Int J Digit Libr (1997) 1: 386–400

INTERNATIONAL JOURNAL ON
Digital Libraries

Scheduling multimedia presentations in educational digital libraries*†

Aidong Zhang, Thomas V. Johnson

226 Bell Hall, Department of Computer Science, State University of New York at Buffalo, Buffalo, NY 14260, USA;
e-mail: azhang@cs.buffalo.edu

Abstract. An educational digital library is a specialized digital library containing instructional materials, such as class lectures, seminar presentations, and various training materials. These materials consist of a combination of audio, video, and image data. In such an environment, basic parts of multimedia data are usually stored in databases and sophisticated multimedia presentations may be assembled to generate various presentations. In this paper, we investigate a theory of the scheduling strategies for supporting the synchronized presentations of multimedia streams which is applicable to educational digital libraries. This scheduling theory includes the specification and representation of synchronization on media streams, the realization of appropriate synchronization granularity, and the scheduling principles for the presentations of multimedia streams. This investigation formulates criteria for specifying and scheduling the skipping/pausing of media streams with asynchronous presentations when various delays occur. Adaptability to various quality-of-service requirements is supported in the scheduling strategies. Various synchronization mechanisms at both client and server sides are proposed to implement the scheduling theory. Experimental analysis is conducted using instructional materials.

Key words: Synchronization – Multimedia presentation – Presentation scheduling – Educational digital libraries

1 Introduction

With the availability of digital media and the advent of the Internet, it is now possible to access large multimedia repositories distributed throughout the world. Such access is becoming increasingly important in a number of applications, including education and training, medical diagnostics, manufacturing, and distributed publishing. Sophisticated multimedia presentations may need to be constructed in these applications. For example, in asynchronous distance learning, basic parts of multimedia presentations stored in multimedia databases may be assembled into different presentations for individual learning and training [35, 3]; in medical diagnosis for training, a sophisticated presentation containing images, video clip, and audio may be assembled from data stored in multimedia databases. Because of the complexity and flexibility of such applications, the existing work in multimedia research cannot be directly applied to such environments.

To meet the demands of the above real-world applications, database systems must have the ability to efficiently store, manage, and retrieve multimedia data. However, there is a recognized lack of fundamental principles and advanced techniques with which one can design a system to support sophisticated multimedia database applications. Because multimedia data differ from conventional data particularly in their time-sampled nature, current database techniques are insufficient to fully support their management. Also, because of the flexibility requirement of constructing comprehensive presentations from databases and a variety of different expectations and requirements regarding the retrieval and presentation of multimedia data, current multimedia techniques cannot be directly used in the database environment. Moreover, different applications are associated with various expectations and requirements regarding the storage and display of multimedia data. These include a variety of quality-of-service (QoS) requirements and styles of user interactivity. Accommodating such diverse expectations and the flexibility needed to access various repositories in a distributed setting poses many challenges.

We target a class of multimedia repositories (termed *educational digital libraries*) which contain instructional materials, such as class lectures, seminar presentations,

* This paper is an extended version of the paper that appeared in the Proceedings of the International Conference on the Advances in Digital Libraries (ADL), Washington, DC, May 7–9, 1997

† This research is supported by NSF under grant IRI-9632394

and various training materials. These materials consist of a combination of audio, video, and image data. Such an environment has diverse uses, including student training, human resource development, and asynchronous distance learning. Accessing such materials on line imposes special system requirements to support various quality of service. The design of the system using database techniques will support the flexibility in the storage and retrieval of such multimedia data. In a multimedia database environment, multimedia data are usually organized into media objects and object classes, thus providing users with a clear-cut method for data specification. In order to re-present the original data stream to users, synchronization constraints among media objects must be specified and maintained. Moreover, if a composite stream is composed of media objects from different media streams, additional complications may arise with the timing relationships that may exist among the different types of media streams. Such media streams may not be merged prior to storage in a database as such a merger will vastly compound the difficulties of retrieving component media. Thus, the synchronization of multiple media streams, becomes an essential prerequisite to any successful multimedia database application [29, 40]. The following example demonstrates an application.

Example 1. *Consider an application involving online computer-assisted learning in undergraduate education. Without loss of generality, we assume that there are two media streams, audio and slides (or video). Synchronization within the slide stream may require that two objects either overlap or be sequentialized. Synchronization within the audio stream requires only that objects be sequentialized. Additional Synchronization requirements between the two media streams are specified among slides and audio objects. In order to successfully display both streams to a student, the system must ensure that all time constraints placed on the individual display operations and the synchronization between slides and audio objects are preserved within a satisfiable scope. Deviations between slides and audio objects greatly affects the quality of the presentation of the streams. (See Fig. 1)*

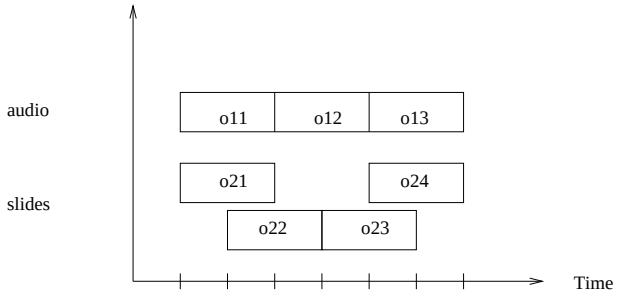

Fig. 1. A presentation of streams

1.1 Related work

Several studies have investigated the modeling of multimedia data from a conceptual perspective, including graphical models, Petri-net based models, object-oriented models, collaborative model, and temporal abstraction models [46, 14, 29, 42, 30, 9, 17, 16, 41, 11]. Some models deal primarily with the synchronization aspects of the multimedia data, while others are more concerned with facilitating the browsing of the multimedia data. These models provide illustrative vehicles for specifying synchronization semantics at the application level. The extensive research of such models are multimedia authoring to provide sophisticated tools for specifying various multimedia presentations [3, 10, 24, 25, 35]. The importance of formulating QoS metrics for continuity and synchronization specifications in media streams has also been realized [47, 6, 34, 39, 48].

Substantial research has been directed toward the support of multimedia stream synchronization within operating systems and network architectures [38, 40, 5, 37, 20, 50, 18, 34, 32, 33]. A significant amount of work has been contributed to server design; in particular, attention has been focused on service time scheduling and admission control. Research has also centered on the network scheduling of media stream transmissions. Various QoS parameters have been incorporated into the consideration of network traffic. Several communication protocols have been proposed to support deterministic, probabilistic, or best-effort transmission of media streams over networks. However, since no synchronization is considered at the client sites, relatively simple combination of media streams can be effectively supported.

Research involving multimedia databases is in its infancy [2, 1, 8]. In [19], a collection of related research papers has been published. In particular, specific issues in storage and retrieval of multimedia data have been summarized [27], and object-oriented approaches are used to design multimedia database management systems [12, 36]. Multimedia data modeling has been addressed in [21]. In recent research, several approaches have been proposed to address specific problems at the system design level. For example, Moser, Kraib, and Klas [31] propose a buffer strategy termed "least/most relevant for presentation." This buffer strategy investigates the effects of user interactions such as "rewind" and "fast forward" on buffer design. A mechanism is proposed which supports minimum waiting time after user interactions. In [22], we reported our initial examination of buffer model and management in distributed multimedia database environments. Other initial approaches for designing distributed multimedia database systems are realized in [45, 44, 43].

These research efforts described above have set the stage for a fuller investigation of maintaining sophisticated multimedia presentations at the client sites in the distributed multimedia database environment. More importantly, because various system requirements must be enforced for different domains of applications, approaches need to be developed to target a specific class of applications.

1.2 Contributions of this paper

In this paper, we will investigate the fundamental principles and criteria for synchronized presentations of multimedia data reconstructed from databases. The specific problems include precise definition of synchronization requirements among different media streams, the integration of various QoS requirements, the realization of an appropriate synchronization granularity among media streams, and multimedia presentation scheduling criteria for scheduling the skipping/pausing of media streams with asynchronous presentations when unexpected delays occur. Adaptability to various quality-of-service requirements is supported in the scheduling strategies. Various synchronization mechanisms at both client and server sides are then proposed to implement the scheduling theory. Experimental analysis is conducted using instructional materials. Thus, the proposed scheduling strategies can be readily used to facilitate various education and training applications, such as student training, human resource development, and asynchronous distance learning.

The remainder of this paper is organized as follows. Sect. 2 discusses the system, data, and QoS models. In Sect. 3, we investigate synchronization specification and granularity. In Sect. 4, we present the scheduling principles and algorithms to ensure the synchronous presentation of media streams in the event of delays. In Sect. 5, we introduce an adaptive approach to the admission and scheduling of client requests. Concluding remarks are offered in Sect. 6.

2 Preliminaries and definitions

2.1 The system model

The system architecture (named NetMedia) for educational digital libraries is illustrated in Fig. 2 [23]. This architecture represents a client-server distributed environment which includes a set of NetMedia servers that is distributedly superimposed upon top of a set of database management systems (DBMSs), a set of multimedia databases, and a set of NetMedia clients which support client access to the NetMedia servers. Each DBMS manages the insertion, deletion, and update of the media data stored in the local database. We assume that the clients accessing NetMedia system through the global NetMedia client interface can only request multimedia data retrieval. Thus, no global updates on media data located in local media databases will be allowed. Furthermore, as certain media streams may be represented and stored in different formats, the underlying DBMSs can be heterogeneous.

The NetMedia client contains a presentation scheduler which is responsible for synchronizing images, audio, and video packets received from the underlying media databases via the network and displaying them to the output devices on the client workstation. Thus, skipping or dropping of data to maintain presentation synchronization must be supported by the NetMedia client. In addition, the NetMedia client will request the server to vary the rate at which data are sent if changes are needed in the presentation schedule. The NetMedia server supports the multi-user aspect of media data caching and transmission. It maintains real time retrieval of media data from the multimedia database and transfers the data to the client sites through the network. Skipping or dropping of data also need to be supported at the server sites. In case the network does not have enough bandwidth to transfer the requested data or a client is requesting fast-forwarded data, some media data may be skipped before transmission. The interpretation of the media data is delegated to workstations at the client locations.

We assume that our system architecture resides on top of guaranteed network services, as many results illustrating methods to provide such services are now mature, such as RSVP [49]. To guarantee the continuity of multimedia data delivery over the network, we have

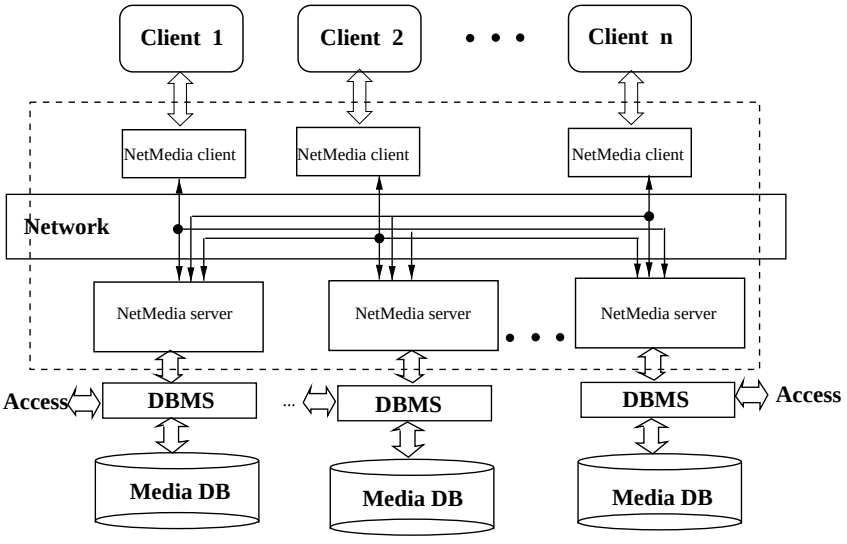

Fig. 2. NetMedia system architecture

investigated a two-phase buffer model and management. Buffer management is needed at both client and server sites to ensure that the retrieval (or transmission) of media streams will not cause hiccups in their presentation. Buffer management at client sites will ensure that transmission of media streams from the servers will not cause hiccups in their presentation. Buffer management at server sites will ensure that the retrieval of media streams from disk to memory will not cause delays in their transmission. The details on this aspect have been discussed in [22]. We will not discuss it further in this paper.

2.2 Data and presentations

A media stream can be viewed abstractly at several levels. At the lowest level, a media stream is viewed as an unstructured *BLOB* (binary large objects) which can be further categorized within several higher-level object classes [30, 15, 13]. Objects from different media streams may also be spatio temporally combined into multimedia objects. Several conceptual data models which follow this general scheme have been proposed. We assume that each media stream is broken into a set of atomic objects. Each atomic object represents a minimum chunk of the media stream that bears some semantic meaning. Atomic objects in different media streams may have different internal structures. For example, a continuous video stream can be segmented into a set of atomic 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 atomic objects, each of which contains a set of audio samples with specific semantic meaning. Higher levels of object classification are then formulated on atomic objects. Each atomic object is associated with a relative start time and a time interval which specifies the duration of its retrieval, with the initial atomic objects in the media stream assumed to start 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.

The atomic objects within a media stream are linked together through *intra-synchronization* time constraints. These constraints may specify discrete, continuous, overlapping, or step-wise constant time flow relationships among the atomic objects. For example, some multimedia streams, such as audio and video, are continuous in nature, in that they flow across time; other streams, such as slide presentations and animation, have discrete, overlapping, or stepwise time constraints. It may, for example, be necessary to display two distinct slide objects jointly within a single slide presentation stream. In general, the temporal relationship between two atomic objects in a single stream may conform to any of the thirteen temporal relationships described in [4]. Media objects from different streams may need to be linked through time constraints to specify their synchronization; such time constraints are termed *inter-synchronization* constraints. For example, in slide presentation applications, an audio object must be played along with a slide object. The temporal relationship between two atomic objects from different media streams may also conform to any of the thirteen temporal relationships described in [4]. Inter-synchronization constraints may be specified as meta-data or specified in task programs. In some cases, the relative time and time interval associated with an atomic object may need to be adjusted to conform with these inter-synchronization constraints.

We will now discuss the proposed multimedia presentation model. Since our primary concern with multimedia data involves retrieval rather than update, our model will consider only display operations of atomic objects. For the elements of a multimedia presentation, we assume the availability of three basic operations: $start(t)$, $end(t)$, and $display(o,t)$, where $start(t)$ and $end(t)$ are beginning and termination operations at a relative time t, and $display(o,t)$ is a display operation of object o at relative time t. A *multimedia presentation* consists of a set of $start(t)$, $end(t)$ and $display(o,t)$ operations upon which synchronization constraints are specified to enforce both intra- and inter-synchronization constraints.

2.3 QoS parameters

We will now discuss QoS parameters and the effect of these parameters in the scheduling of multimedia presentations. The scheduling of multimedia presentations includes the scheduling of time-dependent display operations, synchronized display enforcement among multiple media streams in a multimedia presentation, concurrent retrieval of multimedia streams at the server sides, and display delay recovery. A correctness criterion in this context must verify that display operations are performed according to a predefined synchronization pace and within the time constraints imposed on display operations. Since the correctness of time-based presentations depends on the accuracy of timing that must be maintained on media streams, the execution result of a multimedia presentation is a question of quality rather than consistency. We must thus formulate new correctness criteria for the executions of multimedia presentations which define acceptable quality in real time. Several important QoS parameters must be considered in these correctness criteria.

Various QoS parameters have been defined to specify the quality of service requirements on the presentations of multiple streams [28, 41, 48]. For example, Little and Ghafoor [28] have proposed several parameters to measure the QoS for multimedia data presentation. The following parameters have been listed: 1) average delay, 2) speed ratio, 3) utilization, 4) jitter, and 5) skew. The *average delay* is the average presentation delay of each object in a time interval. The *speed ratio* compares the actual presentation rate to the nominal presentation rate. The *utilization* equals the ratio of the actual presentation rate to the available display rate of a set of objects. Ideally, both the speed and utilization ratios should equal 1. The *jitter* is the instantaneous difference between two

synchronized streams. The *skew* is the average difference in presentation times between two synchronized objects over n synchronization points.

The above QoS measurement needs to be mapped into the design of the scheduling algorithms. Four main parameters to be used have been identified in [48], which are incorporated into this paper:

- Aggregate loss factor (ALF): this is the average number of media granules dropped or skipped from a media object of any stream.
- Cumulative loss factor (CLF): this is the maximum number of continuous media granules that can be dropped or skipped for a given stream.
- Maximum jitter: this is the maximum allowable drift between the constituent streams of the presentation. This drift is the difference between progress of presentation of the fastest and slowest streams.
- Presentation drift: this is the maximum allowable drift between the actual presentation rate and the nominal presentation rate. The absolute value of drift between the individual streams and the nominal presentation rate for the synchronized presentation is taken. The maximum of these values should not at any instant exceed the presentation drift.

The first two QoS parameters pertain to individual streams and the last two pertain to the entire presentation. We define *rendition rate* to be the instantaneous rate of presentation of the constituent streams. When there are no delays, the rendition rate for each stream would be equal to the nominal presentation rate. However, when there are delays, the rendition rate would be slower. It is these delays which cause jitter and presentation drift, resulting in an unsynchronized presentation. In order to reduce the effect of delays, the rendition rate must be increased. But this would entail dropping of media granules. Hence CLF and ALF would increase. The scheduler should be designed to balance this trade off. The same issues hold when the streams are being presented too fast. In this case, the media granules would have to be repeated or paused to slow down the rendition rate. This again causes increase in ALF and CLF.

We will now define a correctness criterion for the execution of a multimedia presentation and then identify those schedules to be considered correct. In this context, the time constraints defined within multimedia presentations assume a position of prime importance. Strictly speaking, the execution of a multimedia presentation $\mathscr{P}$ is correct if the time constraints specified within $\mathscr{P}$ are preserved. However, this semantic correctness criterion is only theoretically applicable to the execution of multimedia presentations. In a practical, delay-prone system, this criterion cannot be applied directly by the scheduler to enforce the execution of multimedia presentations. Given the pervasive nature of delays, a strict application of this rule would result in the aborting of the vast majority of multimedia presentations. A more realistic scheduling criterion is therefore needed. We introduce the concept of *acceptable* schedules by incorporating the effect of delays into the definition of schedules.

Definition 1 (Acceptable execution). *The execution of a multimedia presentation is acceptable if and only if it is within the permissible QoS ranges.*

Note that the presentation scheduler at a client site needs not consider the concurrent execution of multiple multimedia presentations. Rather, it must control the presentation of multiple media streams to a single user. A retrieval scheduler is needed to manage the real time executions of all multimedia presentations at the server site. Adaptability to dynamic client requests is incorporated throughout the design of the NetMedia server. The adaptive features include:

- The system dynamically adjusts its state to best service client requests as specified in the QoS requirements.
- When a client requests a change in the data transfer rate, the server will adjust its policies regarding client scheduling, buffer allocation, and admission control.

We assume that the specification of the QoS parameters and their ranges are given by the users.

3 Synchronization enforcement

In this section, we introduce a framework for the realization, specification, and representation of appropriate synchronization granularity on media streams.

3.1 Synchronization specification

We will first introduce an approach to the specification of synchronization constraints on multimedia presentations. As indicated above, both data and display operations are associated with time constraints. Synchronization constraints may also exist among the display operations of a multimedia presentation. Since synchronization constraints are implicitly imposed by the specification of time constraints, the maintenance of the latter would ideally guarantee the maintenance of the former. However, while the presentation scheduler should make every effort to enforce the time constraints defined on the display operations, even minor delays may create great difficulties in scheduling. Experimental experience demonstrates that such delays are frequent. Thus, the explicit specification and enforcement of synchronization constraints are necessary.

Synchronization dependencies among the display operations in a multimedia presentation can be dynamically generated on the basis of the intra- and inter-synchronization constraints placed on the media streams. Such dependencies are intended to facilitate scheduling by efficiently describing the synchronization constraints existing among the display operations in the multimedia presentation. We define a *synchronization point* to be a point held in common by the display operations within a single multimedia presentation needing to be synchronized. A presentation scheduler must ensure the correct

execution of a multimedia presentation. The time constraints in the execution of a multimedia presentation may differ from what is defined in the multimedia presentation because the dynamic time constraints of the former differ from the static relative time assignments of the latter. Additionally, the tolerance parameters given for the execution of a multimedia presentation will permit further deviations from the multimedia presentation.

Let $\mathscr{P}$ be a multimedia presentation of media streams $m_1, \ldots, m_n$. Each media stream m_i $(1 \leq i \leq n)$ consists of a set of objects o_{ij} $(1 \leq j \leq i_m)$. Without loss of generality, let $\mathscr{P}$ contain s synchronization points. We associate each synchronization point with START event and/or END event, as follows:

- A synchronization point that is a continuous point has both an END event and a START event.
- A synchronization point that is only a starting point of an object has a START event.
- A synchronization point that is only an ending point of an object has an END event.

All START and END events are then classified into synchronization event groups (denoted S-GROUPs) based on the time constraints pertaining to the events. The first S-GROUP$_1$ contains all START events (events at the starting time of the entire presentation), and the last S-GROUP$_n$ contains all END events (events at the ending time of the entire presentation). All events occurring at a given time belong to the same S-GROUP. Thus, each S-GROUP contains all START and END events that must be simultaneously executed.

Note that, in a multimedia presentation, one object o_{ij} may start or end in the middle of another object o_{mn} following the thirteen temporal relationships outlined in [4]. To enforce such synchronization requirements, a START and an END event must be assigned at the beginning and end of o_{ij} and o_{mn}, respectively, and a START and an END event must be assigned in the middle of o_{mn}. In Example 1, seven S-GROUPs, S-GROUP$_1$,..., S-GROUP$_7$, must at least be identified as illustrated in Fig. 3. More synchronization points may need to be enforced on continuous media streams; this issue will be addressed in the following two subsections.

3.2 Media Granularity

We shall now study the sizes of media granules that are generally acceptable to be used for the purpose of enforcing intra-synchronization constraints. In this context, the actual presentation of a media stream is compared with its nominally defined presentation. Deviations between the two presentations must be within an acceptable range in order to guarantee satisfiable presentation of the stream.

Consider a media stream with total presentation time $\mathscr{P}^t$. This media stream could be composed of a number of distinct objects, but for the purpose of analysis, we consider the stream as a continuous one. Suppose we have a synchronization point every s time period. Let the

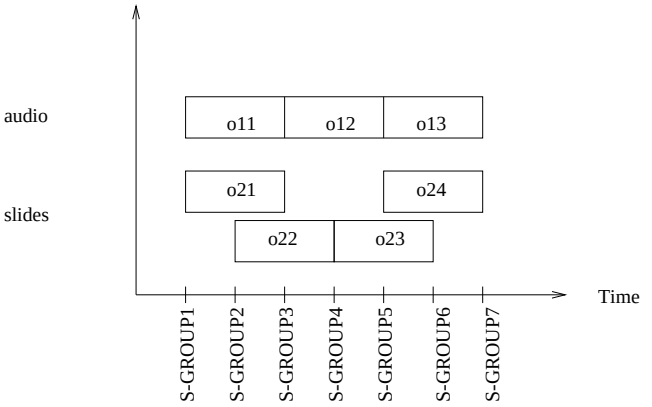

Fig. 3. S-GROUPs

total delay that can be tolerated by the user in the presentation of $\mathscr{P}^t$ time period be τ fraction of $\mathscr{P}^t$. In addition to this, let the average time spent at each synchronization point be δt. This can occur due to computation done at a synchronization point. Finally, let the *system delays* (including the network and server delays) for retrieving and transmitting the $\mathscr{P}^t$ time period of the stream be η fraction of $\mathscr{P}^t$. We then have the system delays for a $\mathscr{P}^t$ time period to be $\eta * \mathscr{P}^t$, and the tolerated delay for $\mathscr{P}^t$ time period to be $\tau * \mathscr{P}^t$. Note that τ and η can be greater than 1. All relevant parameters introduced are listed in Table 1.

We can now determine the lower and upper bounds of the size of media granules. We calculate the data to be dropped per synchronization point as a result of delays. The sum of the synchronization point delay δt and the system delays during every s time period gives the total delay between synchronization points. To ensure the actual presentation compatible to the nominal presentation of the stream, the delayed data must be dropped. Given the system delays and tolerance factors, the total data to be dropped for presentation $\mathscr{P}$ is:

$$\delta t * \mathscr{P}^t / s + \eta * \mathscr{P}^t - \tau * \mathscr{P}^t. \tag{1}$$

To guarantee that the ALF factor is maintained within an acceptable range, the presentation time of the dropped data must not exceed a certain percentage of the entire presentation. This is denoted as d fraction of $\mathscr{P}^t$. Combining this effect with (1), we obtain a lower bound on the value of s:

$$\delta t * \mathscr{P}^t / s + \eta * \mathscr{P}^t - \tau * \mathscr{P}^t \leq d * \mathscr{P}^t, \tag{2}$$

Table 1. Parameters

$\mathscr{P}^t$:	total time of the presentation $\mathscr{P}$.
s:	display time of one media granule.
τ:	fraction of s that can be tolerated due to delay.
δt:	average overhead time spent at each synchronization point.
η:	fraction of s that is delayed by the network and servers.
d:	fraction of $\mathscr{P}^t$ that is tolerable to be dropped.
δd:	the presentation time of the maximum continuous data that can be dropped instantly.

$$s \geq \frac{\delta t}{d + \tau - \eta}. \quad (3)$$

Thus, given the fractions of a presentation that can be dropped and delayed, there exists a lower bound on the size of each media granule, in order to ensure that the computation overhead at each synchronization point and the system delays can be overcome. When the system delays equal the delay tolerated by the user, $s \geq \delta t/d$. Intuitively, when $s = \delta t/d$, it means that every $\delta t/d$ time period we encounter the synchronization point delay $\delta t = d * s$. So every $\delta t/d$ time period, δt time period worth of data must be dropped, resulting in d fraction of the stream not being played. We also note that when $\eta - \tau > d$, no positive value of s can satisfy the condition. Thus, an upper bound on the system delay fraction η is defined:

$$\eta < d + \tau. \quad (4)$$

There is also an upper bound on the size of s. We assume that at every synchronization point, there are two steps, dropping of data depending on the delay in the presentation, and then playing the remaining data. The data that will on average be dropped at each synchronization point is the difference of this delay and the delay that can be tolerated by the user during every s time period. Average presentation time of the dropped data is:

$$\delta t + s * \eta - s * \tau. \quad (5)$$

To guarantee that the CLF factor is maintained within an acceptable range, we introduce another parameter δd to be the presentation time of the maximum continuous data that can be dropped at any given time. When $\tau \geq \eta$, we have $s * \tau \geq s * \eta$. Thus, no synchronization points need be enforced. We now consider $\tau < \eta$. The total data that can be dropped should allow us to recover in the event that the system delays and synchronization point delays exceed the tolerable user delays, so

$$\delta t + s * \eta - s * \tau \leq \delta d. \quad (6)$$

Thus,

$$s \leq \frac{\delta d - \delta t}{\eta - \tau}. \quad (7)$$

Hence we see that if the number of synchronization points are too few, it would not be possible to meet the requirements of the user when the system delays become large. The presentation would only be acceptable to the user when the system delays are smaller than the tolerable delay.

3.3 Synchronization granularity

We shall now study the effects of granularity of synchronization, when inter-stream synchronization is performed. For the purposes of the experiment, we will study synchronization between audio and video streams. We define the *granularity of synchronization* between a set of media streams to be the number of synchronization points that must be identified. Clearly, the finer the synchronization granularity, the more synchronization points will need to be identified. As indicated in Fig. 3, synchronization points should be specified at both beginning and end of each object. However, at the level of multimedia presentation management, the granularity of synchronization can be defined more finely. At this level, additional synchronization points can be defined in the midst of objects to permit finer synchronization control among media streams. We now consider the effect of the number of synchronization points in the presentation of multiple continuous media streams.

Consider the synchronization of video and audio streams. Under normal circumstances, it is the video stream which lags behind the audio stream. This is because video data tends to be larger, and overhead for displaying video frames is more than that for audio. Considering the two streams to be presented without enforcing synchronization, the jitter between the two streams will gradually build up. A small amount of jitter is easily perceived when dealing with audio and video streams, especially in applications where speech is involved. Lip synchronization requires fine grained synchronization between the streams. However, overheads will be encountered at each synchronization point. These would include computational overheads, overheads in using synchronization primitives (semaphores, barriers) and delays due to slower running media streams. Hence, we must keep in balance the synchronization requirements and the restrictions imposed by synchronization overhead.

The granularity of synchronization, in addition to playing a role in inter-stream synchronization, also affects the smoothness of the presentation. Consider the case when the number of synchronization points are few, and one stream is slower than the other. The longer we wait to check the synchronization status of the streams, the larger would be the relative drift between them. This drift may eventually cause the faster stream to pause at the synchronization point. Hence, the lower the number of synchronization points, the larger would be the pauses in the faster stream. The situation is especially bad if it is the audio stream that is faster (which is usually the case). At each synchronization point, the audio stream will have to wait. The human ear is usually very sensitive to the pauses in audio that occur at these points.

We have performed a large number of experiments under varying system loads, speed ratios and media (synchronization) granularity. The video media granularity has been chosen to be 1, 4, 8, and 16 frames. The total presentation lasts for one minute, and audio and video objects are stored in 3 second chunks in separate databases. When the video stream lags behind the audio stream, frames will be selected to drop. MPEG compression is used for the video stream. Rather than skipping a continuous chunk of frames, we select the frames we want to skip from the given video object. For MPEG files, more computation time is saved on the client side by skipping B-frames, as opposed to skipping I-frames or P-frames.

We now present some results which are representative of the effect of various parameters on the presentation

quality. We first consider the case when all video and audio objects are available at the client side without any system delay. A tolerance delay (τ) of up to 15% of the presentation time is allowed. In our implementation, there is an overhead of approximately 15–25% of the presentation time, including parsing and display of the media stream when we enforce synchronization. The processing overhead depends on the amount of load at the client workstation.

As we can see in Table 2, in the case of synchronization granularity with 1 frame, 40% of the video stream has been skipped. This is because δt is 25% of one frame interval. That is, when s is one frame duration, δt is 25% of s. So synchronization at every frame cannot be done when d is less than 40% . The skew is only 25 ms, which shows that the streams are highly synchronized.

For larger synchronization granularities, δt is only 2–6% of s (since s is 4–16 times larger than a frame duration). This should result in lower values of the dropped data. We see that only 5–7% of the stream is skipped. The skews are between 50 and 170 ms. The fewer the number of synchronization points, the larger the skew. In general, to maintain lip synchronization the skew should not be more than the time to speak one syllable. This is typically 250 ms. However, the acceptable value may be even smaller depending on how much pause in the audio can be tolerated by the user.

With no synchronization, the video stream finishes 240 ms after the audio stream, with 0% skip. The jitter between the two streams gradually builds up, with jitter being maximum at the end. We can conclude that when the streams are arriving from the server sides without network delay, it is preferable to have fewer synchronization points, which serve to keep the jitter in check.

We now study the effect of video transmission delays on the presentation. From Table 3, we observe that, with a delay η of 45% , τ of 50% and with synchronization at every frame, 50% of the stream is dropped. For lower number of synchronization points only around 18% of the stream is dropped. Again, this is due to the frame processing overheads. As we increase the value of τ, d increases. We also see that for larger number of synchronization points, the skew is smaller. Similar results are obtained when we keep τ constant and vary η, as seen in Table 4. As suggested by formula (3), when δt is constant, it is only the difference between η and τ which decides the data to be dropped.

The above experiments show that under most values of τ and η, having a synchronization point every 4 frames gives us a reasonable presentation. Optimizations in handling of the video and using hardware decoders could enable us to further lower this limit. The values we have obtained are representative of results that can be expected for any synchronized presentation of continuous media streams. A good balance between s and τ can be used to achieve reasonable presentations under conditions of delay.

Table 2. Presentation with $\eta = 0$

s (frames)	d (%)	*Skew* (ms)
1	40	25
4	7	54
8	6	86
16	5	170

Table 3. Presentation with $\eta = 45\%$

s (frames)	τ (%)	d (%)	*Skew* (ms)
1	30	60	11
1	40	56	14
1	50	51	18
4	30	33	14
4	40	26	58
4	50	18	73
8	30	33	84
8	40	26	112
8	50	17	144
16	30	38	147
16	40	28	222
16	50	18	283

4 Scheduling algorithms

In this section, we will investigate the generation of acceptable schedules. We will first present a basic scheduling strategy for synchronized presentations of multiple streams. We will then discuss an advanced scheduling strategy which can enhance the smoothness of continuous media presentations.

4.1 Basic scheduling

In the presentations of non-continuous media streams, such as transparencies and animation, abruptly skipping or pausing the presentation materials can be acceptable. Thus, relatively simple algorithms can be designed to efficiently handle the presentation scheduling of these streams. In this context, the S-GROUPs specified within

Table 4. Presentation with τ of 40%

s (frames)	η (%)	d (%)	*Skew* (ms)
1	45	56	11
1	55	58	11
1	65	63	9
4	45	26	58
4	55	35	48
4	65	47	40
8	45	26	112
8	55	39	98
8	65	47	79
16	45	28	222
16	55	40	196
16	65	49	171

each multimedia presentation summarize the most critical scheduling information. In our basic scheduling strategy, the scheduler ensures that only acceptable schedules will be generated by controlling the invocation order of events in the formulated S-GROUPs of each multimedia presentation.

Let a multimedia presentation $\mathscr{P}$ have n S-GROUPs. To incorporate the ALF and CLF factors into our scheduling strategy, we assume that each media stream has a permissible delay constraint and that the minimum value of all permissible delay constraints given in the media streams defines the maximum tolerable delay for the multimedia presentation. If the scheduler finds that it has been waiting too long for the completion of a display operation, then it aborts the multimedia presentation. Let two parameters, *maximum tolerable delay*, denoted Δd_i^{max}, and *maximum skip*, denoted Δs_i^{max}, at any point of the stream, be specified for each media stream m_i. If maintenance of good utilization is of highest interest in a particular instance, then the amount that can be skipped should be specified as a relatively small figure. If it is more important to minimize the average delay, then the delay allowed for the media stream should be set at a relatively low level. In our context, users may have different expectations for various presentations of learning materials. Thus, the choice of QoS parameters may vary in different stream presentations. Within the given QoS parameters, the basic scheduling will maximize utilization and minimize the abortion rate of multimedia presentations in order to preserve presentation quality.

Consider a presentation of media streams $m_1, \ldots, m_n$ and a set of media objects from these streams which are currently synchronously displayed. Inter-stream synchronization causes problems which are not present during synchronization of single streams. It may be possible to achieve an acceptable presentation for a single stream, but the overall presentation may not be acceptable when all the streams are synchronized together. As we have seen above, due to delays in the video stream, the resulting pauses in the audio stream may lead to audio which is unrecognizable to the listener. In such cases, it may be better to discard the video altogether, and playback only the audio. This can be done for applications where the video information is less. An example of this is a lecture. It may be possible to display only the audio, and the slides which are used during the lecture. Here we observe the advantage of being able to store each stream separately. Only the streams which must be presented have to be retrieved and displayed.

We must try to minimize the amount of time the audio stream waits at a particular synchronization point. Let Δd_i $(1 \leq i \leq n)$ denote the delay that occurs in the object belonging to m_i. We calculate *PAUSE* to be the pause period that would occur to the audio stream because of delays in the video stream. We could then determine how much part of the video stream we must skip to minimize this *PAUSE*:

$$PAUSE = \Delta d_v - \Delta s_v^{max}, \tag{8}$$

where the v subscript stands for video. This is the case when the video stream lags behind the audio stream. We treat a group of frames as a video object, and the sum of presentation times of the skipped frames must not exceed Δs_v^{max}. Rather than skipping a continuous chunk of frames, we select the frames we want to skip from the given video object. For MPEG files, more computation time is saved on the client side by skipping B-frames, as opposed to skipping I-frames or P-frames. This is especially true when we use a software MPEG decoder. In addition to the savings in computation time, a smoother overall presentation can be achieved. If the video object consists of only one frame, then subsequent objects may be skipped.

We must note that the value for *PAUSE* remains the same whether the delays are caused due to processing delays or system delays. This value for *PAUSE* is used both by the client and the server for delay recovery. In the case that delay is caused by network bandwidth, it would be better to drop macroblocks from frames such that the transmission time for the video object is reduced by at most Δs_v^{max}. This is because just dropping B-frames does not reduce transmission time of the video object greatly.

The specification for *PAUSE* is a very generic one, but the delay recovery implementation depends on the particular circumstance. The delay Δd_v is calculated relative to the time scale for the presentation. When a particular value for τ is specified, this time scale is slowed down by $\tau\%$. At any point, Δd_v is the delay of the video stream relative to this time scale. Whenever this value is positive, it means that the video stream is lagging behind the tolerable presentation. At this point the video stream is skipped. More precisely, Δd_v^{max} is zero when delay is measured relative to the time scale. When τ is larger, fewer skips would be required in the video presentation.

The invocation procedure for the presentation of $\mathscr{P}$ includes the following rules; assume that all START events in S-GROUP$_1$ have been invoked:

(1) The events in S-GROUP$_{i-1}$ always have a higher invocation priority than those in S-GROUP$_i$ for any i such that $1 < i \leq n$.
(2) All START events in a S-GROUP$_i$ $(1 \leq i \leq n)$ are invoked simultaneously.
(3) All END events in a S-GROUP$_i$ $(1 \leq i \leq n)$ are terminated simultaneously.
(4) All START events in a S-GROUP$_i$ $(1 \leq i \leq n)$ can only be invoked after all END events in the same S-GROUP have terminated and if PAUSE (defined in formula (8)) > 0, pause the START events in S-GROUP$_i$ for PAUSE period before invocation.

The basic scheduling strategy guarantees that minimal deviation between streams will occur within the presentation at synchronization points. Synchronization is thus enforced by controlling the invocation of START events. Both jitter and skew are thus minimized. Delays that may occur between the synchronization points are recovered according to the permissible delays and skips. Thus, the utilization value may not be equal to one, but it will lie within the permissible QoS range. Thus, this approach generates only acceptable schedules.

Experiments have been conducted following the basic scheduling strategy. Although the jitter and skew are minimized, hiccups can be observed at some synchronization points for the presentations of continuous streams. This is because abruptly skipping or pausing the presentation materials takes place at these synchronization points for recovery from delays. When an audio stream is synchronously displayed with a slides stream, such hiccups become less visible. Even if the hiccups are visible, they are mostly tolerable. However, when an audio stream is displayed with a video stream, such hiccups may not be acceptable. We will discuss a scheduling strategy for more smooth presentations of the continuous streams.

4.2 Advanced scheduling

Many educational digital library applications require delicate synchronization between continuous streams with skew limited at 80 ms. For example, a video stream of medical surgery integrated with an audio stream of explanation requires precise synchronization between what are seen and heard. An extreme case is the *lip synchronization* which defines the temporal relationships between an audio and visual streams for the particular case of human speaking [41]. For such streams, experimental experience has demonstrated that abrupt skipping or pausing can result in presentation hiccups. Such hiccups may dramatically degrade presentation effectiveness, especially for lecture presentations where lip synchronization must be maintained.

We have observed that more robust scheduling algorithms can be designed for the synchronized presentations of continuous streams. In such algorithms, the recovery of a presentation from asynchrony is accomplished by gradually accelerating or retarding presentation components, rather than abruptly skipping or pausing the presentation. During the presentation of the streams, each stream must report to the presentation scheduler its current progress. The presentation scheduler in turn reports to each stream the rendition rate required to maintain the desired presentation. The individual streams must try to follow this rendition rate.

Let the nominal rendition rate for a stream be expressed as the number of stream granules per unit time. For example, the nominal rendition rate for video could be 30 frames per second, with one frame comprising a granule. To meet this scheduling requirement, the time period between consecutive frames must be 1/30 s. Intra-stream synchronization can thus be achieved by displaying the frames at this rate. However, due to various factors affecting system load, it may not always be possible to achieve this rate. We must therefore specify a reasonable range within which display operations will fall.

To incorporate the ALF, CLF and presentation drift factors into our scheduling strategy, we consider two parameters in the presentation of media stream m_i, maximum rate change c_{m_i} and maximum instantaneous drift d_{m_i}. Let the nominal rendition rate for m_i be denoted by $R^n_{m_i}$. Let $R_{m_i}(t)$ be the rendition rate of m_i at a time instant t. The maximum rate change c_{m_i} represents the range within which the rendition rate of media stream m_i can vary:

$$R^n_{m_i} * (1 - c_{m_i}) \leq R_{m_i}(t) \leq R^n_{m_i} * (1 + c_{m_i}). \tag{9}$$

We define *progress* $p_{m_i}(t)$ of media stream m_i at time t as the presentation time of granules displayed so far, assuming the nominal presentation rate. Let the presentation of media stream m_i is started at time t_0. Stream m_i is said to be progressing at the nominal rate if, at time t, the granules displayed to date are $R^n_{m_i} * (t - t_0)$. Stream m_i is progressing at the nominal rate if, at time t, the progress measured in time is also $t - t_0$. This is given by dividing the granules displayed to date ($R^n_{m_i} * (t - t_0)$ for nominal rate) by $R^n_{m_i}$. We can express the progress in terms of time by dividing the granules displayed by $R^n_{m_i}$. If the progress is either less or greater than $t - t_0$, then the presentation is either below or above the nominal rate. The presentation is at the nominal rate if the progress is equal to $t - t_0$. The nominal length of presentation time of each granule in m_i is $1/R^n_{m_i}$. If, at time t, the kth granule of m_i is being displayed and the display of this granule started at time $t - \Delta t$, the progress of m_i is calculated by

$$p_{m_i}(t) = (k-1)/R^n_{m_i} + \Delta t * R_{m_i}(t - \Delta t)/R^n_{m_i}. \tag{10}$$

The maximum instantaneous drift d_{m_i} at any given time denotes the maximum time difference between the progress of the stream and the progress assumed under the nominal presentation rate. If the stream started at time t_0, we should then have

$$|p_{m_i}(t) - (t - t_0)| \leq d_{m_i}. \tag{11}$$

We assume that the user can specify an inter-stream jitter j in order to enforce synchronization requirements. The inter-stream synchronization requirements can be met if, for each synchronization points at time t and all streams m_i and m_k in the presentation,

$$|p_{m_i}(t) - p_{m_k}(t)| \leq j. \tag{12}$$

The central component in the advanced scheduling algorithms is the determination of the rendition rates for streams at each synchronization point such that the three conditions given in (9), (11), and (12) are satisfied. Both intra-stream scheduling on individual media streams and inter-stream scheduling on the synchronized media streams must be considered.

Intra-stream scheduling. Intra-stream scheduling must be performed so that the constraints imposed by conditions (9) and (11) are met. We enforce intra-stream constraints by determining the progress of the stream at set intra-synchronization points and assigning a value to $R_{m_i}(t)$ such that the constraints are met at the next point. The intra-synchronization points occur at regular intervals Δt_s. The rendition rate is set at time t and remains constant until time $t + \Delta t_s$. If there is an intra-synchronization point at time t and a rendition rate $R_{m_i}(t)$ has been calculated, then we can determine the progress expected at time $t + \Delta t_s$. If, at time t, the kth granule is being displayed and the display of this granule was started at time $t - \Delta t$, we must first calculate the display time for this granule using the old rendition rate:

$$t_{rem} = max(0, 1/R_{m_i}(t - \Delta t) - \Delta t). \quad (13)$$

It is possible that, with the rate of $R_{m_i}(t - \Delta t)$, the granule should have finished by the current time t. This would give a negative value for $1/R_{m_i}(t - \Delta t) - \Delta t$. In this case, t_{rem} would be 0. Here, we assume that, by the time we assign a new rendition rate, the presentation of the current granule would be complete. Once the current granule has been completed, the remaining time until the next intra-synchronization point can be utilized to display the stream at the new rate. The progress at next synchronization point can be determined and we can also get the range for $p_{m_i}(t + \Delta t_s)$ to satisfy the intra-stream constraints.

The selection of a progress rate at the next intra-synchronization point can be guided by a variety of approaches. We may minimize rate fluctuations by selecting a progress rate which is closest to the nominal rate. Alternatively, a progress rate may be chosen which would result in the least deviation from $t + \Delta t_s$. The actual progress made may, in fact, be different from the calculated value. Depending on the current machine and network loads and on other factors. We must rely on a history-based heuristic as a guide to estimating progress. In a practical implementation, it would be necessary to check the actual progress at each intra-synchronization point. If the intra-stream synchronization requirements are not met at a sufficiently large number of intra-synchronization points, recovery actions must be taken by sending requests to the servers.

Inter-stream scheduling. We will now incorporate the inter-stream scheduling on the synchronized media streams into the determination of rendition rates for media streams. We assume that the user can specify an inter-stream jitter j in order to enforce synchronization requirements. Suppose we have n streams, then, at each synchronization point, a value $R_{m_i}(t)$ must be determined such that the intra-stream constraints of each stream are met, and the inter-stream constraint imposed by 12 must also be met. We can formally specify the inter-stream synchronization constraint in terms of the progress of the streams. Without loss of generality, let inter-synchronization points occur at regular intervals Δt_s. We define

$$p_{s_{max}}(t + \Delta t_s) = MAX(p_{m_i}(t + \Delta t_s)), \quad (14)$$

$$p_{s_{min}}(t + \Delta t_s) = MIN(p_{m_i}(t + \Delta t_s)), \quad i = 1, \ldots, n. \quad (15)$$

The inter-stream synchronization requirements can be met if for all synchronization points at $t + \Delta t_s$,

$$p_{s_{max}}(t + \Delta t_s) - p_{s_{min}}(t + \Delta t_s) \leq j, \quad (16)$$

where t denotes the time at which the current synchronization point begins.

Based on the above discussion, various algorithms can be designed to calculate rendition rates for the streams at a synchronization point occurring at time t, satisfying both intra- and inter-stream constraints. We have investigated two algorithms [26]. The first algorithm determines $R_{m_i}(t)$ for all the streams $i = 1, ..., n$ (within constraints of c_{m_i} and d_{m_i}) so that jitter is minimized at the next synchronization point. In other words, the algorithm must minimize $p_{s_{max}}(t + \Delta t_s) - p_{s_{min}}(t + \Delta t_s)$. This value must be less than or equal to j for the inter-stream constraint to be met. The second algorithm attempts to meet user requirements while holding the streams close to their nominal presentation rate. Any change from the nominal presentation rate may result in presentation drift for the presentation of the stream. For a given value of j, the algorithm determines rendition rates R_{m_i} for each stream m_i with nominal presentation rate $R^n_{m_i}$ such that the sum of the individual stream rate changes $\sum_{i=1}^{n} \frac{|R_{m_i}(t) - R^n_{m_i}|}{R^n_{m_i}}$ is minimized. By minimizing this sum, the change of rendition rate in all the streams combined is minimized.

The operation of these algorithms is presented graphically in Fig. 4, which depicts the differential progress among three streams at a synchronization point. The solid box depicts the progress to be made by media stream m_i if the rendition rate is limited by c_{m_i}. The dashed box depicts the progress made using rendition rates which satisfy d_{m_i}. Areas of intersection are shown in dark values. The area of intersection covers the progress range between the highest of the low-end progress values (denoted $p_{s_{left}}(t + \Delta t_s)$ in the figure) and the lowest of the high-end values (denoted $p_{s_{right}}(t + \Delta t_s)$ in the figure). No intersection is possible if the former progress values are greater than the latter. Using the calculation of the progress range given in the intra-stream scheduling, we can obtain the intersection of the ranges of all participating streams. At any point within this area of intersection, the

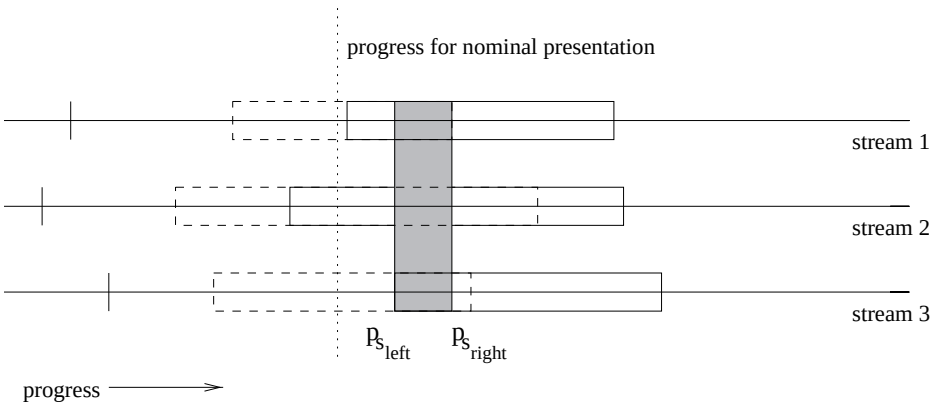

Fig. 4. A presentation of streams

intra-stream constraints for all three streams will be met. If no area of intersection exists, the streams may be scheduled, but some jitter will be introduced. Thus, the two cases of $j = 0$ and $j \neq 0$ should be treated separately. The detailed discussion of these cases can be found in [26].

To implement the above framework, the client must have several threads running concurrently to achieve the presentation. Data from the server are collected by the network threads and inserted into separate buffers for each media stream. In addition to the actual data to be presented, additional information such as the sequence number of the media granule should be provided to the individual stream threads. The individual streams use this information to inform the presentation scheduler thread about the progress information. The individual threads present the media data at the rate specified by the scheduler.

Experiments have been conducted to analyze the performance of the proposed algorithms. For example, for the first algorithm scheduling inter-stream synchronization, jitter between streams was plotted against time with synchronization points at a 250 ms interval. This plot indicates a minimal level of jitter between streams. Audio and video stream rates were then superimposed; These vary between $\pm 20\%$. The CPU was heavily loaded at three points, as indicated in Fig. 5. While there is a momentary increase in jitter, the streams quickly return to synchronization. Under a load, the video stream is retarded, since the video decoding process requires significant CPU time. To compensate, the algorithm slows the audio stream and accelerates the video stream, thus reducing the jitter to zero. The user perceives only a momentary loss of synchronization, with no hiccups within the streams. Since this algorithm reduces jitter and minimizes change in presentation rate, good synchronization and a constant presentation rate are both maintained.

The proposed algorithms permit the local adjustment of unsynchronized presentations by gradually accelerating or retarding presentation components, rather than abruptly skipping or pausing the presentation materials. The experiments have demonstrated that these algorithms can avoid presentation hiccups. The smooth presentations generated by these algorithms is of particularly importance to applications of education and training. For example, the smoothness and synchronization of a lecture presentation play an important role for the students to understand the materials. Other synchronization algorithms can also be designed based on the

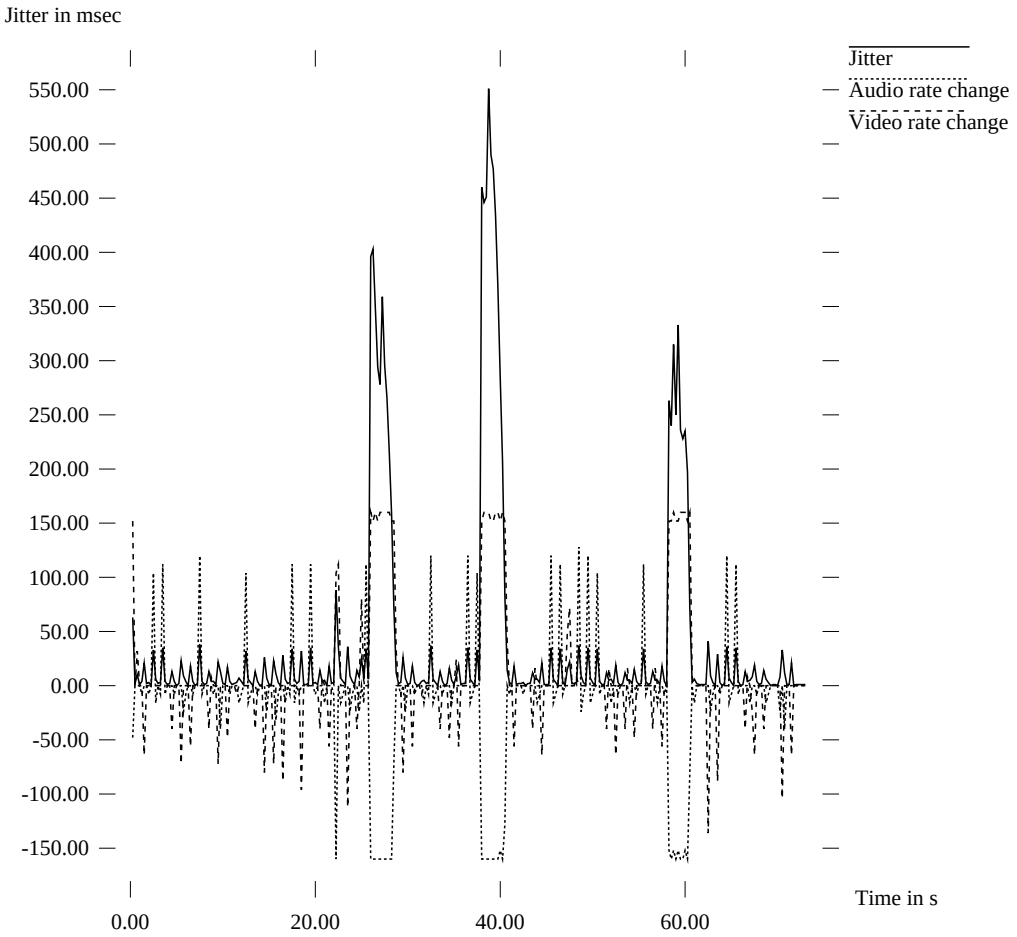

Fig. 5. Jitter in a loaded system

above framework, considering different tradeoffs among jitter, presentation drift, aggregate loss, and cumulative loss.

Note that this framework is proposed for the scheduling at the client side. We assumed that the server provides sufficient support for displaying media data and is adaptable to the various requests from the client side. The design of the server to support the multimedia presentations at the client sides will be provided in the next section.

5 Design of an adaptive server

In this section, we will discuss an adaptive approach to admission control and scheduling server service time. The purpose of the section is to demonstrate that an adaptive server can be designed to support the presentation algorithms proposed at the client side. Also, dynamic client requests of changes in the data transfer rates are supported. As the primary focus of this paper is on the presentation scheduling at the client side, some other issues are not addressed here.

The server is designed to handle multiple clients. To exploit the periodic nature of media streams, the requests from these clients are handled in a round robin fashion. Each round r_i is divided into slots s_{ij} ($j = 1, \ldots, n$), where n is the number of clients serviced by the server. A client request to a server identifies a set of media objects that the client wishes to retrieve. These objects are stored in the local DBMS. It is the responsibility of the server to identify all the relevant objects and load them into the server buffer before transmitting them to the client. The time period of a round determines the latency for display of objects to the client. The larger the round, the more is the latency.

The admission controller handles connection requests from the client and admits the client if enough resources are available. The admission control policy assumes that there is sufficient network bandwidth to transfer the requested data. A network admission controller would be needed to perform this function. A network manager to address the admission control in Ethernet can be found in [7]. This issue will not be addressed in this paper.

The client request R_i specifies the amount of data x that it requires in each round for m rounds and the tolerated degradation α in this amount. For the given disk retrieval rate r, the service time required to retrieve x can be determined to be:

$$t = (\alpha * x)/r.$$

A strict servicing policy would set α to 1, thereby requiring the server to meet all the deadline requests of the client. A flexible request would have a value in the range [0,1], suggesting that the server can drop some data if it cannot meet a deadline requirement or the client requests a change in the data sending rate. In such a case, the server can also delay the transmission of data so that the new deadline requirements are within the QoS specifications set by the client. Thus, α sets the fraction of data which can be dropped (such as frames in a video clip) or the permission time through which the server can delay the transmission of the data.

Without loss of generality, let the duration of each round be T and t_{ij} be the duration of client request R_i in round r_j. The admission of a new client request to the server is conservatively approached. Let the current round be r_k, servicing n current client requests $R_1, \ldots, R_n$. We consider the addition of client request R_{n+1} to the server starting at r_{k+1} for the next m rounds. For all rounds r_j, where $k+1 \leq j \leq k+m$, if $\Sigma_{i=1}^{n+1} t_{ij} \leq T$, then the new request R_{n+1} is admitted to r_{k+1}, assuming that buffer consumption is less than the maximum buffer space [22]. Otherwise, R_{n+1} cannot be admitted to r_{k+1}.

We now discuss the scheduling of service time to retrieve data from disk. Following the admission set up, the scheduler also handles the client requests in a round robin fashion, servicing each client in the slots specified by the admission controller. During each slot, the retrieval of data from disk is performed and loaded into the server buffer. In addition, the scheduler must also be able to dynamically respond to the client requests of changes in the data transfer rates over the network. The scheduler achieves such adaptability by dynamically adjusting the slots in each round for current client requests. However, the assigned slots to the client in the m rounds should always be owned by this client to guarantee a flexible service for the client. Even after the client request has been degraded by α, the degradation can be recovered at any time the client asks for. Otherwise, the client may never be able to be recovered if sufficient slots is no longer available. More specifically, the scheduler handles messages which are sent by the client as follows:

- Connection termination: notify the admission controller.
- Increase rendition rate request: if the allocated slots for the client are not fully used, increase the utilization of the given slots; otherwise, drop media granules to obtain faster transmission.
- Reduce rendition rate request: decrease the utilization of the given slots; the extra idle slots may be temporarily used for retrieving extra data.

The requests to increase or decrease the rendition rate provide information which can be used for appropriate slot manipulation. In normal operation, network delays will not happen (since bandwidth is allocated during admission). But this may occur due to unforeseen circumstances. The system should be able to recover in this case.

6 Conclusions

In this paper, we have developed a scheduling theory for supporting multimedia presentations in educational digital libraries. This scheduling theory includes the specification and representation of synchronization on media streams, the realization of appropriate synchronization granularity, and the scheduling principles for the presentation of multimedia streams. We have formulated criteria for specifying and scheduling the skipping/

pausing of media streams with asynchronous presentations when various delays occur. Adaptability to various quality-of-service requirements are supported in the scheduling theory. Various synchronization mechanisms at both client and server sides are then proposed to implement the framework. Experimental analysis are conducted using instructional materials.

This framework can be readily used to facilitate various education and training applications. A computer-assisted learning environment can thus be established which will greatly enhance the available range of teaching techniques to facilitate student training, human resource development, and asynchronous distance learning.

References

1. 2nd International Workshop on Multimedia Information Systems. West Point, New York, September 1996
2. Proc. of the 1996 International Workshop on Multi-media Database Management Systems. Blue Mountain Lake, NY, August 1996
3. G. Abowd, C. Atkenson, A. Feinstein, C. Hmelo, R. Kooper, S. Long, N. Sawhney, M. Tani. Teaching and learning as multimedia authoring: the classroom 2000 project. In: Proc. of ACM Multimedia 96, pp. 187–198. Boston, MA, 1996
4. J. Allen. Maintaining knowledge about temporal intervals. Communications of ACM 26(11), 1983
5. D.P. Anderson, George Homsy. A communication media I/O server and its synchronization mechanisms. IEEE Computer 24(10), 51–57, 1991
6. R.T. Aptekar, J.A. Fisher, V. Kisimov, H. Nieshlos. Video acceptability and frame rate. IEEE Multimedia 3(3), 32–40, 1995
7. V. Balachandran, G. Bhat, S. Chakravarty, A. Zhang. A network manager on ethernets for distributed multimedia systems. In: Proc. of the 22nd Annual IEEE Conference on Computer Networks, Minneapolis, MN, November 1997
8. S. Boll, W. Klas, M. Lohr. Integrated database services for multimedia presentations. In: Soon M. Chung, (ed.) Multimedia Information Storage and Management, pp. 399–448. Boston, MA, Kluwer Academic, 1996
9. J.F. Koegel Buford. Multimedia Systems. Addison-Wesley, Reading, MA, 1994
10. K. Selcuk Candan, B. Prabhakaran, V. Subrahmanian. Chimp: a framework for supporting multimedia document authoring and presentation. In: Proc. of ACM Multimedia 96, pp. 329–340. Boston, MA, 1996
11. K. Selcuk Candan, V.S. Subrahmanian, P. Venkat Rangan. Towards a theory of collaborative multimedia. In: Proc. of the 3rd IEEE International Conference on Multimedia Computing and Systems (ICMCS'96), pp. 279–282, Hiroshima, Japan, June 1996
12. C.Y.R. Chen, D.S. Meliksetian, M.C.-S. Chang, L.J. Liu. Design of a multimedia object-oriented DBMS. Multimedia Systems 3(5/6), 217–227, 1995
13. S. Christodoulakis, N. Ailamaki, M. Fragonikolakis, Y. Kapetanakis, L. Koveo. An object oriented architecture for multimedia information systems. IEEE Data Engineering 14(3), 1991
14. Y. Deng, S.K. Chang. A framework for the modeling and prototyping of distributed information systems. Int. J. Software Engineering and Knowledge Engineering 1(3), 203–226, 1991
15. M. Lucia Escobar-Molano. Management of resources to support continuous display of structured video objects, 1994. Technical Report, Department of Computer Science, University of Southern California
16. B. Furht. Multimedia Systems and Techniques. Boston, MA, Kluwer Academic, 1996
17. B. Furht. Multimedia Tools and Applications. Boston, MA, Kluwer Academic, 1996
18. D.J. Gemmell, H.M. Vin, D.D. Kandlur, P.V. Rangan. Multimedia storage servers: a tutorial and survey. Computer 28(5), 40–51, 1995
19. A. Ghafoor. Special issue on multimedia database systems. Multimedia Systems 3(5/6), 1995
20. S. Ghandeharizadeh, Luis Ramos. Continuous retrievals of multimedia data using parallelism. IEEE Transactions on Knowledge and Data Engineering 5(4), 658–669, 1993
21. S. Gibbs, Christian Breiteneder, Dennis Tsichritzis. Data modeling of time-based media. In: Proc. of the ACM-SIGMOD International Conference on Management of Data, pp. 91–102, Minneapolis, MN, May 1994
22. S. Gollapudi, A. Zhang. Buffer management in multimedia database systems. In: Proc. of the 3rd IEEE International Conference on Multimedia Computing and Systems (ICMCS'96), Hiroshima, Japan, June 1996
23. S. Gollapudi, A. Zhang. NetMedia: A client-server distributed multimedia database environment. In: Proc. of the 1996 International Workshop on Multi-media Database Management Systems. Blue Mountain Lake, NY, August 1996
24. V. Hakkoymaz, G. Ozsoyoglu. Automating the organization of presentations for playout management in multimedia databases. In: Proc. of the 1996 International Workshop on Multi-media Database Management Systems. Blue Mountain Lake, NY, August 1996
25. V. Hakkoymaz, G. Ozsoyoglu. A constraint-driven approach to automate the organization and playout of presentations in multimedia databases. Int. J. Multimedia Tools and Applications 4(2), 171–198, 1997
26. T. Johnson, A. Zhang. A framework for supporting quality-based presentation of continuous multimedia streams. In: Proc. of the 4th IEEE International Conference on Multimedia Computing and Systems (ICMCS'97), Ottawa, Canada, June 1997
27. T.L. Kunii, Y. Shinagawa, R.M. Paul, M.F. Khan, A.A. Khokhar. Issues in storage and retrieval of multimedia data. Multimedia Systems 3(5/6), 298–304, 1995
28. T.D.C. Little, A. Ghafoor. Network considerations for distributed multimedia object composition and communication. IEEE Network, pp. 32–49, 1990
29. T.D.C. Little, A. Ghafoor. Synchronization and storage models for multimedia objects. IEEE Journal on Selected Arears in Communications 8(3), 413–427, 1990
30. Y. Masunaga. Design issues of OMEGA: an object-oriented multimedia database management system. J. Information Processing 14(1), 60–74, 1991
31. F. Moser, Achim Kraib, Wolfgang Klas. L/MRP: a buffer management strategy for interactive continuous data flow in a multimedia DBMS. In: Proc. of the 21st VLDB Conference, Zurich, Switzerland, 1995
32. K. Nahrstedt. End-to-End QoS guarantees in networked multimedia systems. ACM Computing Surveys 27(4), 1995
33. K. Nahrstedt, J. Smith. New algorithms for admission control and scheduling to support multimedia feedback remote control applications. In: Proc. of the 3rd IEEE International Conference on Multimedia Computing and Systems (ICMCS'96), pp. 532–539, Hiroshima, Japan, June 1996
34. K. Nahrstedt, R. Steinmetz. Resource management in networked multimedia systems. IEEE Computer 28(5), 52–64, 1995
35. G. Ozsoyoglu, V. Hakkoymaz, J. Kraft. Automating the assembly of presentations from multimedia databases. In: Proc. of the 12th Intl. Conf. on Data Engineering, New Orleans, LA, February 1996
36. T.C. Rakow, M. Lohr. Audio support for an Object-Oriented Database-Management System. Multimedia Systems, 3(5/6), 286–297, 1995
37. P. Venkat Rangan, Srinivas Ramanathan, Thomas Kaeppner. Performance of inter-media synchronization in distributed and

heterogeneous multimedia systems. Computer Networks and ISDN Systems, 1993
38. P. Venkat Rangan, Harrick M. Vin. Efficient storage techniques for digital continuous multimedia. IEEE Transactions on Knowledge and Data Engineering 5(4), 564–573, 1993
39. R. Staehli, J. Walpole, D. Maier. A quality-of-service specification for multimedia presentations. Multimedia Systems 3(5/6), 251–263, 1995
40. R. Steinmetz. Synchronization properties in multimedia systems. IEEE Journal on Selected Arears in Communications 8(3), 401–412, 1990
41. R. Steinmetz, K. Nahrstedt. Multimedia: Computing, Communications and Applications. Prentice Hall, Englewood Cliffs, NJ, 1995
42. P.D. Stotts, R. Furuta. Petri-net-based hypertext. ACM Transactions on Office Automation Systems 7(1), 1989
43. H. Thimm, W. Klas. Delta-Sets for optimized reactive adaptive playout management in distributed multimedia database systems. In: Proc. of the 12th Intl. Conf. on Data Engineering, New Orleans, LA, February 1996
44. H. Thimm, W. Klas, C. Cowan, J. Walpole, C. Pu. Optimization of adaptive data-flows for competing multimedia presentational database sessions. In: Proc. of the 4th IEEE International Conference on Multimedia Computing and Systems (ICMCS'97), pp. 328–335, Ottawa, Canada, June 1997
45. H. Thimm, W. Klas, J. Walpole, C. Pu, C. Cowan. Managing adaptive presentation executions in distributed multimedia database systems. In: Proc. of the 1996 International Workshop on Multi-media Database Management Systems. Blue Mountain Lake, NY, August 1996
46. F.W. Tompa. A data model for flexible hypertext database systems. ACM Transactions on Information Systems 7(1), 1989
47. A. Vogel, B. Kerherve, G. von Bochmann, J. Gecsei. Distributed multimedia and QoS: a survey. IEEE Multimedia 2(2), 10–19, 1995
48. D. Wijesekera, J. Srivastava. Quality of Server (QoS) metrics for continuous media. Int. J. Multimedia Tools and Applications, pp. 127–166, September 1996
49. L. Zhang, S. Deering, D. Estrin, S. Shenker, D. Zappala. RSVP: A new resource reservation protocol. IEEE Network, pp. 8–18, 1993
50. T.F. Znati, B. Field. A network level channel abstraction for multimedia communication in real-time networks. IEEE Transactions on Knowledge and Data Engineering 5(4), 590–599, 1993