A Study on Multimedia Documents Behavior: a Notion of Equivalence

Paola Bertolotti (bertolot@di.unito.it)
Dipartimento di Informatica, Università degli Studi di Torino
Corso Svizzera 185, 10149 Torino, Italia

Ombretta Gaggi (gaggi@dsi.unive.it)
Dipartimento di Informatica, Università Ca' Foscari di Venezia
Via Torino 155, 30172 Mestre (VE), Italia

Abstract. In this paper we address the problem of comparing multimedia documents, which can be described according to different reference models. If we consider presentations as collections of media items and constraints among them, expressed according to their reference model, they must be translated to a common formalism in order to compare their temporal behavior and detect if they have a common component (i.e., *intersection*), if one of them is included in another one (i.e., *inclusion*), or if they have the same temporal evolution along time (i.e., *equivalence*).

In this paper, we propose the use of automata, to describe the temporal evolution of a document, and the SMIL language as a case study, since this standard allows to describe the same behavior with different sets of tags. In case of behaviorally equivalent SMIL documents, we propose an algorithm to extract the canonical form that represents this behavior.

1. Introduction

A multimedia presentation can be defined as a collection of different types of media items and a set of spatial and temporal constraints among them. Authoring such complex documents is more difficult when they are interactive, since unexpected user interaction can alter the correct timing relationships between media objects. In literature, many design models [1, 6, 8] have been proposed to address this problem. Among them, SMIL [9], Synchronized Multimedia Integration Language, is a W3C standard markup language to describe multimedia presentations.

The variety of reference models available for multimedia documents provides great expressiveness, but also some problems in the management of such documents, since it is very difficult to compare multimedia presentations designed with different reference models. In particular it is not possible to state when they have the same temporal behavior unless the playback. Sometimes, the same problem could arise even if the documents use the same reference model: as an example, we

consider SMIL presentations, in which different sequences of tags (and then, different scripts) can describe the same temporal behavior.

Consider for instance, the following SMIL document:

```
⟨par id="par1" dur="60s"⟩
    ⟨video id="intro" end="20s"/⟩                                    (1)
    ⟨audio id="artwork" begin="intro.end" dur="30s"/⟩
    ⟨img id="picture" begin="artwork.begin+5"/⟩
⟨/par⟩
```

Intuitively, its temporal behavior is described in Figure 1: the video starts with the script, the audio starts when the video ends (as specified by the value of the `begin` attribute) and the image starts five seconds after the audio. After the end of media *artwork*, *picture* is rendered until the termination of the whole script.

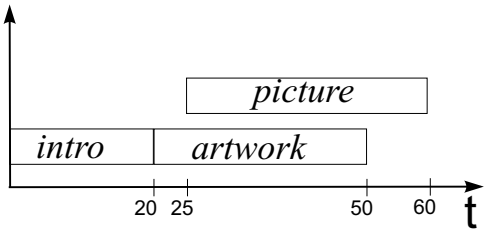


Figure 1. Timeline of the SMIL documents.

The same behavior is also obtained in a different way, since the use of attributes `begin`, `end` and `dur` can deeply modify the behavior of the corresponding media items, as described by the following SMIL script:

```
⟨seq id="seq2"⟩
    ⟨video id="intro" dur="20s"/⟩
    ⟨par id="par2"⟩
        ⟨audio id="artwork" end="30s"/⟩                              (2)
        ⟨img id="picture" begin="5s" dur="35s"/⟩
    ⟨/par⟩
⟨/seq⟩
```

In fact, at the end of the media *intro*, the audio starts, followed by *picture* after five seconds. These two documents are, therefore, *behaviorally equivalent*.

The notion of *behavioral equivalence* is particular important when we do not deal with single presentations but with database of multimedia

documents. In this case, we need a formalism to compare them in order to avoid redundancy (i.e., the same presentation expressed through different reference models) and to express queries. In particular, we aim at defining a formalism able to describe documents' behavior independently from the design model. This can be used to formulate an example presentation to query (by example) the heterogenous database on the base of the multimedia content, but also of the overall behavior of the document (e.g., a sequence of images regarding the Pacific Ocean).

Since the behavioral equivalence finds out only exact matches in the behavior of the multimedia presentations, two more useful notions are also discussed in this paper: the notion of *inclusion*, i.e., when a document is a section of a second one, and the *behavioral intersection*, i.e., when two documents share a common section, both in term of media items and temporal relations.

If we consider presentations as collections of media items and temporal and spatial constraints among them, in order to detect equivalence, inclusion or intersection in their behavior, we need to compare only their temporal evolution. For this reason, we consider only the temporal relationships of the document and we do not address the spatial constraints.

In this paper we use the automata as a general formalism to describe the temporal evolution of multimedia documents independently from the original design models: two documents can be compared, with respect to the notion of equivalence, inclusion and intersection, as long as their behaviors can be described by an automaton.

In the rest of the paper, we propose the SMIL language as a case study, since this standard allows to describe the same behavior in more than one script. In case of behaviorally equivalent SMIL documents, we propose an algorithm to extract the canonical form that represents this behavior.

2. Related work

2.1. The notion of equivalence

The notion of *equivalence* is not new in the context of multimedia documents, but other works use this concept in different contexts and with different meanings and purposes.

In [4], Boll et al. define a notion of *semantic* equivalence in the context of the adaptation of multimedia presentation: in order to deliver multimedia information to the user, media elements can be replaced (i.e., adapted) by other media elements of different quality and type,

to be playable in the user context, but with the same content, i.e., *semantic equivalent alternatives*. Therefore, Boll et al. do not compare different documents, but redundant media alternatives inside the same adaptive presentation. Different from our approach, the authors define an equivalence only with respect to the content of the presentation fragment, and does not consider in details the temporal structure.

Another work [7] focuses its attention on the temporal evolution of multimedia applications and presents a semantics for UML models incorporating temporal information. Instead of automata, the dynamic behavior is specified by transformation of timed graphs. The authors discuss the equivalence of graph transformation rules: if it is not possible to distinguish different interleavings of concurrent actions, the sequences of transformation steps are equivalent. In other words, Hausmann et al. do not compare different documents, but sequences of events, i.e., different behaviors, of the same document, that are equivalent if they lead to the same result.

In [10], Lo Presti et al. propose a similar approach to the one defined in this paper, but they are able to compare only documents using the TAO (Temporal Algebraic Operators) reference model. TAO is a formal language to describe temporal composition of multimedia objects. using well-defined operators which can be composed. Two *terms* (i.e., TAO programs) are equal if the semantics of their operators are equal. Different from our approach, equivalence can be decided on the base of a *static* analysis on the program structure.

2.2. Modelling multimedia documents with automata

Other works in literature propose a model to describe the behavior of a multimedia presentation but does not exist an agreement on an abstract representation of the dynamics of a multimedia document which is independent from the design model it adopts.

In particular, Yang [13] studies the SMIL language to find out possible errors, resulting in temporal conflicts, in the definition of the document. He proposes a conversion of SMIL 1.0 documents in the *Real Time Synchronization Model (RTSM)*. Yang uses a model based on Petri Nets and converts the synchronization elements in SMIL into transitions and the related attributes are associated with places. Unfortunately Yang's approach applies only to SMIL documents and requires to convert every single SMIL element.

In [11] Sampaio et al. present a formal approach to design an Interactive Multimedia Document (IMD) and to perform the schedule of a presentation, if the temporal constraints are satisfied. In this case, the document can be described using any design model (SMIL or others);

then the created document is translated into RT-LOTOS specification; from RT-LOTOS, the reachability graph is obtained and it is used to verify consistency property. The scheduling graph is derived from the consistent reachability graph and it is called *Time Labeled Automaton* (TLA): at every state in the automaton is associated a clock (timer), that measures the time during which the automaton remains in that state; at every transition are associated some conditions, i.e., the constraints that have to be satisfied for the firing of the transition.

In [12] Stotts et al. use an automaton to describe hyperdocuments, that is, interactive documents characterized by dynamic properties. An automaton describes the document from the point of view of the process of browsing within it: the linked structure can be seen as a state transition diagram of a finite state machine. In this case, the automaton is not used to represent the temporal evolution of a multimedia presentation but the browsing strategy adopted by the user, i.e., which link he, or she, followed, and at what time.

Automata with temporal characteristics to model the behavior of real-time systems are also used in [2]: the authors define timed automata which are able to express timing delay in real-time systems. A timed automaton has a set of real-valued clocks: they can be reset with a transition and constraints over the clocks control the firing transition, that is, a transition takes place if the associated clocks constraints are satisfied. In this case, the clocks control if an event occurs in the right time instant.

3. The SMIL language

In this paper we use SMIL [9] language as a case study, since it is a W3C standard and it allows to define the same temporal behavior through different sets of tags and attributes (see script 1 and 2).

A SMIL file is divided in two sections: the `layout` section defines the regions, i.e., rectangular areas on the user screen in which media items are visualized, and the `body` section contains the definitions of media items involved in the presentation, and the temporal relationships among them. The language SMIL also allows to define *transitions*, i.e., visual effects between two objects, and *animations*, i.e., modifications to the value of some attributes (e.g., the color, the size, the position, etc.) of a media item.

SMIL does not define a reference model for the data structure, but only tags for describing media objects behavior. Synchronization is achieved essentially through two tags: `seq` to render two or more objects sequentially, one after the other, and `par` to play them in parallel.

Using attributes `begin`, `end`, and `dur` it is possible to fix the start and the end time of a media item. Consider a media item inside a `par` block. If its attributes `begin`, `end` or `dur` are undefined, the media item starts at the beginning of the `par` block and ends with its natural termination. Otherwise it begins (ends) a certain amount of time after the beginning of the `par` block, given by the corresponding attribute value. In script 1, the attribute `begin = ‘‘intro.end’’` makes audio *artwork* start when video item *intro* ends. Therefore, the beginning and the end of a media item can be defined with reference to the tag in which it is contained, or according to a particular event, even completely changing the semantics of tags `par` and `seq`, as in the case of script 1. The attribute `dur` defines the duration of an object.

The tag `excl` is used to model some user interactions. It provides a list of children elements, and only one of them may play at any given time. We refer to [9] for more details about this standard.

4. A formal description of the temporal evolution of a multimedia document

In order to compare the behaviors of different multimedia documents, we need a common formalism, which is independent from the reference model of the document.

Several aspects must be taken into account: first of all, a presentation is characterized by the set of media items presented to the user; secondly, we consider a set of *events*, i.e., the “registration” of some changes in the situation of the components of the document, and, accordingly, to the document itself. These changes are due to some properties of the objects, like their duration, to temporal constraints of the document, and to user interactions. Using SMIL, the temporal constraints are defined by the tags `par` and `seq` and the attributes `begin`, `end`, and `dur`. Such elements provide an intensional representation of the evolution of the presentation in time.

A critical aspect when comparing the evolution of multimedia documents concerns user interaction, since different models allows different types of interactions (e.g., only to pause/resume the presentation playback or also to fast forward/rewind it, etc.) which can also be influenced by the interface which can allow to interact only with whole document rather than with each single media item.

Moreover, in some particular contexts, we are not interested in the response to the user, but in the natural evolution of the document. This is the case of database of multimedia presentations: if we query for a particular sequence of media items, we can neglect the documents’

evolution in case of user interaction. For these reasons, in this paper we consider only the natural evolution of a multimedia document. In this case our events "register" the beginning of a presentation (event *start*) and the natural termination of media items (event *end(m)*, where m is a media).

Beside media items, events and temporal constraints, our formalism also introduces the concept of *timer* to better specify synchronization relationships among objects. A timer is a dynamic object with a specified duration, that can represent offset between media items playback, or constraints on their duration. As an example, if a media must start twenty seconds after another one, a timer is initialized to twenty; this value is decremented according to the system clock and when the timer expires, i.e., its value becomes zero, the constrained media can start. Therefore, according to temporal constraints in a document, a timer is initialized to a value that represents its duration: adding this value to the time instant in which t is initialized, we obtain the time instant $n \in \mathbb{N}$ in which t will expire.

Therefore, the role of the timers is to mark the meaningful time instants for the evolution of the presentation, i.e., the instants in which synchronization takes place; then another possible event, recorded by the system, is the expiration of a timer (event *end(t)*, where t is a timer), that causes the start and the end of a set of objects. The set of timers used by a multimedia presentation is defined on the base of the temporal constraints of the document itself. We also need a mapping, formalized in the following, which identifies the set of objects that depends on a timer (i.e., they are waiting for its expiration).

A formal definition of a multimedia document is as follows:

Definition 4.1. (Multimedia Document) A multimedia document is a 4–*tuple* $D = \langle \mathcal{MI}, \mathcal{TS}, \mathcal{E}, \mathcal{TC} \rangle$ where

- $\mathcal{MI}$ is a set of media items $\{m_0, m_1, \ldots, m_n\}$;
- $\mathcal{TS}$ is a set of timers $t(n)$ where $n \in \mathbb{N}$ indicates the time instant in which t expires;
- $\mathcal{E}$ is a set of events $ev \in (\{start, end\} \times (\mathcal{MI} \cup \mathcal{TS}) \times \mathbb{N})$, where $i \in \mathbb{N}$ is the time instant in which an event occurs; [1]

[1] Since we describe only the natural evolution of the document, there is only a *start* event, corresponding to the start of the presentation, and then a sequence of *end* events of timers and media items. Moreover, the beginning of the whole document corresponds to the start of the first media.

– $\mathcal{TC}$ is a set of temporal constraints representing the synchronization relationships among the objects, described according to the reference model of the document.

For the sake of readability we shall denote event instances by pairs of the form $\langle e(m), i\rangle$ where e is an event type (i.e., *start* or *end*), m is a media item or a timer, and i is the time instant in which the event occurs.

If we observe a multimedia document evolution along time, it can be divided into a number of time intervals in which some conditions hold, e.g., some media are active, while others are waiting for a specific time instant. If $\mathcal{MI}$ is the set of media components that play a role in a presentation, we can describe the presentation evolution in terms of active media at any given time instant. We assume that time is discrete, and marked by a variable i ranging over $\mathbb{N}$ which is updated by a clock. The actual time resolution is not important as long as it allows the capture of all the events related to media execution and it is possible to observe the effect of time distinct events as distinct effects. Two or more events are *contemporary* if they occur at times denoted by the same value of the variable i. In this case, the instances with the same i are denoted by a unique complex event in the form $\langle\{e_1(m_1), \ldots, e_n(m_n)\}, i\rangle$, where $\{e_1(m_1), \ldots, e_n(m_n)\}$ is the set of events that occur at the same time instant i. In practise, a complex event is a more compact form for the list of contemporary events $\langle e_1(m_1), i\rangle, \ldots, \langle e_n(m_n), i\rangle$.

We define a mapping function *waiting*: $\mathcal{TS} \rightarrow 2^{\mathcal{MI}}$ which returns, for each timer, the list of media items that are waiting for it, i.e., media items whose start is triggered by a timer expiration, or which are currently playing and the expiration of the timer will trigger their forced end. For example, since video *intro* (script 1) has an attribute `end = ''20s''`, once started, it will be inserted into a list of items waiting for the expiration of a timer which lasts for twenty seconds (see t_2 in Table II).

We introduce also the two functions *begin*, *stop*: $\mathcal{MI} \rightarrow \mathbb{N}$ which return, for each element contained in the document, the time instant in which it starts or ends, respectively, according to the temporal constraints defined in the presentation.

Considering the SMIL language as a case study, since we are interested in the behavior of the document, it is sufficient to investigate only the section `body` which contains the synchronization tags among the objects: in this case the set $\mathcal{TC}$ of temporal constraints is equal to the set $\mathcal{T}ag$ of tags contained in the SMIL script. For the same reason, we do not consider transition effects.

Each tag $tg \in \mathcal{T}ag$ has three attributes which we denote by $tg.begin$, $tg.end$ and $tg.dur$, which contain the values of the corresponding attributes in the SMIL document (zero if undefined). The function $parent(tg)$ returns the tag in which tg is contained, and $children(tg)$ returns the list of tags defined inside the considered tag; $first(tg)$ returns the first child. A *succ()* function is also available that, given a tag in the list of children, gives as output the next element in that list[2].

Tables I and II show how timers and functions are applied to the script 1 introduced in Section 1.

Table I. Values of attributes and functions for the tags in script 1.

	par1	intro	artwork	picture
attribute `begin`	0	0	intro.`end`	artwork.`begin`+5
attribute `dur`	60	0	30	0
attribute `end`	0	20	0	0
parent()	null	par1	par1	par1
children()	intro, artwork, picture	null	null	null
first()	intro	null	null	null
succ()	null	artwork	picture	null
begin()	0	0	20	25
stop()	60	20	50	60

Table II. Timers and function *waiting()* applied to script 1.

timer	associated constraints	*waiting()*
$t_1(60)$	`dur=''60s''`	$\{intro, artwork, picture\}$
$t_2(20)$	`end=''20s''`	$\{intro\}$
$t_3(20)$	`begin=''intro.end''`	$\{artwork\}$
$t_4(25)$	`begin=''artwork.begin+5''`	$\{picture\}$
$t_5(50)$	`dur=''30s''`	$\{artwork\}$

[2] The children list is ordered with the same order in which the media items (or tags) appear in the SMIL file.

4.1. Evolution of a multimedia document

At any time instant, the document behavior is completely described by the set of media and timers that are active at that time. This information is captured by the notion of *state* of the presentation. Before the presentation starts, no media item or timer is active, thus no media item is waiting for a timer expiration, and the variable i, that represents the time progress of the system clock, is equal to 0. When an event occurs, the state of the presentation changes: some items that were not active become active, some active items naturally end, some timers expire and force the starting or the interruption of a list of items.

Definition 4.2. (State) The state of a multimedia presentation is a 5–tuple $S = \langle \mathcal{AM}, \mathcal{T}, \mathcal{B}g, \mathcal{E}nd, \mathcal{W} \rangle$, where

- $\mathcal{AM}$ is the set of active media;
- $\mathcal{T}$ is the set of timers whose expiration denotes an instant in which some objects start or end;
- $\mathcal{B}g$ and $\mathcal{E}nd$ are the set of pairs $\langle m, i \rangle$ where m is a media item and i is the time instant in which the item m begins or ends, respectively, according to the temporal constraints;
- $\mathcal{W}$ is the set of pairs $\langle t, mi \rangle$ where t is a timer and mi is a set of media items which are waiting for the timer t.

For clarity, in the following of the paper we shall refer to the association between media items and time instants with $begin(m) = i$ if $\langle m, i \rangle \in \mathcal{B}g$, and $stop(m) = i$ if $\langle m, i \rangle \in \mathcal{E}nd$. Similarly, we will use the functional notation $waiting(t(i)) = mi$ if $\langle t(i), mi \rangle \in \mathcal{W}$, to denote the association between timers and media items.

All the possible evolutions along time of a multimedia document D can be described by a finite state automaton, defined as follows.

Definition 4.3. (Automaton) Let $D = \langle \mathcal{MI}, \mathcal{TS}, \mathcal{E}, \mathcal{TC} \rangle$ be any multimedia document. Its associated finite state automaton is the 5–*tuple* $AUT(D) = \langle S, \mathcal{E}, s_0, next, Final \rangle$, where

- S is the set of possible states for the document D;
- $\mathcal{E}$ is the set of possible event instances in the form $\langle \{e_1(m_1), \ldots, e_n(m_n)\}, i \rangle$, $e_j \in \{start, end\}$ $\forall j = 1 \ldots n$, $m \in \mathcal{MI} \cup \mathcal{TS}$, $i \in \mathbb{N}$;

– s_0, the initial state, is $\langle \mathcal{AM}_0, \mathcal{T}_0, \mathit{begin}_0, \mathit{end}_0, \mathit{waiting}_0 \rangle$, where $\mathcal{AM}_0 = \mathcal{T}_0 = \emptyset$, $\mathit{begin}_0(el) = \mathit{end}_0(el) = \text{null}$ for all $el \in \mathcal{MI}$ and $\mathit{waiting}_0(t) = \emptyset$, for all $t \in \mathcal{TS}$;

– the transition function $\mathit{next} : S \times \mathcal{E} \rightarrow S$ is the mapping that deterministically associates any state s to the state s' in which s is transformed by an event $ev \in \mathcal{E}$;

– $\mathit{Final} = \{s_f, s_{err}\}$ is the set of states which correspond to the end of the presentation playback. The state s_f is $\langle \mathcal{AM}_f, \mathcal{T}_f, \mathit{begin}_f, \mathit{end}_f, \mathit{waiting}_f \rangle$, where $\mathcal{AM}_f = \mathcal{T}_f = \emptyset$ and $\mathit{waiting}_f(t) = \emptyset$, for all $t \in \mathcal{TS}$. The state s_{err} is equal to $\langle \mathcal{AM}_{err}, \mathcal{T}_{err}, \mathit{begin}_{err}, \mathit{end}_{err}, \mathit{waiting}_{err} \rangle$, where $\mathcal{AM}_{err} = \emptyset$, $\mathcal{T}_{err} = \mathcal{TS}$, $\mathit{begin}_{err}(el) = \mathit{end}_{err}(el) = \text{null}$ for all $el \in \mathcal{MI}$ and $\mathit{waiting}_{err}(t) = \emptyset$, for all $t \in \mathcal{TS}$.

The state s_f corresponds to the natural termination. If the document evolution reaches the state s_{err}, then the presentation contains some time conflicts.

The automaton describes the evolution of the system as consequence of a particular event. A word of an automaton is a sequence of events in the form $\langle e_0(el_0), ist_0 \rangle \ \ldots \ \langle e_m(el_m), ist_m \rangle$. An *accepted* word is a word that allows to reach a final state s_f. Since we consider only the natural evolution of the document, a word always begins with a *start* event, corresponding to the start of the first media of the presentation, followed by a sequence of *end* events of timers and media items. Moreover, there can be contemporary events whose effects apply after an unique step.

We note here that, if the sets of media items $\mathcal{MI}$ and timers $\mathcal{TS}$ of the document D are finite, also $AUT(D)$ is finite, even if it can contain infinite paths, as for example for a forever looping presentation.

Figure 2 depicts the evolution of script 1. Initially both $\mathcal{AM}_0$ and $\mathcal{T}_0$ are empty, neither any media nor any timers are active. Then, the activation of the `par` block causes the activation of the video *intro*, while the other media are not activated because of their `begin` attributes. Four timers are inserted in $\mathcal{T}_1$ (the associated media that wait for their expiration are showed in Table II): t_1 controls the termination of `par1`, t_2 the termination of media *intro*, t_3 and t_4 the beginning of *artwork* and *picture*, respectively, according to the attributes in the script. The obtained time instants for the begin and the end of the media are stored in the sets $\mathcal{B}g$ and $\mathcal{E}nd$. At time instant 20, timers t_2 and t_3 expire: as a consequence, media *intro* ends and media *artwork* becomes active (adding in $\mathcal{T}_2$ timer t_5 that controls its termination). After five seconds, the expiration of t_4 forces the beginning of media *picture*. At time

instant 50, when corresponding timer t_5 expires, media *artwork* ends and, finally, the expiration of the last timer t_1 forces the termination of all active media contained in the `par` block which are active and then of the whole script.

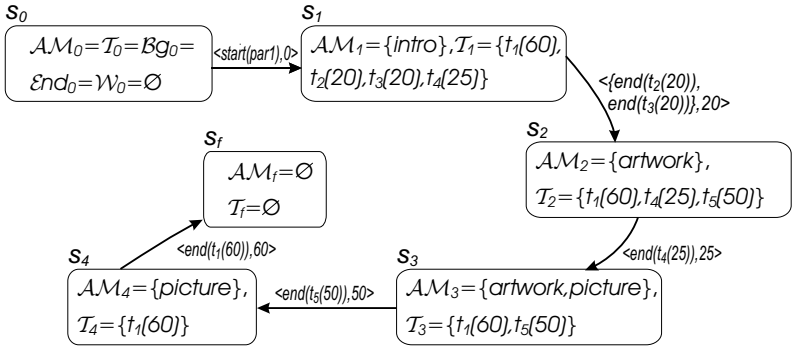


Figure 2. The evolution of script 1.

5. Studying the behavior of multimedia documents

The automaton defined in Section 4 completely describes the behavior of a multimedia document independently from the reference model used to define the synchronization relationships it contains. Therefore, it can be used to infer some properties of a presentation, e.g., if the presentation naturally ends or loops infinitely, or to compare the evolution of different documents, even if they are modelled using different reference models.

In the following, in order to compare multimedia presentations, we define the notion of *inclusion*, *intersection* and *equivalence*, based on the comparison of automata.

5.1. Inclusion

From the users point of view, the only relevant modifications in the state of a presentation are obtained from the set of media items which are active during the playback. Therefore, we must check the set of active media $\mathcal{AM}$ of each state, to find out possible common paths in the automata. For example, the automata describing the evolution of two multimedia documents can be compared to decide if one of them has the same behavior of a section of the other one: in this case, the first document is a component of the second one, therefore it is *included* in the bigger document.

More formally:

Definition 5.1. (Inclusion) Let D and D' be two multimedia documents, $D = \langle \mathcal{MI}, \mathcal{TS}, \mathcal{E}, \mathcal{TC} \rangle$, $D' = \langle \mathcal{MI}', \mathcal{TS}', \mathcal{E}', \mathcal{TC}' \rangle$ and $\mathcal{MI}' \subseteq \mathcal{MI}$. D' *is included in* D, denoted by $D' \subseteq D$, iff:

- for each word $w' = \langle e'_0(el'_0), ist'_0 \rangle \ \ldots \ \langle e'_p(el'_p), ist'_p \rangle$, accepted by $AUT(D')$, there exists a word $w = \langle e_0(el_0), ist_0 \rangle \ \ldots \ \langle e_q(el_q), ist_q \rangle$ accepted by $AUT(D)$ where $\exists j, \epsilon, hop \ 0 \leq j \leq q, \ \epsilon \geq 0, hop \geq 0$ such that $\forall i, k \ 0 < i < p, \ j \leq k < j + p - 1 + hop$, and if $ist'_i = ist_k + \epsilon$ then if $el'_i \in \mathcal{MI}'$ then $el'_i = el_k$;

- for each pair of words w, w' satisfying the above condition, if $sq = s_0 \ldots s_k \ldots s_{q+1}$ and $sq' = s'_0 \ldots s'_i \ldots \ s'_{p+1}$ are the sequences of states reached by $AUT(D)$ and $AUT(D')$, respectively, and $s_k = \langle \mathcal{AM}_k, \mathcal{T}_k, \mathcal{B}g_k, \mathcal{E}nd_k, \mathcal{W}_k \rangle$ and $s'_i = \langle \mathcal{AM}'_i, \mathcal{T}'_i, \mathcal{B}g'_i, \mathcal{E}nd'_i, \mathcal{W}'_i \rangle$ then $\forall i, k \ 0 < i \leq p, \ j \leq k \leq j + p - 1 + hop$ and if $ist'_i = ist_k + \epsilon$ then $AM'_i \subseteq \mathcal{AM}_k$ and $(\mathcal{AM}'_i \setminus \mathcal{AM}'_{i-1}) \cap (\mathcal{AM}_k \setminus \mathcal{AM}'_{i-1}) = \emptyset$.

Considering two multimedia documents D and D', D' *is included in* D if, for the sequence of states and events which describes the natural evolution of D', a corresponding sequence of states and events can be found in the automaton of D in which the sets of active media items $\mathcal{AM}_k$ contains the corresponding $\mathcal{AM}'_i$ and the events are equal[3]. If media items contained in the bigger document, i.e., in $\mathcal{AM}_k$, but not in $\mathcal{AM}'_i$ $(\mathcal{AM}_k \setminus \mathcal{AM}'_i)$, start or end at different time instants, these events are not considered; the number of this type of events is equal to *hop*. As an example, if media items B and C play in sequence during the playback of A in D (see Figure 3(a) for the automaton) and D' contains only the sequence B and C (see Figure 3(b) for the automaton), then $D' \subseteq D$.

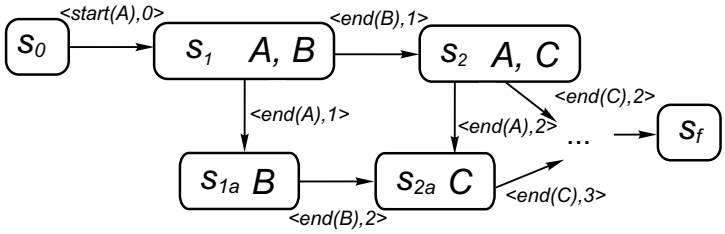
5.2. Behavioral Intersection

We can extend Definition 5.1 to a more general definition of *intersection*, which finds out possible intersections of the behaviors of the documents. Given two documents, D and D', the behavior of D' *intersects* the behavior of D if a common (sub)path exists in the two automata.

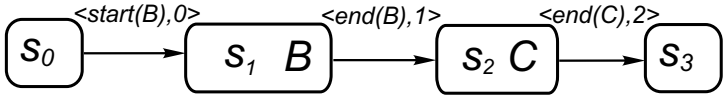
More formally:

Definition 5.2. (Behavioral Intersection) Let D and D' be two multimedia documents, $D = \langle \mathcal{MI}, \mathcal{TS}, \mathcal{E}, \mathcal{TC} \rangle$, $D' = \langle \mathcal{MI}', \mathcal{TS}', \mathcal{E}', \mathcal{TC}' \rangle$ and $\mathcal{MI} \cap \mathcal{MI}' \neq \emptyset$. D *intersects* D', denoted by $D \cap D'$, iff:

[3] We do not consider the first and the last event because we are not interested in which is the media that starts the presentation and in the subsequent evolution of D. Moreover we do not consider timers since their names are arbitrary.



(a)



(b)

Figure 3. The automaton for D (a) and D' (b) where $D' \subseteq D$.

– for each word $w' = \langle e'_0(el'_0), ist'_0\rangle \ \ldots \ \langle e'_p(el'_p), ist'_p\rangle$, accepted by $AUT(D')$, there exists a word $w = \langle e_0(el_0), ist_0\rangle \ \ldots \ \langle e_q(el_q), ist_q\rangle$ accepted by $AUT(D)$ where $\exists i, j, \delta, \epsilon, hop \ 0 \leq i \leq p, \ 0 \leq j \leq q, \ \delta \geq 0, \ \epsilon \geq 0$ and $hop \geq 0$, such that $\forall h, k \ i \leq h < i + \delta, \ j \leq k < j + \delta + hop$ and if $ist'_h = ist_k + \epsilon$ and $el'_i \in \mathcal{MI}'$ then $el'_h = el_k$;

– for each pair of words w, w' satisfying the above condition, if $sq = s_0 \ldots s_k \ldots s_{k+\delta} \ldots s_q$ and $sq' = s'_0 \ldots s'_h \ldots s'_{h+\delta} \ldots \ s'_p$ are the sequences of states reached by $AUT(D)$ and $AUT(D')$, respectively, and $s_k = \langle \mathcal{AM}_k, \mathcal{T}_k, \mathcal{B}g_k, \mathcal{E}nd_k, \mathcal{W}_k\rangle$ and $s'_h = \langle \mathcal{AM}'_h, \mathcal{T}'_h, \mathcal{B}g'_h, \mathcal{E}nd'_h, \mathcal{W}'_h\rangle$ then $\forall h, k \ i \leq h \leq i + \delta, \ j \leq k \leq j + \delta + hop$ and if $ist'_h = ist_k + \epsilon$ then $\mathcal{AM}'_h \subseteq \mathcal{AM}_k$ and $(\mathcal{AM}'_{i+1} \setminus \mathcal{AM}'_i) \cap (\mathcal{AM}_k \setminus \mathcal{AM}'_i) = \emptyset$.

Two sub-paths of length δ, in which we do not consider *hop* events involving media items in $\mathcal{AM}_k \setminus \mathcal{AM}'_i$, define the same behavior (and so can be considered a common sub-path of the automata) if, for the set of the active media items, respectively, for each pair of states (s_k, s'_h) in the sequences, $\mathcal{AM}_k$ and $\mathcal{AM}'_h$, $\mathcal{AM}'_h \subseteq \mathcal{AM}_k$ holds and the events, involving media items, are equals [4]. In this case the sequences of events and states do not need to cover the entire evolution of the documents. As an example, if D contains the sequence of media items A, B, C during the playback of E (see Figure 4(a) for the automaton) and D' contains the sequence of media B, C, F during the playback of G

[4] As in Definition 5.1, we do not consider the first and the last event for the same reason.

as illustrated in Figure 4(b), $D \cap D'$ since they share the common (sub)path which describes the sequence of media items B and C.

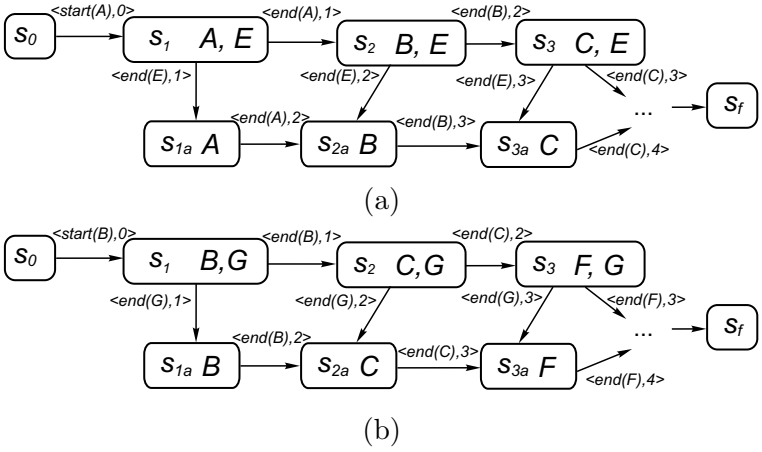


Figure 4. The automaton for D (a) and D' (b) where $D \cap D'$.

We define intersection using a common path and not a sequence of states because both the multimedia documents can also contain other media items (in the previous example E in D and G in D'): in this case, the set of active media in the automata's states are different.

5.3. Behavioral Equivalence

Given Definition 5.1 we can check if a multimedia presentation is included into another document by considering the automata which describe their behavior. In the same way we can decide if two documents, possibly described using different reference models, have the same playback, i.e., two documents share the same behavior when the states of the respective automata, after the same event, contain the same set of active media. More formally:

Definition 5.3. (Behavioral Equivalence) Let D and D' be two multimedia documents, $D = \langle \mathcal{MI}, \mathcal{TS}, \mathcal{E}, \mathcal{TC} \rangle$, $D' = \langle \mathcal{MI}', \mathcal{TS}', \mathcal{E}', \mathcal{TC}' \rangle$ and $\mathcal{MI} = \mathcal{MI}'$. D and D' are *behaviorally equivalent*, denoted by $D \sim D'$, iff:

- for each word $w = \langle e_0(el_0), ist_0 \rangle \dots \langle e_i(el_i), ist_i \rangle \dots \langle e_q(el_q), ist_q \rangle$, where $\forall i \ ist_i \leq ist_{i+i}$, accepted by $AUT(D)$ there exists a word $w' = \langle e'_0(el'_0), ist'_0 \rangle \dots \langle e'_i(el'_i), ist'_i \rangle \dots \langle e'_q(el'_q), ist'_q \rangle$ where $\forall i \ ist'_i \leq ist'_{i+i}$, accepted by $AUT(D')$ such that $\forall i = 0 \dots q \ ist_i = ist'_i$ and if $el_i \in \mathcal{MI}$ then $el_i = el'_i$;

– for each pair of words w, w' accepted by the automata, if $sq = s_0 \dots s_j \dots s_q$ and $sq' = s'_0 \ \dots \ s'_k \ \dots \ s'_q$ are the sequences of states reached by $AUT(D)$ and $AUT(D')$, respectively, and $s_i = \langle \mathcal{AM}_i, \mathcal{T}_i, \mathcal{B}g_i, \mathcal{E}nd_i, \mathcal{W}_i \rangle$ and $s'_i = \langle \mathcal{AM}'_i, \mathcal{T}'_i, \mathcal{B}g'_i, \mathcal{E}nd'_i, \mathcal{W}'_i \rangle$ then $\forall i = 0 \dots q+1 \ \mathcal{AM}_i = \mathcal{AM}'_i$.

Intuitively, the documents have the same behavior if the automata accept the same sets of words and, for each possible path in the automata, the resulting states contain always the same set of active media.

The relationship of *behavioral equivalence* ($\sim$) is an equivalence relation. Indeed, it directly follows from the definition that it is *reflexive* (every document is behaviorally equivalent to itself: $D \sim D$), *symmetric* (if D is behaviorally equivalent to D', then D' is behaviorally equivalent to D: $D \sim D' \Rightarrow D' \sim D$), and *transitive* (if D is behaviorally equivalent to D' and D' is behaviorally equivalent to D'', then D is behaviorally equivalent to D'': $D \sim D'$ and $D' \sim D'' \Rightarrow D \sim D''$).

Figure 5 depicts the evolution of script 2. The activation of the presentation forces the activation of media *intro* and timer t'_1, that controls its termination, is inserted in $\mathcal{T}'_1$. At time instant 20, this timer expires: media *artwork* becomes active, while media *picture* is not activated because of its `begin` attribute (the corresponding timers t'_2 and t'_3, that control the termination and the beginning of *artwork* and *picture*, respectively, are inserted in $\mathcal{T}'_2$). After five seconds, the expiration of timer t'_3 forces the activation of media *picture* and timer t'_4 for its termination is inserted in $\mathcal{T}'_3$. At time instant 50, timer t'_2 expires and *artwork* ends; finally, the last timer t'_4 expires and forces the termination of *picture* and of the whole document.

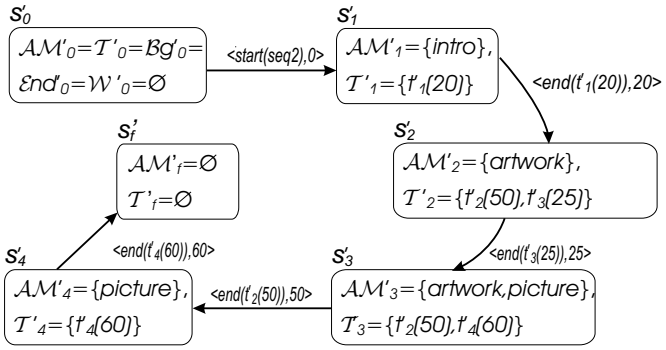


Figure 5. The evolution of script 2.

Let D and D' the multimedia documents corresponding to script 1 and script 2, respectively. Figures 2 and 5 represent the finite state automata $AUT(D)$ and $AUT(D')$ where the word $w = \langle \{start(par1)\}, 0 \rangle$,

$\langle \{end(t_2(20)),\ end(t_3(20))\}, 20 \rangle$, $\langle\{end(t_4(25))\}, 25\rangle$, $\langle\{end(t_5(50))\}, 50\rangle$, $\langle\{end(t_1(60))\}, 60\rangle$ is accepted by $AUT(D)$ with $ist_1 = 0$, $ist_2 = 20$, $ist_3 = 25$, $ist_4 = 50$ and $ist_5 = 60$ and the word $w' = \langle\{start(seq2)\}, 0\rangle$, $\langle\{end(t'_1(20))\}, 20\rangle$, $\langle\{end(t'_3(25))\}, 25\rangle$, $\langle\{end(t'_2(50))\}, 50\rangle$, $\langle\{end(t'_4(60))\}, 60\rangle$ is accepted by $AUT(D')$ with $ist'_1 = 0$, $ist'_2 = 20$, $ist'_3 = 25$, $ist'_4 = 50$ and $ist'_5 = 60$.

Therefore we have $\forall i = 1...5$, $ist_i = ist'_i$ and at every instant the events do not involve any element in $\mathcal{MI}$ since no media items spontaneously end, but they are controlled by a timer expiration; moreover, for each word accepted by the $AUT(D')$, the corresponding word in $AUT(D)$ is accepted, and viceversa. Then the first step of Definition 5.3 is verified.

Given w and w' accepted by the automata, we obtain the sequences of states $sq = s_o s_1 s_2 s_3 s_4 s_5$ and $sq' = s'_o s'_1 s'_2 s'_3 s'_4 s'_5$ reached by $AUT(D)$ and $AUT(D')$ (detailed in Figure 2 and 5) such that $\forall i = 0...5$, $\mathcal{AM}_i = \mathcal{AM}'_i$: the second step is verified therefore the two SMIL documents are *behaviorally equivalent* w.r.t. Definition 5.3.

6. A case study: building the automata for SMIL documents

Definition 4.3 can describe the behavior of multimedia documents designed with any synchronization model, provided the *next* function is properly defined.

In this paper we provide the definition of the *next* function needed to describe the behavior of SMIL scripts. SMIL functionalities are very complex and some of them do not change the temporal behavior of media items. For this reason, the proposed algorithms do not deal with transitions and animations. Moreover, the automaton does not capture the difference between the effective end of an element and the result of the `fill` attribute and its values `freeze` or `hold`. The same problems arise with the `excl` whose behavior, in some case, cannot be distinguished from a sequence (tag `seq`).

In order to define the state transformation function we need to define a set of auxiliary functions: for the lack of space, the algorithms are not complete, and the reader is referred to [3] for the missing code. Moreover, the following algorithms make use of the functions defined in Section 4 and resumed in Table I.

First of all, we need two functions **ChkBEGIN** and **ChkEND** which calculate the start and the end time of an element, and activate the suitable timers, updating the mapping functions *begin*, *stop* and *waiting*, following the W3C specification and algorithms for SMIL documents [9].

Another useful function is **NextEL** which, given a tag as input, returns the next element to playback, according to the synchronization constraints contained in the SMIL script.

The **ACTIVATE** function checks if a media item can start or if it must wait for a delay. In the first case it starts the element (function **ActEL**), otherwise it simply activates a timer (function **ChkBEGIN**).

The function **ActEL** adds the element to be activated to the set of active media $\mathcal{AM}$, if it is a media item; otherwise, it starts all its children (if such element is a `par` block), or only its first child (if it is a sequence of items).

ActEL (x: a SMIL Tag; $\mathcal{TS}$, $\mathcal{T}$: set of timers; t_{curr}: time instant;
$\mathcal{AM}$: set of media items; begin, stop: item-time mapping;
waiting: timer-media mapping)
begin
  $y = first(x)$;
  if $y =$ null **then** // x is a media item
    if ChkEND(x, $\mathcal{TS}$, $\mathcal{T}, t_{curr}, \mathcal{AM}, begin, stop, waiting$) **then exit** ;
    else $\mathcal{AM} = \mathcal{AM} \cup \{x\}$;
  else
    begin
      if $x =$ "seq" **then**
        if ChkEND(x, $\mathcal{TS}$, $\mathcal{T}, t_{curr}, \mathcal{AM}, begin, stop, waiting$) **then exit** ;
        else ACTIVATE (y, $\mathcal{TS}$, $\mathcal{T}, t_{curr}, \mathcal{AM}, begin, stop, waiting$)
      if $x =$ "par" **then**
        if ChkEND(x, $\mathcal{TS}$, $\mathcal{T}, t_{curr}, \mathcal{AM}, begin, stop, waiting$) **then exit** ;
        else for all $child \in children(x)$ **do**
          ACTIVATE($child$,$\mathcal{TS}$, $\mathcal{T}, t_{curr}, \mathcal{AM}, begin, stop, waiting$);
    end
end.

The function **DeACTIVATE** stops a single media object if it is contained in the set of active media $\mathcal{AM}$ using the function **StopMEDIA**. When applied to a `par` or a `seq` block, if this block was waiting for a timer, the next element, according to the synchronization rules, is started, otherwise all active elements in the block are deactivated.

The function **StopMEDIA** deactivates a media item removing it by the set $\mathcal{AM}$. If the media item is contained in a `par` block, the function checks if the block ends with this media (e.g., when its attribute `end` is equal to the end event of this media), and in this case it stops the whole block and activates the next element. If the media item is contained in a `seq` block, the function starts the next element according to the SMIL script.

StopMEDIA(x: a SMIL Tag; $\mathcal{TS}$, $\mathcal{T}$: set of timers; t_{curr}: time instant;

```
                    AM: set of media items; begin, stop: item-time mapping;
                    waiting: timer-media mapping)
begin
    AM = AM \ {x};
    y = parent(x);
    if y = null then exit ;
    if y = "par" then
        if y.end = "first" or y.end = x then
          begin
            DeACTIVATE(y,TS, T, t_curr, AM, begin, stop, waiting);
            y =NextEL(y);
            if y ≠ null then
                ACTIVATE(y,TS, T, t_curr, AM, begin, stop, waiting);
          end
        else
            if children(y) ∩ AM = ∅ and
                (stop(y) = null or stop(y) = t_curr) then
              begin
                y =NextEL(y);
                if y ≠ null then
                    ACTIVATE(y,TS, T, t_curr, AM, begin, stop, waiting);
              end
    else                    //y = "seq"
      begin
        if succ(x) ≠ null or (stop(y) = null or stop(y) = t_curr) then
          begin
            y =NextEL(x);
            if y ≠ null then
                ACTIVATE(y,TS, T, t_curr, AM, begin, stop, waiting);
          end
      end
end.
```

Definition 6.1. (State transition function) The state transition function $next : S \times \mathcal{E} \rightarrow S$, where S is the set of all possible states, and $\mathcal{E}$ is the set of events, is the function that, given a state s and a complex event $ev(i) = \langle \{e_k\}, i \rangle$ at the time instant i, returns the state $s' = next(s, ev(i))$ at the time instant $i + 1$ where $s' = \langle \mathcal{AM}', \mathcal{T}', \mathcal{B}g', \mathcal{E}nd', \mathcal{W}' \rangle$, and $\mathcal{AM}'$, $\mathcal{T}'$, $\mathcal{B}g'$, $\mathcal{E}nd'$, $\mathcal{W}'$ are defined according to the following process, in which $ev = \{e_k\}$ is set of occurring events, processed one at time, *el* the element to which the event applies and i is the current time instant:

```
for all e ∈ ev do
    case e = start(el):
        if el ∉ AM_n then
```

```
        ACTIVATE (el, TS, T, i, AM, begin, stop, waiting);
case e = end(el):
  begin
    if el ∈ AM and (stop(el) = null or stop(el) = i) then
      begin
        StopMEDIA (el, TS, T, i, AM, begin, stop, waiting);
        stop(el) = i;
      end
    if el ∈ T then
      begin
        if stop(m) = i for some m ∈ Tag[5] then
            for all item ∈ waiting(el) do
              begin
                DeACTIVATE (item, TS, T, i, AM, begin, stop, waiting);
                stop(item) = i;
              end
        if begin(m) = i for some m ∈ Tag[5] then
            for all item ∈ waiting(el) such that begin(item) = i do
                ActEL (item, TS, T, i, AM, begin, stop, waiting);
        waiting(el) = ∅;
        T = T \ {el};
      end
```

7. A canonical form for SMIL documents

In Section 5, we study the evolution of multimedia documents in the absence of user interactions. Considering the relation of behavioral equivalence ($\sim$) (Definition 5.3), we can classify multimedia documents according to their behavior along the time.

We can refer to the classes of equivalence induced by the relation $\sim$[6], in order to define a canonical form, i.e., a document which can be used as a "representative" for all the documents with the same temporal behavior.

If we call $AUT(\mathcal{DC})$ the set of automata describing the behavior of all possible multimedia documents $\mathcal{DC}$, we can consider these functions:

[5] Since in SMIL language also media items are defined using appropriate tags (`img`, `video`, or `audio` according to the object's type), media items are contained also in $\mathcal{T}ag$ and $\mathcal{MI} \subset \mathcal{T}ag$.

[6] Let $\mathcal{DC}$ be the set of all multimedia documents. Every element in the set $Q = \mathcal{DC}_{/\sim} = \{Q_1, \ldots, Q_n\}$ is a class of equivalence such that $\forall i = 1 \ldots n$ we have:

$$Q_i = \{D_{1i}, \ldots, D_{ki} \mid D_{ji} \sim D_{ri}, j = 1 \ldots k, r = 1 \ldots k\}.$$

– *Automaton* : $\mathcal{DC} \rightarrow \mathcal{AUT}(\mathcal{DC})$, which returns the automaton describing the behavior of a multimedia document;

– *Representative* : $AUT(\mathcal{DC}) \rightarrow \mathcal{DC}$, which extracts the representative of a class of multimedia documents.

In our case study, the SMIL language, function *Automaton* is implemented, event by event, by the *next* function defined in Definition 6.1.

Definition 7.1. (Canonical Form) Let D be a multimedia document $\in \mathcal{DC}$. The canonical form of D is the document D' such that:

$$Representative(Automaton(D)) = D'.$$

A document D is in the canonical form iff $D = Representative(\ Automaton(D))$. If two documents have the same canonical form, they are equivalent since their behaviors are described by the same automaton.

The algorithm for the extraction of a SMIL script, which can be considered the representative of an equivalence class, is very complex and cannot be entirely presented here due to space constraints. The reader is referred to [3] for the complete discussion about the algorithm.

This algorithm needs some auxiliary functions which allow to create a new SMIL tag (**CreateELEM**), defining a single media with its attribute (e.g., `begin`, `src`, etc.), and to insert a set of media items into a new parallel (**CreatePAR**) or sequential (**CreateSEQ**) block. A media item can also be inserted into an already existing block using the function **Insert**: if media items are inserted into a `par` block of already active media, an offset must be defined, otherwise the tag is inserted in the last position of the given sequence block.

A SMIL script, which is in the canonical form, contains a `dur` or an `end` attribute for each media item defined: the algorithm does not distinguish between continuous media, i.e., media items with a natural evolution, and static media like images and text files which need a duration, because this information is not deducible from the automaton. Function **DefStopAttr** is used to insert the appropriate attributes `end` and `dur` to each media item, according to the event of the timer which causes its termination.

Let us consider the graph representing the automaton of a multimedia presentation and the shortest path on it, connecting the initial state s_0 to the final state s_f[7]. Given two states, s_i and s_{i+1}, they are connected by the edge labelled with the event $\langle\{e_1(m_1), \ldots, e_n(m_n)\}, i\rangle$, where i represents the time instant in which the event occurs.

[7] If more than one path have the same length, the choice is made randomly.

For each transition, we can define the following sets:

- $Started = \mathcal{AM}_i \setminus \mathcal{AM}_{i-1}$, containing the media activated by the events which cause the transition;
- $Stopped = \mathcal{AM}_{i-1} \setminus \mathcal{AM}_i$, containing the media that are stopped by the transition;
- $Inv = \mathcal{AM}_{i-1} \setminus Stopped$, containing the media that remain active.

Different combinations of the status of these sets represent the different situations that have to be dealt with to create a correct SMIL script. We define a function that, given an automaton representing the evolution of a document, builds the SMIL script representing the canonical form for the desiderate behavior: the algorithm calculates, for each transition, the sets introduced previously, in order to characterize the different situations, and afterwards creates the corresponding SMIL tags.

Given the sequence $s_0 \dots s_i \dots s_f$ which represents the states in the shortest path of the automaton, where s_0 and s_f are empty states, i.e., states in which neither media items nor timers are active, we have to apply the following algorithm:

$s_i = s_0$; $k = 0$;
//*next* RETURNS THE NEXT STATE IN THE SEQUENCE AND k IS A COUNTER
repeat
  $s_f = s_i$.next;
  //DEFINITION OF THE TIME INSTANT
  if $\exists ev | s_i \stackrel{ev}{\rightarrow} s_f$ **and** $ev = \langle e(m), ist \rangle$ **then**
    $time = ist$; //*time* IS THE INSTANT IN WHICH THE EVENT OCCURS
  //DEFINITION OF THE MEDIA SETS
  $Started = \mathcal{AM}_f \setminus \mathcal{AM}_i$; //*Started* CONTAINS THE STARTED MEDIA
  $Stopped = \mathcal{AM}_i \setminus \mathcal{AM}_f$; //*Stopped* CONTAINS THE STOPPED MEDIA
  $Inv = \mathcal{AM}_i \setminus Stopped$; //*Inv* CONTAINS THE ALREADY ACTIVE ITEMS
  if $Started \neq \emptyset$ **then**
    if $Stopped = \emptyset$ **then**
      if $Inv = \emptyset$ **then**
        Case 1: $Started \neq \emptyset$ and $Stopped = \emptyset$ and $Inv = \emptyset$
        there exist only started media, i.e., the document
        begins; a parallel block is created containing media
        items in $Started$ (**CreatePAR**($Started$,k,NULL,$time$))
        or a single tag if $|Started| = 1$ (**CreatELEM**(m, NULL,
        $time$))
      else
        Case 2: $Started \neq \emptyset$ and $Stopped = \emptyset$ and $Inv \neq \emptyset$
        some media items are already active, while others are

beginning; two parallel blocks are created, **CreatePAR**(*Started*,*k*,NULL,*time*) (if $|Started| \neq 1$) and **CreatePAR**(*Inv*,*k*,NULL,*time*) (if it does not already exist); `par` block containing *Started*, or the single media item, is inserted into the `par` block containing *Inv* with an appropriate offset

else

Case 3: $Started \neq \emptyset$ and $Stopped \neq \emptyset$ and ($Inv = \emptyset$ or $Inv \neq \emptyset$) some media items start, while others end; a sequence is created by function **CreateSEQ**(m,k,$parent(m)$,$begin(n)$) containing first media items in *Stopped* (eventually in a `par` block) and then media items in *Started*

else

if $Stopped = \emptyset$ **then**

Case 4: $Started = \emptyset$ and $Stopped = \emptyset$ and $Inv \neq \emptyset$ this transition does not affect the set of active media items; an attribute "`repeat`" is added to media item which causes the transition

else

Case 5: $Started = \emptyset$ and $Stopped \neq \emptyset$ and $Inv \neq \emptyset$ some media end, while others stay active; the function **DefStopAttr** defines the attributes "`end`" or "`dur`" of ended media items

if $Inv = \emptyset$ **then**

Case 6: $Started = \emptyset$ and $Stopped \neq \emptyset$ and $Inv = \emptyset$ there exist only terminating media, i.e., the document ends; the function **DefStopAttr** defines the attributes "`end`"or "`dur`" of ended media items

$s_i = s_f$;

until $s_i \in Final$

We now introduce an example to show how these algorithms are used. Let us consider a multimedia presentation available at the web site of a company to give some information about it. The SMIL script 3 is divided in three sections: an introduction, a little guide tour through the company, and the conclusion.

The introduction contains some news about the company presented with a video, *vIntro*, and a caption, *cIntro*, and another video (*vLoc*) with its caption (*cLoc*) that show how to reach the company. While these media are active, a soundtrack (*sound*) plays.

Figure 6 describes the temporal behavior of the presentation. After the introduction, the user is guided through the departments of the company by a video clip and an audio comment (respectively *vManuf* and *aManuf* for the manufacturing department and *vAcc* and *cAcc* for the accounting department). At the end, an image (*iConcl*) and an audio track (*aConcl*) give the contact information.

The same behavior can be expressed in a different manner even using the SMIL language. For example, script 4 describes the same temporal execution depicted in Figure 6.

More formally, we can apply our definition of behavioral equivalence to check if both the scripts 3 and 4 belong to the same class of equivalence.

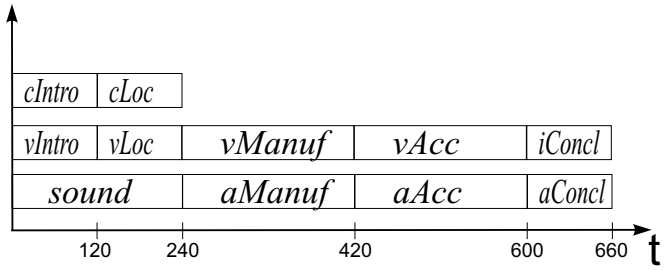


Figure 6. Timeline of the presentation about a company.

The evolution of script 3 is represented by means of the automaton depicted in Figure 7: from the initial state, in which neither media nor timers are active, the activation of the `seq` block causes the activation of the audio *sound*, the video *vIntro* and the text *cIntro*, while the media *vLoc* and *cLoc* are not activated because of their `begin` attributes. Five timers are inserted in $\mathcal{T}_1$: t_1 controls the termination of media *sound*, t_2 and t_3 the termination of *vIntro* and *cIntro*, respectively, and, finally, t_4 and t_5 control the beginning of *vLoc* and *cLoc*. At time instant 120, timers t_2, t_3, t_4 and t_5 expire: as a consequence, media *vIntro* and *cIntro* end and media *vLoc* and *cLoc* become active. At time instant 240 the expiration of t_1 forces the termination of all active media items; `par1` ends and the sequence activates `par2`, i.e., the video *vManuf* and the audio *cManuf* start, (activating timer t_6 that controls their termination). At time instant 420, when timer t_6 expires, media *vManuf* and *cManuf* end, while *vAcc* and *cAcc* begin (activating timer t_7 that controls their termination). At time instant 600, the expiration of t_7 forces the termination of the two active media and the beginning of *aConcl* and *iConcl*: t_8 is inserted in $\mathcal{T}_5$ and, after 60 seconds, its expiration causes the termination of the whole document.

Script 3

```
⟨ seq id="seq0" ⟩
 ⟨ par id="par1" end="sound.end" ⟩
  ⟨ audio id="sound" dur="240s"/ ⟩
  ⟨ video id="vIntro" dur="120s"/ ⟩
  ⟨ text id="cIntro" end="vIntro.end"/ ⟩
  ⟨ video id="vLoc" begin="vIntro.end"/ ⟩
  ⟨ text id="cLoc" begin="vIntro.end"/ ⟩
 ⟨ /par ⟩
 ⟨ par id="par2" dur="180s" ⟩
  ⟨ video id="vManuf"/ ⟩
  ⟨ audio id="aManuf"/ ⟩
 ⟨ /par ⟩
 ⟨ par id="par3" dur="180s" ⟩
  ⟨ video id="vAcc"/ ⟩
  ⟨ audio id="aAcc"/ ⟩
 ⟨ /par ⟩
 ⟨ par id="par4" dur="60s" ⟩
  ⟨ audio id="aConcl"/ ⟩
  ⟨ image id="iConcl"/ ⟩
 ⟨ /par ⟩
⟨ /seq ⟩
```

Script 4

```
⟨ par id="par0" ⟩
 ⟨ seq id="seq4" ⟩
  ⟨ audio id="sound" dur="240s"/ ⟩
  ⟨ par id="par5" dur="180s" ⟩
   ⟨ video id="vManuf" dur="180s"/ ⟩
   ⟨ text id="aManuf" dur="180s"/ ⟩
  ⟨ /par ⟩
  ⟨ par id="par6" dur="180s" ⟩
   ⟨ video id="vAcc" dur="180s"/ ⟩
   ⟨ text id="aAcc" dur="180s"/ ⟩
  ⟨ /par ⟩
  ⟨ par id="par7" dur="60s" ⟩
   ⟨ audio id="aConcl" dur="60s"/ ⟩
   ⟨ image id="iConcl" dur="60s"/ ⟩
  ⟨ /par ⟩
 ⟨ /seq ⟩
 ⟨ seq id="seq2" ⟩
  ⟨ par id="par1" dur="120s" ⟩
   ⟨ video id="vIntro" dur="120s"/ ⟩
   ⟨ text id="cIntro" dur="120s"/ ⟩
  ⟨ /par ⟩
  ⟨ par id="par3" ⟩
   ⟨ video id="vLoc" dur="120s"/ ⟩
   ⟨ text id="cLoc" dur="120s"/ ⟩
  ⟨ /par ⟩
 ⟨ /seq ⟩
⟨ /par ⟩
```

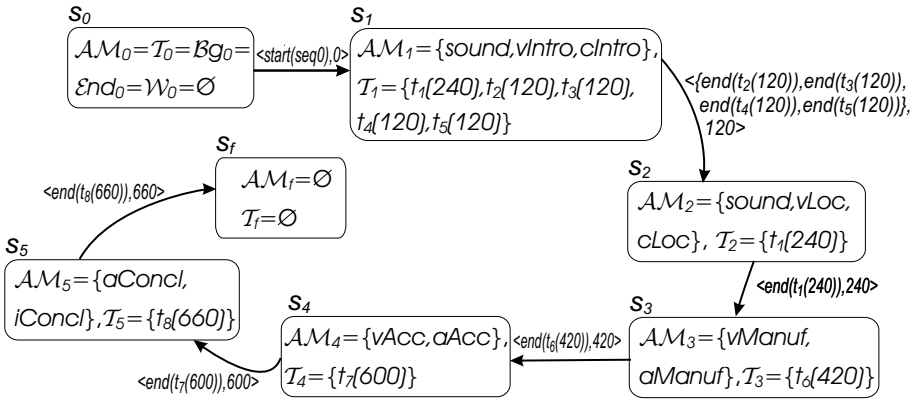


Figure 7. Automaton representing the evolution of script 3.

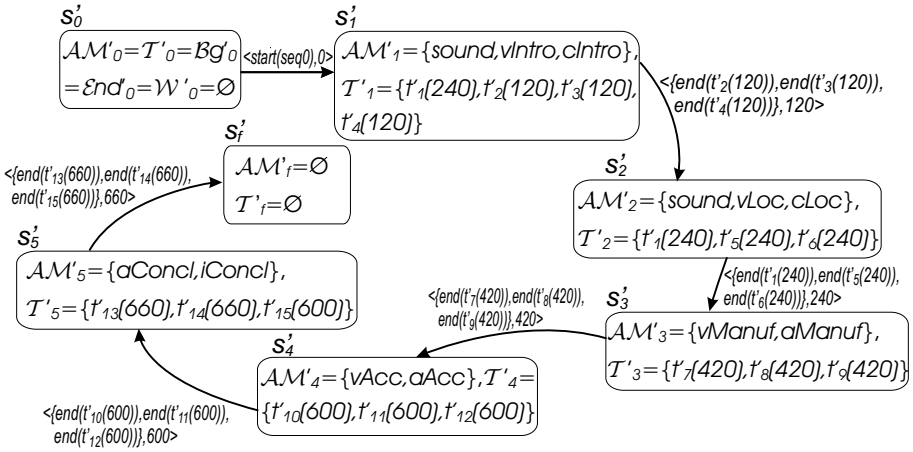


Figure 8. Automaton representing the evolution of script 4.

The evolution of script 4 is represented by means of the automaton depicted in Figure 8. If we consider the sequences of events which build up the accepted words of the automata depicted in Figures 7 and 8, we note that the automata step to another state at the same time instants, i.e., $ist_0 = 0$, $ist_1 = 120$, $ist_2 = 240$, $ist_3 = 420$, $ist_4 = 600$ and $ist_5 = 660$.

Therefore we have $\forall i = 1...5$, $ist_i = ist'_i$ and at every instant the events do not involve any element in $\mathcal{MI}$ since no media items spontaneously end, but they are controlled by a timer expiration; moreover, for each word accepted by the AUT(script 4), the corresponding word in AUT(script 3) is accepted, and viceversa. Then the first step of Definition 5.3 is verified.

Given w and w' the words accepted by the two automata, we obtain the sequences of states $sq = s_o s_1 s_2 s_3 s_4 s_5 s_6$ and $sq' = s'_o s'_1 s'_2 s'_3 s'_4 s'_5 s'_6$ reached, respectively, by AUT(script 3) and AUT(script 4), such that $\forall i = 0...6$, $\mathcal{AM}_i = \mathcal{AM}'_i$: the second step is verified and the two documents are *behaviorally equivalent* w.r.t. Definition 5.3.

Scripts 3 and 4 are contained in the same class of equivalence, since they have the same natural evolution: then we can build the canonical form, S_{CF}, which can be used as the representative for the behavior of both the SMIL scripts.

Consider the automaton in Figure 7; for every edge in AUT(script 3) we apply the algorithm defined in this section.

The first event, $\langle start(seq0), 0\rangle$, represents the beginning of the presentation (case 1): the three starting media ($sound$, $vIntro$, $cIntro$) are nested in a parallel block (`par0`). Then four timers end at time instant 120. Case 3 is verified since media items $vIntro$ and $cIntro$ end while

vLoc and *cLoc* start: two different parallel blocks are created (`par1` and `par3`) and inserted into a sequence (`seq2`). In the same way, the algorithm build three other par blocks containing media items playing in parallel ($vManuf$ and $aManuf$, *vAcc* and *aAcc*, and *iConcl* and *aConcl*) and put them into a sequence `seq4`.

At the last event $\langle\{end(t_9)\}, 660\rangle$, the algorithm defines the termination attributes for the ending media (case 6). The resulting script is equal to script 4, i.e., 4 is in the canonical form.

8. Conclusion

In this paper we study the problem of comparing the evolution along time of multimedia documents and we define the notions of temporal *inclusion*, *intersection* and *equivalence*.

Although other works in literature address similar problems, our approach is interesting since it defines a general formalism, independent from the design model, to describe the temporal behavior of multimedia presentations in absence of user interactions. Using this approach, two documents can be compared as long as their behaviors can be expressed through an automaton. We can argue that this is always possible: some examples are commented in Section 2 and in [5], the authors propose the use of automaton to describe the temporal evolution of a document to build up a complete fragment to be returned as result of a query. In the future, we plan to better investigate this issue, defining suitable transition functions for different kinds of multimedia reference models.

In this paper we adopt the SMIL language as a case study and we provide the algorithms to build up the automaton of a SMIL document and, if two SMIL scripts have the same behavior, i.e., they belong to the same *equivalence class*, to extract the canonical form which represents the evolution of that class of scripts. Due to space constraints, the paper presents only a rough sketch of the proposed algorithms; a complete description can be found in [3].

In this paper, we propose a set of definitions that can be useful to compare multimedia documents, but we do not discuss the cost of this operation. Since the automaton which represents the evolution of a multimedia document can be considered as a particular type of oriented and labelled graph, the study of the complexity of this operation can be reduced to a graph matching problem, for which, well-known algorithms exist in literature, which can be easily adapted to our automata.

In the future we plan to extend our definition of intersection considering a notion of maximality: actually, two documents intersects as long as they share a pair of media items played in sequence. Our

notion of intersection, in fact, aims at discovering a minimal set of common components and behavior, and do not investigate any more. The documents, however, could have other common components. Our future goal is the definition of an algorithm to identify the maximum intersection between two multimedia presentations.

References

1. J. F. Allen. Maintaining Knowledge about Temporal Intervals. Comm. ACM, 26(11):832–843, November 1983.
2. R. Alur and D.L. Dill. A Theory of Timed Automata. Theoretical Computer Science, 126:183–235, 1994.
3. P. Bertolotti and O. Gaggi. A Study on Multimedia Documents Behavior: a Notion of Equivalence. Technical Report 82/05, Department of Computer Science, University of Turin, Italy, January 2005, http://www.di.unito.it/∼bertolot/tech-report-mtap04.pdf.
4. S. Boll, W. Klas, and J. Wandel. A Cross-Media Adaptation Strategy for Multimedia Presentations. In Proc. of Multimedia Conference 1999, Orlando, Florida, USA, October 30-November 5 1999.
5. A. Celentano, O. Gaggi, and M. L. Sapino. Retrieval in multimedia presentations. Multimedia Systems Journal, 10(1):72–82, 2004.
6. L. Hardman, D.C.A. Bulterman, and G. van Rossum. The Amsterdam Hypermedia Model: Adding Time, Structure and Context to Hypertext. Communications of the ACM, 37(2):50–62, February 1994.
7. J.H. Hausmann, R. Heckel, and S. Sauer. Dynamic Meta Modeling with Time: Specifying the Semantics of Multimedia Sequence Diagrams. In Proc. of Workshop on Graph Transformation and Visual Modelling Techniques (GT-VMT 2002), Barcelona, Spain, October 11-12 2002.
8. M. Jourdan, N. Layaïda, C. Roisin, L. Sabry-Ismail, and L. Tardif. Madeus, an Authoring Environment for Interactive Multimedia Documents. In ACM Multimedia 1998, pages 267–272, Bristol, UK, September 1998.
9. Synchronized Multimedia Working Group of W3C. Synchronized Multimedia Integration Language (SMIL) 2.0 Specification. http://www.w3.org/TR/smil20, August 2001.
10. S. Lo Presti, D. Bert, and A. Duda. TAO: Temporal Algebraic Operators for modeling multimedia presentations. Journal of Network and Computer Applications, 25(4):319–342, October 2002.
11. P.N.M. Sampaio and J.-P. Courtiat. A Formal Approach for the Presentation of Interactive Multimedia Documents. In Proc. of ACM Multimedia 2000 (short paper), Los Angeles, USA, October 30, November 4 2000.
12. P. D. Stotts, R. Furuta, and C. R. Cabarrus. Hyperdocuments as Automata: Verification of Trace-based Browsing Properties by Model Checking. ACM Transactions on Information Systems, 16(1):1–30, January 1998.
13. C.-C. Yang. Detection of the time conflicts for SMIL-based Multimedia Presentation. In Proc. of 2000 Computer Symposium (ICS-2000)-Workshop on Computer Networks, Internet, and Multimedia, pages 57–63, Chiayi, Taiwan, 2000.