On Synchronized Presentation Management in Multimedia Database Systems

Aidong Zhang
Department of Computer Science
State University of New York at Buffalo
Buffalo, NY 14260

Extended Abstract

The extension of database systems to support multimedia applications requires new mechanisms to ensure the synchronized presentation of multiple media data streams. In order to flexibly and efficiently present multimedia data streams to users, media streams must be segmented into media objects and time constraints among these objects must be specified and maintained. New management tools to effectively manage the time-related characteristic of multimedia data must thus be superimposed on existing database systems. In this paper, we discuss issues relevant to synchronized presentation management in multimedia database systems. We present principles underlying the synchronized presentation of multiple media data streams when delay effects are considered. Various synchronization mechanisms can be designed based on the proposed principles.

1 Introduction

Multimedia data refers to the simultaneous use of data in different media forms, including images, audio, video, text, and numerical data. Many multimedia applications, such as recording and playback of motion video and audio, slide presentations, and video conferencing, require continuous presentation of a media data stream and the synchronized display of multiple media data streams. Such synchronization requirements are generally determined by either spatial or temporal relationships that need to be specified among multiple data streams. For example, a motion video and its caption must be synchronized spatially at the appropriate position in a movie, and, in a slide presentation, a sequence of images and speech fragments must be temporally combined and presented to compose one unified and meaningful data stream. This research will focus on only temporal synchronization.

Current database systems are not equipped to represent the entire multimedia data flow, nor may it be desirable for them to support the many retrieval types required by the variety of multimedia data. Thus, continuous media data must be parsed into representable segments, which we term *media objects*. In order to re-present the original data stream to users, synchronization constraints among media objects must be specified and maintained. Such synchronization is usually termed *intra-synchronization*. An additional complication arises with the timing relationships that may exist among the different types of media data streams. Such media data streams may not be merged prior to storage in a database, and such a merger will vastly compound the difficulties of retrieving component media. Thus, the synchronization of multiple media data streams, termed *inter-synchronization*, becomes an essential prerequisite to any successful multimedia database application. For these reasons, synchronization has been recognized as one of the central problems in multimedia system development [LG90, Ste90].

Substantial research has been directed toward the support of synchronization within operating systems and network architectures [RV93, Ste90, AH91, RRK93, GR93]. However, the techniques proposed at these levels are insufficient to address the problems encountered at the presentation level in database systems.

For example, Anderson et al. [AH91] describe techniques for recovering from loss of synchronization between interrupt-driven media I/O devices, and Rangan et al. [RRK93] devise techniques for inter-media synchronization during on-demand multimedia retrival from a server to multiple destinations over integrated networks. Both research efforts are directed toward the synchroniztion of two types of media streams: master and slave streams. In cases of asynchrony, synchrony is restored by deleting (skipping) media units of a slower slave by the amount it lags behind the master, or by duplicating (pausing) media units of a faster slave by the amount by which it leads the master. Consequently, slower slaves catch up with the master, and faster slaves are forced to wait for the master. However, in the application level, it may not be clear whether there exists such a master/slave relationship between media data streams. Furthermore, the presentation of some media streams may not allow skipping if complete preservation of content is crucial to the presentation.

This paper will propose a framework for supporting continuous and synchronized presentation of multiple media data streams in a multimedia database system. We primarily seek to develop

- A practical framework, with sound theoretical underpinnings, for specifying multimedia data and inherent synchronization requirements; and
- Scheduling principles for synchronized presentation of multiple media data streams.

Our framework proposes approaches to restore synchrony on the presentation of asynchronic multiple media streams. These approaches include a buffering strategy and criteria for scheduling the skipping/pausing of media data streams with asynchronous presentations. These principles guide the design of the various algorithms needed to implement the presentation of multiple media streams. The pragmatic design of these presentation mechanisms is considered elsewhere.

2 Multimedia Data Model

A multimedia data stream consists of a set of data upon which some time constraints are defined. The time constraints may specify discrete, continuous, or step-wise constant time flow relationships among the data. For example, some multimedia streams, such as audio and video, are continuous in nature, in that they flow across time; other data streams, such as slide presentations and animation, have discrete or step-wise time constraints. These multimedia streams may not have convenient boundaries for data representation. To facilitate retrieval of such data in databases, we must break each media stream into a set of media objects. Each media object represents a minimum chunk of the media stream that bears some semantic meaning. Media objects in different media streams may have different internal structures. For example, a continuous video stream can be segmented into a set of media objects, each of which contains a set of video frames with specific semantic meaning. Similarly, a continuous audio stream can be segmented into a set of media objects, each of which contains a set of audio samples with specific semantic meaning. Without loss of generality, we assume that basic data elements delivered from the transportation layer are media objects.

Media objects from different data streams may need to be linked through time constraints to specify their synchronization. For example, in slide presentation applications, an audio object must be played along with a slide object. We define a *multimedia unit* u_i to be the composition of a set of media objects $o_{1i}, ..., o_{ni}$, where o_{ni} represents an individual media stream participating in the synchronized stream. Thus, a composite data stream made up of multiple media streams consists of a set of multimedia units, whereas each multimedia unit unifies media objects from multiple media streams. Such multimedia units may also be considered as composite objects. Furthermore, a collection of objects from either a single data stream or a composite data stream may be conceptually modeled as a hierarchical structure. At each increasing level, a set of media objects may be considered to be a superclass object which may then, in turn, be a component of another superclass. Further development of such a conceptual model will not be pursued here.

We will now consider the synchronization requirements that need be placed on media data streams. There are two types of constraints to be specified on media objects: intra- and inter-stream constraints. Intra-stream constraints specify the synchronization requirements to be placed on a single media stream and inter-stream constraints specify the synchronization requirements to be placed on more than one media stream.

Let m_i be a single media stream which consists of a set of media objects $o_{i1}, ..., o_{il}$. Intra-stream constraints on $o_{i1}, ..., o_{il}$ define time flow relationships among these objects. These intra-stream constraints may be continuous, discrete, or step-wise constant. Unlike temporal constraints defined on data in temporal databases [TCG+93], media objects are not typically statically associated with independent time constraints. Owing to the sequential character of media objects in a single media stream, each media object must be associated with a relative start time and a time interval which specifies the duration of its retrieval, assuming that the first media object in the media stream starts at time zero. The actual start time of a media object is usually dynamically determined. Once a media stream is invoked, it is associated with an actual *start time*; each media object within that stream will similarly be associated with an actual start time. We use $< o, t, \triangle t >$ to denote that object o is to be presented at time t and will last time period $\triangle t$.

Let $\mathcal{U}$ be a composite data stream of media streams $m_1, ..., m_n$. $\mathcal{U}$ can then be considered as consisting of a set of multimedia units $u_1, ..., u_l$, with each multimedia unit u_i $(1 \leq i \leq l)$ being a composition of media objects $o_{1i}, ..., o_{ni}$ $(1 \leq i \leq l)$ from $m_1, ..., m_n$, respectively. Inter-stream constraints on $u_1, ..., u_l$ define synchronization requirements among the participating component media streams. Time-related inter-stream constraints are implicitly defined in each media object. That is, for any two media objects o_{ki}, o_{ji} $(1 \leq k, j \leq n)$, we have $t_{ki} = t_{ji}$ and $\triangle t_{ki} = \triangle t_{ji}$. Other inter-stream constraints are specified on media objects as inter-dependency relationships that must be satisfied before the retrieval of the media objects is invoked. As discussed in [COC94], boolean expressions for media objects provide a simple but elegant way to expresss these inter-dependency relationships. For example, consider a training program that consists of three media streams m_1, m_2, and m_3. Let m_1 contain media objects $o_{11}, ..., o_{1l}$, m_2 contain media objects $o_{21}, ..., o_{2l}$, and m_3 contain media objects $o_{31}, ..., o_{3l}$. Let inter-dependency relationships specify that o_{1i}, o_{2i}, and o_{3i} must be played together for all $i = 1, ..., l$. We then have the following inter-stream constraints defined on o_{1i}, o_{2i}, and o_{3i} for all $i = 1, ..., l$, respectively:

$o_{1i}^D = o_{2i} \wedge o_{3i}$; $o_{2i}^D = o_{1i} \wedge o_{3i}$; $o_{3i}^D = o_{1i} \wedge o_{2i}$.

3 Principles of Synchronizing Media Data Streams

In this section, we will investigate the principles of synchronizing media data streams. We assume that the transportation level provides sufficient support for delivering media objects on time. A framework will be developed to permit efficient buffering and the resynchronization of the presentation of multiple media streams in the event of delays.

Intra-stream synchronization maintains intra-stream constraints that are defined on a single data stream. As discussed in Section 2, each data stream must be segmented into a set of media objects in order to facilitate flexible retrievals. The re-presentation of the original data stream can be achieved by ensuring the intra-stream constraints defined on the media objects of the stream. In this context, the central issue is to provide an efficient buffering mechanism so that neither starvation nor overrun will occur.

Let t_{r_s} be the time at which the reading of data stream m begins and *reading function* $R_m(t, t_{r_s})$ be the total number of media objects of m read at time t. Let t_{c_s} be the time at which the consumption of data stream m begins and *consuming function* $C_m(t, t_{c_s})$ be the total number of media objects consumed at time t. The number of media objects that must be buffered at any given time is then $R_m(t, t_{r_s}) - C_m(t, t_{c_s})$, which we denote as

$$B_m(t, t_{c_s}) = R_m(t, t_{r_s}) - C_m(t, t_{c_s}). \tag{1}$$

In (1), t_{r_s} is fixed when the reading of data stream m is initiated and $t_{c_s} \geq t_{r_s}$. We then want to find the smallest t_{c_s} such that $B_m(t, t_{c_s}) \geq 0$ in the range $t_{c_s} \leq t \leq t_{r_f}$, where t_{r_f} is the time at which the reading of data stream m is completed. Obviously, if $R_m(t, t_{r_s}) \geq C_m(t, t_{r_s})$ for any t in the range $t_{r_s} \leq t \leq t_{r_f}$, then $t_{c_s} = t_{r_s}$. Othervise, let the minimum value for $B_m(t, t_{r_s})$, in the range $t_{r_s} \leq t \leq t_{r_f}$, be at time t_{min} with $B_m(t_{min}, t_{r_s}) = -k$, where $k > 0$.

Suppose that a solution is to begin playback at time x. That is, $B_m(t, x)$ is at least zero for any time $x \leq t \leq t_{r_f}$. If we compare $B_m(t, x)$ with $B_m(t, t_{r_s})$ in the range $x \leq t \leq t_{r_f}$, we see that

$$\begin{aligned} B_m(t, x) &= R_m(t, t_{r_s}) - C_m(t, x) \\ &= R_m(t, t_{r_s}) - C_m(t, t_{r_s}) + C_m(t, t_{r_s}) - C_m(t, x) \end{aligned}$$

$$= B_m(t, t_{r_s}) + C_m(t, t_{r_s}) - C_m(t, x). \tag{2}$$

This shows us that, in the range $x \leq t \leq t_{r_f}$, $B_m(t, x)$ differs from $B_m(t, t_{r_s})$ by $C_m(t, t_{r_s}) - C_m(t, x)$. In (2), we know that, in the range $t_{r_s} \leq t \leq t_{r_f}$, $B_m(t, t_{r_s}) \geq -k$ and $B_m(t, x) \geq 0$. Thus, x must be the minimum start time such that

$$C_m(t, t_{r_s}) - C_m(t, x) \geq k. \tag{3}$$

Let us now consider a situation in which the consuming function is linear [GC92]. That is,

$$C_m(t, x) = \begin{cases} 0 & \text{if } t < x, \\ r_c(t - x) & \text{if } t \geq x. \end{cases}$$

Following (3), in the range $x \leq t \leq t_{r_f}$, we have

$$\begin{aligned} & r_c(t - t_{r_s}) - r_c(t - x) \geq k, \\ & r_c x - r_c t_{r_s} \geq k, \\ & x \geq t_{r_s} + \tfrac{k}{r_c}. \end{aligned} \tag{4}$$

Thus, x should be $t_{r_s} + \frac{k}{r_c}$. As the left side of (4) is a constant when x is determined, $B_m(t, x)$ and $B_m(t, t_{r_s})$ thus have the same slope. As demonstrated by Gemmell [GC92], this value of x is also the intersection of $R_m(t, t_{r_s})$ with $B_m(t, t_{r_s}) + k$. In general, the consuming function may not be a single linear function or even a continuous function. In such a situation, we must ensure that formula (3) always hold true.

Inter-stream synchronization maintains inter-stream constraints that are defined on multiple data streams. As media objects from different data streams are composed into multimedia units in the multimedia database, various inter-stream constraints specified on those units must be maintained during the re-presentation of these data streams. We will now discuss the maintenance of these inter-stream constraints.

Let $\mathcal{U}$ be a multimedia stream of multimedia units $u_1, ..., u_m$ which are from media streams $m_1, ..., m_n$. Let each multimedia unit u_i $(1 \leq i \leq l)$ consist of media objects $o_{1i}, ..., o_{ni}$. Each media object o_{ki} $(1 \leq k \leq n)$ in u_i has intra-stream constraint $< o_{ki}, t_{ki}, \triangle t_{ki} >$. Obviously, if all media objects can be displayed at the defined time, then all intra- and inter-stream constraints can be easily maintained. However, the presentation of a media object may be delayed for many reasons, including network delays and physical storage delays. When such delays arise, simply cancelling the whole presentation with its combined multiple media streams may not be an acceptable option. We thus investigate the principles of resynchronization among media objects in the event of delays.

We define a *synchronization point* to be a point held by all participating media streams needing to be synchronized. Each point of separation between two multimedia units forms a handy synchronization point. Users may define additional synchronization points within each multimedia unit. Synchronization points defined on the composite stream specify the places that synchronous presentation must be checked and maintained. Let $\triangle s_i$ denote the maximum time interval that media m_i can skip and $\triangle d_i$ denote the current delay time interval with the presentation of media stream m_i. We then have the following situations:

(1) $\triangle d_i \leq \triangle s_i$ for all integer i in the range $1 \leq i \leq n$.

(2) For all i $(1 \leq i \leq n)$, if $\triangle d_i > \triangle s_i$, then $\frac{1}{2}\triangle d_i \leq \triangle s_i$.

(3) There exists some $\triangle d_i$ $(1 \leq i \leq n)$ such that $\frac{1}{2}\triangle d_i > \triangle s_i$.

In case (1), synchronous presentation can be restored by simply skipping the amount the delayed media streams lag behind. However, this approach may be inefficient. It requires either substantial extension of buffer space to hold extra media data or pausing of undelayed media streams to wait for the delayed media streams to be read into the buffer. Let $\triangle d_{max}$ be the maximum delay interval occurred among the delayed participating media streams. If each delayed media stream m_i skips the smaller time interval of either $\frac{\triangle d_{max}}{2}$ or $\triangle d_i$, and simultaneously, the undelayed media streams pause $\frac{\triangle d_{max}}{2}$ interval, the size of buffer can then be reduced and the pausing interval is also shortened. Clearly, case (2) can follow the same strategy. In case (3), pausing on the undelayed media streams is unavoidable. However, handling of partial delay of each media stream m_i within the interval of $2\triangle s_i$ can still be compromised beween skipping and pausing based on the above approach. In practice, as actual delays are not known in advance, the criteria on maximum delay permitted on media streams must be given. Due to space limitation, detailed analysis will not be pursued.

Following the above approach, the basic buffering criterion given in (2) and (3) must be revised in the event of delays. Let D_m be the number of media objects that need to be buffered because of delays. We then have

$$\begin{aligned} B_m(t,x) &= R_m(t,t_{r_s}) - (C_m(t,x) + D_m) \\ &= R_m(t,t_{r_s}) - C_m(t,x) - D_m \end{aligned} \tag{5}$$

A buffering mechanism can be designed based on the above formula; this will not be considered further in this paper.

4 Conclusions

In this paper, we have discussed issues relevant to synchronized presentation management in multimedia database systems. Resynchronization criteria in the event of delays are formulated to support the maintenance of both intra- and inter-stream constraints. The concepts proposed here provide a framework for the design of the mechanisms necessary for multimedia presentation in database systems. The implementation of a multimedia presentation tool on top of an object-oriented distributed database system is currently under investigation. This implemetation is being conducted using O_2, a well-known object-oriented database system created in France.

References

[AH91] David P. Anderson and George Homsy. A Communication Media I/O Server and Its Synchronization Mechanisms. *IEEE Computer*, 24(10):51–57, 1991.

[COC94] J. Courtiat, R. Oliverira, and L. Carmo. Toward a New Multimedia Synchronization Mechanism and its formal Specification. In *Proceedings of ACM Multimedia 94*, pages 133–140, San Francisco, California, October 1994.

[GC92] Jim Gemmell and Stavros Christodoulakis. Principles of delay-sensitive multimedia data storage and retrieval. *ACM Transactions on Information Systems*, 10(1):51–90, January 1992.

[GR93] Shahram Ghandeharizadeh and Luis Ramos. Continuous Retrievals of Multimedia Data Using Parallelism. *IEEE Transactions on Knowledge and Data Engineering*, 5(4):658–669, 1993.

[LG90] Thomas D.C. Little and Arif Ghafoor. Synchronization and Storage Models for Multimedia Objects. *IEEE Journal on Selected Arears in Communications*, 8(3):413–427, April 1990.

[RRK93] P. Venkat Rangan, Srinivas Ramanathan, and Thomas Kaeppner. Performance of Inter-Media Synchronization in Distributed and Heterogeneous Multimedia Systems. *Computer Networks and ISDN Systems*, 1993.

[RV93] P. Venkat Rangan and Harrick M. Vin. Efficient Storage Techniques for Digital Continuous Multimedia. *IEEE Transactions on Knowledge and Data Engineering*, 5(4):564–573, 1993.

[Ste90] Ralf Steinmetz. Synchronization Properties in Multimedia Systems. *IEEE Journal on Selected Arears in Communications*, 8(3):401–412, April 1990.

[TCG+93] A.U. Tansel, J. Clifford, S. Gadia, S. Jajodia, A. Segev, and R. Snodgrass. *Temporal Databases: Theory, design, and Implementation.* The Benjamin/Cummings Publishing, 1993.