Maintaining Temporal Consistency of Multimedia Documents Using Constraint Networks

Nabil LAYAÏDA
Loay SABRY-ISMAIL

Opera project, INRIA Rhône-Alpes.
2, Rue Vignate, Gières 38610, France.
e-mail : { Nabil.Layaida, Loay.Sabry }@imag.fr

1 INTRODUCTION

The recent advances in multimedia systems, together with the advent of high speed networks, paved the way to a new generation of applications. In particular, the authoring environments have found in multimedia the means of increasing the richness of information contained in electronic documents. With the evolution of new computer systems that can handle multimedia information, time-based data can be integrated in electronic documents taking into account their temporal dimension. In such documents, temporal dependencies between the different media objects define a temporal structure within the document. This structure is the basic support for the representation of dependencies between data such as audio, video and virtual images. Furthermore, it allows the scheduling of presentation actions during the document presentation.

The presentation of multimedia documents is dynamic and the positioning of objects in time together with their duration have to be specified. To achieve this operation efficiently, a high level temporal representation is needed which allows the author to specify all the temporal dependencies between multimedia objects.

In this paper, we propose an interval-based temporal model and constraints which provide a basis for the management of the consistency of multimedia documents. We propose an efficient algorithm allowing the detection of a wide range of inconsistencies. The emphasis in the design of these algorithms is put on the handling of both the flexibility of temporal specifications and the indeterministic behaviour of some media objects. Furthermore, we use the logical organization of the document in nested entities to enhance the performance of the methods used for detecting inconsistencies. The aim of our approach is to fulfill the following requirements :

- **Structured modelling:** The document content is defined by a hierarchy of nested components where leaves are basic media objects and nodes composite objects.
- **Incremental manipulation of the document:** This means that the author adds one constraint at a time, retaining all previously introduced relations as the current state of the document.
- **Consistency checking:** For every specification introduced by the author, the system checks if there exists a solution to the set of constraints, otherwise an error is reported to the author.
- **Integration of the different dimensions of multimedia documents:** In our model, we focus on four dimensions of the document (logical, temporal, spatial and hyperlink). These dimensions are closely related

and have to be considered simultaneously as they may interfere. In this paper, we will focus on the temporal and logical dimensions of the document but we will describe briefly how this fits with the spatial one.

- **Automatic production of temporal layout:** The author specifies relations using a high-level language to describe the document temporal layout, and does not have to explicitly set the low-level synchronization dates.

In section 2, we present our model of document composition showing the logical, spatial, and temporal dimensions. In section 3, we describe how we can create and manipulate the internal representation of the multimedia document in the form of constraint networks. In section 4, a consistency checking algorithm is proposed, followed by a discussion. Finally, in section 5, we conclude our work and give some future directions.

2 DOCUMENT COMPOSITION MODEL

A temporal model is a set of abstractions that allows users to specify temporal constraints in multimedia documents. This model consists of a given representation of time (a time model), a temporal expression that defines a temporal language, and some capabilities or mechanisms for automatically processing the temporal information contained in multimedia documents.

The basic parameter of a temporal relation is an interval, which is an abstraction of a media object. This interval has a non-zero duration and is described in the time space by two instants or end points (begin, end). This duration can be an intrinsic information of the media object or dependent on some constraints in the context of a given document. In our model, we distinguish between three kinds or classes of basic media objects regarding their temporal behaviour[10] :

1. **Discrete media objects** which do not have a notion of time. The presentation of these objects is not bound to temporal flow constraints. Examples are text, still pictures and graphics.
2. **Continuous media objects** which have a notion of time. The presentation of these media objects is bound to a certain playback rate. Therefore, their duration can be known at the stage of authoring. Examples are video and audio.
3. **Indeterministic media objects** (discrete or continuous) which have an indeterministic presentation duration. The presentation of this kind of media objects will lead to several durations when presented several times. Examples are user interactions, objects obtained as a result of the executions of programs like database queries (discrete) or computed synthetic images produced during the presentation (continuous).

A multimedia document is a mixture of sub-components which are related by several types of relationships. These various natures of bindings confer to the document a multi-dimensional skeleton which reflects its spatial, temporal, logical and semantical (hyperlinking) structure.

2.1 Logical Composition

In the context of a given document, objects are logically organized as a collection of nested entities (document scenes). This organization of the document is achieved to construct pieces of information so that semantically related elements build a new atomic entity with higher granularity. We call this operation the logical composition of the document.

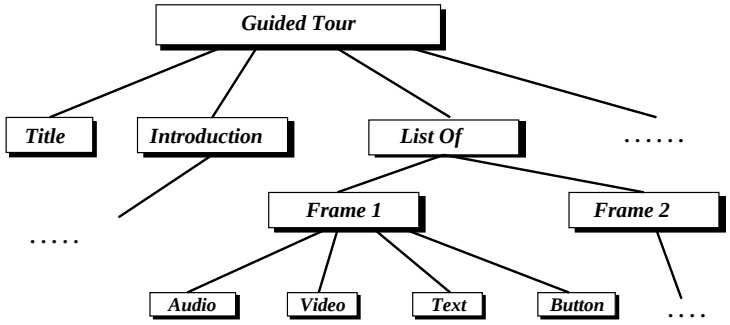

Figure 1: The logical Structure

In our approach, a multimedia document is considered as an organized set of *typed elements* which are logically related to each other. Typical elements of these documents are titles, paragraphs, tables, graphics, figures, 3D animation, audio, video, etc. All these elements have various structural relationships: a movie *contains* a video, some paragraphs and *optionally* background music; a scene *follows* another scene; a part of an image may *refer to* a section item, etc. International standards like SGML[6] or HyTime[8] have extended this approach by defining classes of documents or DTDs (Document Type Definitions). A DTD is a BNF-like grammar that describes the document content and drives the authoring process through the user interface[12].[10] Several projects have used the structuring scheme to automatically produce spatial layout in classical documents[13].[9]

We claim that similar work can be achieved for the temporal dimension. Usually, existing models do not make a clear difference between the logical and the temporal dimensions since they use the temporal operators to build logical clusters.

2.2 Spatial Composition

The spatial composition corresponds to the operation of laying out the multimedia objects on the media channels (screen, audio channels, ..). In Thot,[13] the layout presentation of multimedia elements is defined by the size of their bounding boxes and their position on the screen. These presentation parameters are determined by the help of some user defined generic rules and attributes defined for each class of documents (a DTD). These generic rules use the information of the object types, the document structure and the temporal constraints to define the spatial layout automatically. The organization of the document as a hierarchy is also used to propagate the style attributes like text fonts, colors and sizes. For flexibility reasons, this mechanism is completed by specific rules that have higher priority than the generic ones, for example, to introduce a different text font for a subset of a section other than its default one.

Due to the time based nature of multimedia presentations, media objects appear and disappear on the different channels at specific time instants. In our model, four events are of interest for the spatio-temporal information processing, these are the allocation of a media channel for a given object, its start and end instants on this channel according to temporal constraints and the deallocation of the channel. These events allow us to model various situations of the presentation. For example, when playing a video on the screen, the end of the play of a video object (i.e. end of its play time) can be disassociated from the instant corresponding to the action of freeing the area that it occupies on the screen. Similarly, instants are defined for buttons where playtime and endtime instants correspond to their enabling and disabling in the user interface.

The relation between the temporal and the spatial dimension of the document is defined as follows: Every start or end time instant of a temporal object corresponds to a given set of spatial operations called spatial

Relation	Translation to time points relations	Constraint on the delay t
x before(t) y	end(x) < begin(y)	t = [0, ∞[
x equals y	(begin(x) = begin(y)) $\wedge$ (end(x) = end(y))	-
x meets y	end(x) = begin(y)	-
x overlaps(t) y	(begin(x) < begin(y)) $\wedge$ (end(x) > begin(y)) $\wedge$ (end(x) < end(y))	t=[0, dur(x)[
x during(t) y	(begin(x) > begin(y)) $\wedge$ (end(x) < end(y))	t=[0, dur(y)-dur(x)[
x starts y	begin(x)=begin(y)	-
x finishes y	end(x) = end(y)	-

Figure 2: The set of temporal operators

events. These events are used to fire the *mapping* or *unmapping* of objects on the screen, and their *start* or *stop* at specific moments of the presentation. The presentation layer of our authoring tool uses the logical and temporal information to decide what actions are to be performed at each step of the presentation. For example, in Fig. 1 we have a composite object, called frame1, whose all component objects are required to be *mapped* spatially on the screen at the same instant, then each object will *start* at its right time according to the temporal constraints. Similarly, another composite object, called frame2, is to make the same thing but after all the component objects of frame1 deallocate their media channels.

2.3 Temporal Composition

Temporal composition is a language driven process that defines the temporal synchronization of multimedia objects. Generally, temporal languages are based on a functional description so that encapsulation becomes straightforward[7].[11] The drawbacks of these descriptions is that encapsulation is achieved at the cost of the temporal language expressiveness, because the temporal representation of the document is restricted to a tree structure.[11] In contrast, other models use relational grammars for composition which permits a better temporal expressiveness[2] but a consistency checking mechanism is required. Our model is based on the general relational model of Allen[1] extended to take into account both the quantitative aspect of intervals and delays and the nesting of logical parts of multimedia documents. We believe that this approach provides a better compromise between the expressiveness of the temporal language and the efficiency of the consistency checking algorithms.

First, we distinguish between two types of temporal information: symbolic specifications and numerical constraints. Symbolic specifications are primitive relations introduced by Allen and possibly their disjunctions permitted by the pointizable interval algebra.[14] This restricted algebra can be translated into time point algebra relations and the associated numerical constraint propagation methods can be achieved in a polynomial time. In our temporal language, we restrict the set of operators to the non-disjunctive case called STP (Simple Temporal Problems[145]). This symbolic representation allows the user to set the temporal constraints of the document through a high level interface.

From the author's point of view, the temporal structure of a multimedia document is defined as a set of temporal relations, as defined in Fig. 2. Some types of relations may be associated with a delay parameter t. Each relation relates two objects which can be one of the following types:

- **A basic media object (BMO)** which consists of a basic media data like audio, video, graphics and text. It corresponds to the leaves of the logical structure's hierarchy.

- **A composite object** which represents an entity composed of a number of media objects (basic or composite) having a certain semantic in itself (a type). It corresponds to the nodes of the logical structure's hierarchy.

A composite element is in fact an isolated part of the document which is spatially, temporally and logically independent. Therefore, it is composed with the other objects only by its end points (the type for logical composition, the bounding box for spatial composition and begin and end instants for temporal synchronization). This property ensures the global consistency of the whole multimedia document, where consistent composite objects ensure that they won't be the source of any inconsistency (i.e. considered as basic media objects).

A basic media object is described by a set of attributes. These attributes reflect the temporal nature of the object and are as follows :

Attributes of BMO = { *Name, Type,* {*Temporal Attributes*}, {*Spatial Attributes*}}

Temporal Attributes = { *Temporal Type, Duration Range*}

The attribute *Name* is a global identifier that allows access to objects at the storage level. The attribute *Type* defines the type of the object according to the logical structure. The *Temporal Attributes*, with its two components *Temporal Type* and *Duration Range*, describes the temporal behaviour of an object. The *Temporal Type* attribute takes the values {*Discrete, Continuous, Indeterministic*}. The *Duration Range* attribute is the allowable range of durations, finite or infinite, which an object can have at the presentation stage.

3 TEMPORAL CONSTRAINT NETWORKS

A temporal constraint network is an internal representation of the document which is extracted from the symbolic temporal relations introduced by the author. This network is represented in the form of a Directed Acyclic Graph (DAG) that is composed of a number of nodes. Each node in the network represents a time instant at which one or more intervals can start or finish. Each arc represents a temporal interval between the nodes at its start and end, showing a temporal constraint for a duration or a delay as defined in Fig. 2. We add two abstract time-points BEGIN and END corresponding to the dates of the beginning and the end of a given composite object. The playing of a multimedia document can be considered as a navigation in the constraint network starting from its BEGIN node to the END one, passing by all the nodes in the network at the right instants according to the temporal constraints and the logical structure. This representation of the document is the basic data structure that we use to hold the numerical constraints of the entire document.

3.1 Controllability and Flexibility of Temporal Presentations

In the classification of multimedia objects introduced in Section 2, we distinguish between *discrete*, *continuous* and *indeterministic* objects. At the abstract level, all these types of objects as well as the delays can be represented as intervals with given durations. But when considering the mechanism of constraints propagation, one can notice a fundamental difference between two types of intervals: free or contingent.

Definition 1: A *free* interval is an interval with a numerical constraint $\mathbf{[l, u]}_f$ (where l < u, l and u are positive values and u can have an infinite value) on its duration such that its value can be freely assigned to any value in the range **[l, u]**. This kind of interval models the behaviour of continuous objects whose durations can be chosen in a finite range according to allowable play-back rates or discrete objects whose durations can be assigned to any value (the range, in this particular case, is infinite).

Definition 2: A *contingent* interval is an interval with a numerical constraint $[\mathbf{l}, \mathbf{u}]_c$ on its duration which is a random value in the range [**l, u**]. This kind of intervals models the behaviour of indeterministic objects whose values are random.

Following these two definitions, we can distinguish between two aspects of a temporal scenario : the *flexibility* and the *controllability*. The *flexibility* measures the ability of a temporal scenario to be adjusted in order to meet the consistency requirements. It can be achieved using the free delays and the properties of some multimedia objects to be shrunk or stretched in a given range of durations. This range is generally chosen in such a way that it preserves the intelligibility of these objects at the presentation time. As we will see in the next section, *flexibility* will also be used to tackle the indeterministic behaviour of contingent intervals and consequently it extends the consistency checking methods. The *controllability* of a temporal scenario means the possibility to re-adjust it in order to compensate, at run-time, for the behaviour of indeterministic objects.

3.2 Constraint Network Construction

Now, we will describe a method for creating the constraint network from a list of user defined temporal constraints. We follow the assumptions: (1) no links are created from(to) the BEGIN (END) node unless a non empty list of temporal constraints exists; (2) no interval is represented in the network unless it appears explicitly in the list of temporal constraints. The first step, is creating the BEGIN and END nodes but unlinked. Next step, for each temporal constraint, we create, if not created, the nodes for the intervals appearing in it. For each interval we create two nodes, start and end nodes, with a link from the start node to the end one. Then, the nodes of different intervals are related to each other according to the type of the constraint. We either add delay links between the nodes, as in the case of the "before" constraint, or fuse two nodes into one node, as in the case of the "starts" constraint; in this case start nodes are fused. Note that each added delay link, must have a corresponding delay interval in the intervals list. In the case of fusing nodes, outgoing and incoming edges of the two fused nodes must appear in the resulting fused node, and redundant edges have to be deleted. Finally, delay links are added between the BEGIN (END) node and the nodes having no incoming (outgoing) edges.

First, we introduce the notion of temporal chains. A temporal chain $[\mathbf{e}_1, \mathbf{e}_2, .., \mathbf{e}_n]$ is a sequence of contiguous edges where each edge $\mathbf{e}_j$, $\mathbf{1 < j < n}$ is only related to one successor and one predecessor so that $\mathbf{begin\ (e}_j\mathbf{) = end\ (e}_{j-1}\mathbf{)}$ and $\mathbf{end\ (e}_j\mathbf{) = begin\ (\ e}_{j+1}\mathbf{)}$, every single interval being free or contingent.

Now, we will describe a method to identify the chains found in a constraint network. Let `N` be the list of nodes constituting the network. Let `T` be the list of nodes in `N` which are considered as starting nodes for chains. Any node `i` in `T` must satisfy the following condition: "node `i` is *not* the END node, *and either* node `i` is the BEGIN node *or* the number of its *either* outgoing *or* incoming edges is greater than one". Each node in `T` is a start node for a number of chains equals to the number of its outgoing edges. Then, each subsequent node `j` in a chain `C` is considered as an intermediate node of chain `C`, if it satisfies the following condition: "node `j` is *not* the END node, *and* the number of *both* outgoing *and* incoming edges of `j` are equal to one". If node `j` doesn't satisfy the latter condition, then it is the end node of this chain `C` and we will investigate other chains and so on.

3.3 Constraint Network Manipulation

By the manipulation of a constraint network we mean the modifications of its chains as a result of any modification in the list of temporal constraints or intervals, such as appending or removing a temporal constraint or an interval.

3.3.1 Appending Constraints

Appending a new constraint leads to: (1) the explicit appearance of its intervals in the network, and (2) the addition of new delay links and/or fusion of nodes in one node. This leads to the creation of new chains and/or the modification of previously existing ones. For the newly created chains, there is no problem to identify them as previously mentioned. The question now is how are we going to identify the modifications in the previously existing chains. The answer is as follows. It can be easily shown that we don't need to modify any old chain `C` unless it includes an intermediate node `i` whose number of either incoming or outgoing edges has been changed due to the newly added constraint. If this node `i` exists then chain `C` will be divided into two chains `C1` and `C2`. As for chain `C1`, it has the same start node as chain `C` and its end node is node `i`, and for chain `C2`, its start node is node `i` and it has the same end node as chain `C`.

3.3.2 Removing Constraints

The removal of a constraint leads to (1) the deletion of some links in the network and/or the splitting of a node into two separate nodes, and (2)perhaps, the removal of intervals from the network. The method of removing a constraint depends on its type. A constraint is removed by either the *removal of a delay link* or the *unfusion of a node* into two nodes.

The removal of a delay link is needed in case of removing temporal constraints of types: before, during, overlaps, and their reverse types. Suppose that we need to remove the delay link <`i,j`> which starts at node `i` and goes to node `j`. Let the chain `C` be that which contains the link <`i,j`>. Apply the following steps:

1. Delete link <`i,j`> and remove chain `C` from chain list.
2. If the number of outgoing edges of `i` becomes zero, then add a delay link <`i,END`>, and create a new chain `C1`, where it has the same sequence of nodes as chain `C` from the start node of `C`, until node `i`, then `C1` ends with the node `END`.
3. If the number of incoming edges of `j` becomes zero, then add a delay link <`BEGIN,j`>, and create a new chain `C2`, where it starts by node `BEGIN` then it has the same sequence of nodes as chain `C` from node `j` until the end node of `C` where `C2` ends.
4. If the number of both incoming and outgoing edges of `i` becomes equal to one, and `i` was both the end node of chain `CH1` and the start node of chain `CH2`, then add new chain `CH` which is the result of concatenation of chains `CH1` and `CH2`.
5. Repeat step (4) for node `j`.

Unfusing (or splitting) a previously fused node `n` into its two original nodes `i` and `j`, is needed in case of removing temporal constraints having the following types of operators: equals, meets, starts, finishes, and their reverse operators. In the unfusing operation, one must ensure that the resulting nodes `i` and `j` have the correct subset of both incoming and outgoing edges of the node `n`. To handle the modification of chains, we add a *dummy* delay link <`i,j`> with zero duration, then remove it using the above method of *removing of a delay link*.

Finally, an interval, of the removed constraint, has to be removed from the network *only* when (1) its nodes are not fused with other nodes, and (2) it is linked only to BEGIN and END nodes.

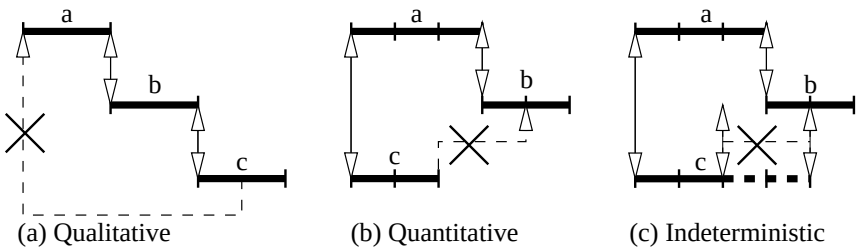

Figure 3: Temporal inconsistency examples

3.3.3 Appending and Removing Intervals

A newly appended interval will not appear in the network until it appears explicitly in the temporal constraints list. For the removal of an interval `I`, the above method of *removing constraints* can be applied for every constraint in which interval `I` appears.

4 DOCUMENT CONSISTENCY

Modifying a document by incremental specification of new objects and relations, introduces additional temporal constraints. This operation may lead to a temporal inconsistency because temporal parameters of objects (duration, begin-end dates) must satisfy invariants related to the time progression. In the temporal DAG, this invariant ensures the causal ordering of presentation events. For example, given three objects **a**, **b** and **c** as in (Fig. 3-a), then specifying that **a meets b** and **b meets c** makes the specification **c overlaps a** inconsistent. This is called a *qualitative inconsistency* because it results from the nature of the relations independently of the durations of objects they link. However, qualitative consistent specifications may be *quantitatively inconsistent* with respect to objects' durations. For example in (Fig. 3-b), specifying that **a meets b** and **a starts c** makes the qualitative consistent specification **c overlaps b** inconsistent, if durations of **a** and **c** are 20 and 30 seconds, respectively. Finally, in (Fig. 3-c) which is the same as the previous example but interval **c** is an indeterministic object whose duration is bounded in the interval [20, 40], we have an inconsistency due to *indeterminism*, since the end of **c** may occur at any time, therefore this specification may possibly fail.

A considerable range of the existing interval based temporal models of multimedia documents are incomplete in the sense that they don't handle the indeterministic behaviour of objects. Therefore, they don't make any difference in the way of handling free and contingent intervals. Consequently, temporal information such as user interactions, latency, and network delays become difficult to be represented and managed.

In the following section, we will expose a way to address both the problem of maintaining the temporal consistency in presence of indeterministic media objects and the efficiency of the algorithm achieving this operation.

4.1 Consistency Checking Algorithm

Given the DAG representation extracted from the symbolic graph, we can make the following statements :

- Circuits are not allowed (Acyclic Graph) since "instants coming from the past can not be equal to instants in the future" and vice versa.

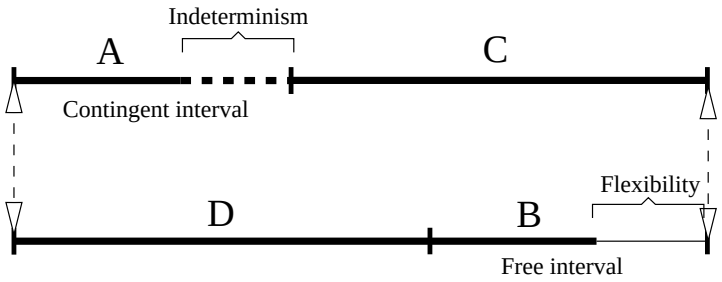

Figure 4: Example of scenario adjustment

- Cycles are always composed of two parallel chains (DAG property) whose total durations must be compatible. The temporal compatibility of two temporal chains means that the intersection of the range of allowable durations of the two chains is not empty. This property ensures that the temporal coincidence of the end points of the two paths can be guaranteed at run-time.

These two statements distinguish the sources of the three types of inconsistencies introduced earlier. First, we make a fundamental assumption about the observance of the scenario at run-time. Since the specification stage of a temporal scenario is somehow a prediction of how events will occur over time, we assume that the only observable states of the temporal scenario at run-time are the begin-end points of the intervals.The main idea behind this assumption is to provide by static methods the means for deciding if a given scenario can be dynamically adjusted or not (dynamic resynchronization of the presentation).

Our algorithm uses these observable states of the presentation in two stages: first, to establish the effective durations of unpredictable intervals, then to compensate these random delays by a duration adjustment (change of presentation speed) of the forthcoming free intervals, called provision intervals. The amount of flexibility provision taken from a free interval to compensate for one random delay must be reserved as it can't be used to compensate for another random delay. Furthermore, the value of random delays must be observed before the start of continuous provision intervals because of our assumption, unlike delay and discrete provision intervals whose end can be delayed dynamically without affecting the presentation's quality.

Before getting into the description of the algorithm, let us first show how this way of checking the temporal consistency of two chains works through simple examples.

In the first example, the author specifies an equal relation between an indeterministic object **A** *(a button)* and a discrete object **B** *(a text)*. This is represented in the DAG by two parallel edges, one for each interval, with one common fused start node as well as one common fused end node. It is obvious that this scenario is consistent since the end of the object **B** (represented by a free interval $[\mathbf{0}, \infty]_f$) can be freely delayed until the occurrence of the end instant of object **A** (represented by a contingent interval $[\mathbf{l},\mathbf{u}]_c$), whenever it happens.

In the second example, we take a part of a scenario where we have two chains that are constrained to be of equal durations. In order to satisfy this constraint we have to ensure that begin and end points of these chains start and finish at the same time. The object **A** is an indeterministic object whose end point can occur at any time in the range represented by a dashed line (see Fig. 4). The object **B** is a continuous object with an amount of flexibility represented by a thin line in the figure. Here again, it is easy to see that if the amount of flexibility of the object **B** is higher than the unpredictable range of the object **A** *(condition 1)*, and if the the observable point precedes the start point of the object **B** *(condition 2)*, then the temporal coincidence of the end instant of the objects **C** and **B** can be achieved. Therefore this part of the scenario is consistent since it is controllable.

One can notice that a given chain, if considered through its end points, defines a new interval, and therefore its total duration can be reduced into two parts, one part representing the total duration of free intervals and another part for contingent ones as follows :

Let **C** be a chain containing the sequence $\mathbf{A}_1$, $\mathbf{A}_2$, ..., $\mathbf{A}_n$ of intervals, each interval **A** has a duration range **[L, U]** and can be free or contingent. Note that a given chain can be considered as a free interval if it contains only free intervals, and contingent if it contains at least one contingent interval. The total duration of a chain and some of its parameters are defined as follows ($\oplus$ is the composite operator):

Duration_Range(C) = Free_Range(C) $\oplus$ Contingent_Range(C)
Free_Range(C) = $[\sum_{Free(C)} L_i, \sum_{Free(C)} U_i]_f$ = [CLF, CUF]$_f$
Continget_Range(C) = $[\sum_{Ctg(C)} L_i, \sum_{Ctg(C)} U_i]_c$ = [CLC, CUC]$_c$
Flexibility = CUF - CLF , Indeterminism = CUC - CLC

Where ***Free*(C)** and ***Ctg*(C)** are the set of free and continget intervals, respectively, of chain **C**. The consistency checking algorithm which is based on the previous assumption and statements works according to these four stages. For each new specification:

Stage 1 (qualitative consistency): If the new relation introduces a circuit in the DAG then this specification leads to a qualitative inconsistency (statement 1). It is therefore rejected. This can be detected by maintaining a topological sort of the constraint network.

Stage 2 (forming the chains): If it introduces new cycles, we build the corresponding temporal chains and add them to the existing ones to form the new cycles (if any). Each duration range of these chains is then computed as described earlier. As a result, the numerical constraints of the whole document are, after this stage, represented by this set of chains through their free and contingent duration ranges. The temporal consistency of the scenario is then reduced to the problem of finding a combination of values that ensures for every cycle of the graph a temporal coincidence of its two end points. Since we proceed by incremental steps, this operation is first achieved for the new chains then for the other depending chains (progressive propagation of the constraints).

Stage 3 (testing the quantitative consistency): The newly introduced chains are then checked one by one against the existing ones as follows: As we have four possible combinations of chains depending on their behaviour (free/contingent), a first pass of the algorithm is used to solve some trivial cases by checking some necessary conditions on the durations ranges.
For every cycle composed of two chains **C1** and **C2**:

```
Range = Duration_Range(C1) ∩ Duration_range(C2)
if (Range = Null) then Inconsistent,  goto stage 4
Ind = Indeterminism(C1) + Indeterminism(C2)
Free = Flexibility(C1) + Flexibility(C2)
if (Ind>Free) then Inconsistent, goto stage 4
if (C1 not free) or (C2 not free)
   then if (Adapt (C1, C2) = True)
           then success, proceed with the next cycle
        else Inconsistent,  goto stage 4
```

The function **Adapt (C1, C2)** looks for a correspondence between contingent intervals contained in **C1** and **C2** and forthcoming flexible intervals in a way that recovers all the indeterministic behaviour. Here again some trivial cases arise, for example two chains can not be ended by contingent intervals since one can not guarantee that their two end-points finish at the same time. In the general case, for each contingent interval **c**, the choice of corresponding recovery interval is based on the following hint : if **c** belongs to **C1** and the duration of **C1** is less than the duration of **C2**, we try first to find a recovery interval in **C1** so that the overall durations of **C1** and **C2** gets closer, otherwise we look for one in **C2**. At the end of this stage, if all the contingent intervals have

been recovered and the durations of **C1** and **C2** are still intersecting we have found a solution, in all the other cases our method fails (jump to stage 4).

Stage 4 (Restoring the previous state of the document): In this stage, an inconsistency was detected, therefore we restore the state of the document as it was before **step 1**, and an error is then reported to the author.

This algorithm is applied to check the consistency of both the entire document and single composite objects defined by the user. Composite objects are considered, at each level of the logical structure, as a single interval with its own total duration (having free and contingent parts). This total duration corresponds to maximum duration of the BEGIN-END paths in the DAG.

4.2 Discussion

Our current algorithm is a first step in extending the temporal synchronization methods to handle indeterministic behaviour of multimedia objects. We have shown that this kind of object can be included in the model by considering contingent intervals. This latter class of interval models a wide range of multimedia objects like unpredictable delays in media delivery and user interactions. As we have shown through simple examples, one can handle such situations using similar techniques. Our algorithm based on the **Adapt** function is not minimal. Since we use an opportunistic method to find recovery intervals tackling unpredictable delays, some of the configurations are not solved even if a solution exists. We are currently investigating recent methods developed in planning which provide minimal solutions[4].[3]

5 CONCLUSION

We presented in this paper a temporal model that allows the construction of consistent temporal presentations. The consistency checking is performed using qualitative as well as quantitative temporal relations through constraint networks. The user interface is based on the set of operators introduced by Allen, extended to the quantitative aspect of intervals, delays and unpredictable behaviour. In particular, our model allows the representation of indeterministic objects which have unpredictable durations. In the design of this algorithm, we put the emphasis on the following aspects :

- The flexibility of the temporal specifications by allowing the user to specify free delays and durations of multimedia objects.
- The encapsulation which fits the intuitive organization of the document as a set of logical entities. This aspect allows us to reduce the complexity of the abstract representation of the document since the temporal constraints are first solved within composite objects.
- The consistency checking by the use of an internal representation of the document as a labelled DAG holding the underlying numerical constraints. This DAG allows the user to identify precisely the sources of inconsistencies. The consistency of a given specification is therefore reduced to the compatibility of incrementally produced temporal chains allowing higher efficiency.[5]

This work is currently being undertaken as part in the Opera project. The goal of this project is to develop an editorial environment for the construction, manipulation and storage of complex multimedia documents. The Opera system is based on a logical document structuring scheme that was introduced and tested during the development of the Thot editor.[12]

6 REFERENCES

[1] ALLEN J.F., "Maintaining Knowledge about Temporal Intervals", *Communications of the ACM,* vol. Vol. 26, num. No. 11, pp. 832-843, november 1983.

[2] BUCHANAN M.C., ZELLWEGER P.T., "Automatic Temporal Layout Mechanisms", *Proceedings of the First ACM International Conference on Multimedia,* pp. 341-350, Anaheim, California, august 1993.

[3] DUBOIS D., FARGIER H., PRADE H., "The Use of Fuzzy Constraints in Job-Shop Scheduling", *In Proceedings of IJCAI 93 Workshop on Knowledge-Based Planning, Scheduling and Control,* Chambéry (FRANCE), 1993.

[4] FARGIER H., LANG J., SCHIEX T., "Mixed Constraint Satisfaction: a Framework for Decision Problems under Incomplete Knowledge", *submitted to IJCAI 95.*

[5] GHALLAB M., VIDAL T., "Focusing on a Sub-Graph for Managing Efficiently Numerical Temporal Contraints", *Proceedings of FLAIRS,* vol. Melbourne Beach (FL), , 1995 (To appear).

[6] GOLDFARB C. F., *The SGML Handbook,* Oxford University Press, Oxford, 1990.

[7] HARDMAN L., VAN ROSSUM G., BULTERMAN D.C.A., "Structured Multimedia Authoring", *Proceedings of the First ACM International Conference on Multimedia,* pp. 283-289, Anaheim, California, august 1993.

[8] HyTime Information Technology, "Hypermedia/Time-based Structuring Language (HyTime)", *ISO/IEC 10743,* november 1992.

[9] LAMPORT L., *LATEX: a document preparation system,* Addison-Wesley, New York, 1986.

[10] LAYAÏDA N., VIONDURY J.Y., "Multimedia Authoring: a 3D Interactive Visualization Interface based on a Structured Document Model", *Sixth International Conference on Human-Computer Interaction,* Yokohama, Japan, july 1995.

[11] LITTLE T.D.C., GHAFOOR A, "Interval-Based Conceptual Models for Time-Dependent Multimedia Data", *IEEE Transactions on Knowledge and Data Engineering (Special Issue : Multimedia Information Systems),* vol. Vol. 5, num. 4, pp. 551-563, august 1993.

[12] QUINT V., VATTON I., "Grif: An Interactive System for Structured Document Manipulation", *Text Processing and Document Manipulation, Proceedings of the International Conference,* pp. 200-213, Cambridge, 1986.

[13] ROISIN C., VATTON I., "Merging Logical and Physical Structures in Documents", *Proceedings of the Fifth International Conference on Electronic Publishing, Document Manipulation and Typography,* pp. 327-337, Wiley Publishers, Darmstadt, Allemagne, april 1994.

[14] VILAIN M., KAUTZ H. A., "Constraint Propagation Algorithms for Temporal Reasoning", *Proceedings of AAAI,* pp. 377-382, Philadelphia, august 1986.

Contents

1 INTRODUCTION **1**

2 DOCUMENT COMPOSITION MODEL **2**

2.1 Logical Composition 2

2.2 Spatial Composition 3

2.3 Temporal Composition 4

3 TEMPORAL CONSTRAINT NETWORKS **5**

3.1 Controllability and Flexibility of Temporal Presentations 5

3.2 Constraint Network Construction 6

3.3 Constraint Network Manipulation 6

3.3.1 Appending Constraints 7

3.3.2 Removing Constraints 7

3.3.3 Appending and Removing Intervals 8

4 DOCUMENT CONSISTENCY **8**

4.1 Consistency Checking Algorithm 8

4.2 Discussion 11

5 CONCLUSION **11**

6 REFERENCES **12**