In case of RSM with the time advance model, a clock of an RSM starts some seconds($=\delta$) later than the first frame inputs. We give an example of RSM (figure 5) with the time advance model. Let the average period, the minmal jitter and the maximal one be c,x and y, respectively. When the advance time δ is not taken into account, this RSM inputs CM data and tries to make temporal corrections and outputs CM data with the quality of $c = c$,$x = r$ and $y = s$ successfully if it inputs the CM data with the quality of $c = c$,$x = 0$ and $y = s$. The RSM, with the advance time δ, however, output CM data successfully if it inputs the CM data with the quality of $c = c$,$x = -\infty$ and $y = s + \delta$.

4 Conclusion

We introduce a class of mathematical machines, RSM, in order to input CM data, make temporal corrections and attempt to output them with given requirements. RSM

- are constructed straightforwardly from the requirements, called temporal QoS,
- use the simple principles of behaviour, sending and receiving time intervals and outputting frames within the intervals, for

the inter-/intra-synchronizations,

- can simulate the concept of clock media,
- can realize inter-synchronization with the frequency ratio of m to n, and
- can better the quality of CM data using the time advance model.

Therefore, RSM can serve as a basic framework of the mechanism of synchronizations and temporal corrections for multimedia systems.

References

[1] Roger B. Dannenberg and Dean Rubin. A comparison of streams and time advance as paradigms for multimedia systems. Technical Report CMU-CS-94-124, CMU, 1994.

[2] Howard Bowman Gordon Blair, Lynne Blair and Amanda Chetwyd. Formal support for the specification and construction of distributed multimeida systems(the tempo project). Technical Report MPG-93-23, Department of Computing, Lancaster Univerity, 1993.

[3] Thomas D.C. Little and Arif Ghafoor. Synchronization and storage models for multimedia objects. *IEEE Journal on Selected Areas in Communications*, 8(3):413–427, April 1990.

[4] Thomas D.C. Little and Arif Ghafoor. Interval-based temporal models for time-dependent multimedia data. *IEEE Transactions on Data and Knowledge Engineering*, August 1993.

[5] Patrick Sénac Michael Diaz. Time stream petri nets, a model for timed multimedia infomation. *Lecture Notes in Computer Science*, (815):219–238, 1994.

[6] Tatsuo Nakajima. The operating system supporting operations of continuous media data(in japanese). *bit*, pages 13–24, February 1995.

[7] M Tanabe and R Nakajima. A logical basis for continuous-media presentation with qos. In *Multimedia Modeling '95*, The International Conference of Multimedia Modeling. IEEE, World Scientific, November 95.

[8] Miae Woo, Naveed U.Qazi, and Arif Ghafoor. A synchronization framework for communication of pre-orchestrated multimedia information. *IEEE Network*, pages 52–61, January/February 1994.