A SPATIO-TEMPORAL SEMANTIC MODEL FOR MULTIMEDIA PRESENTATIONS AND MULTIMEDIA DATABASE SYSTEMS

Shu-Ching Chen, Member, IEEE and R. L. Kashyap, Fellow, IEEE

ABSTRACT

An abstract semantic model based on augmented transition network (ATN) to model multimedia presentations and temporal/spatial multimedia database searching is presented in this paper. The inputs for ATNs are modeled by regular expressions. Regular expressions provide an efficient means for iconic indexing of the temporal/spatial relations of media streams and semantic objects. An ATN and its subnetworks are used to represent the appearing sequence of media streams and semantic objects. The arc label is a substring of a regular expression. In this design, a presentation is driven by a regular expression. Each subnetwork has its own regular expression. Database queries relative to text, image, and video can be answered via substring matching at subnetworks. Users have the flexibility to select any part of the presentation to watch by issuing approximate database queries. This means that ATN and its subnetwork can be included in the multimedia database systems which is controlled by a database management system (DBMS). User interactions, loops, and embedded presentations are also provided in ATN.

Key words: Multimedia Presentations, Multimedia Database Systems, Augmented Transition Network (ATN), Regular Expression, Semantic Object.

• The authors are with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907-1285. E-mail:{shuching, kashyap}@ecn.purdue.edu.

1. INTRODUCTION

In multimedia systems, a variety of information sources – text, voice, image, audio, animation, and video – are delivered synchronously or asynchronously via more than one device. The important characteristic of such a system is that all of the different media are brought together into one single unit, all controlled by the computer. Normally, multimedia systems require the management and delivery of extremely large bodies of data at very high rates and may require the delivery with real-time constraints. A semantic model is needed to handle the temporal and spatial requirements and the rich semantics of multimedia data.

In order to keep the rich semantic information, abstract models are developed to let users specify the temporal and spatial requirements at the time of authoring the objects, and to store a great deal of useful information (such as video clip start/end time, start/end frame number, and semantic objects relative spatial locations). Also, a semantic model should provide a pictorial form to increase the information content so that the cognitive load on the users is reduced. Some researchers have proposed models for this purpose such as petri-net, time-interval based, timeline, and graphic.

Some of the semantic models developed in the recent past are graphical structures to model the multimedia presentations. Semantic models such as OCPN [14], MDS [6], Firefly [4], and Hirzalla et al's graphical temporal model [11] all fall into this category. In other multimedia database systems [16] [17] [9], the emphasis is to present different media streams to users with query specifications. These models provide the searching capabilities to allow users to retrieve information from the database.

Augmented transition network (ATN), developed by Woods [21], has been used in natural language understanding systems and question answering systems for both text and speech. We use the ATN as a semantic model to model a multimedia presentation and the temporal, spatial, or spatio-temporal relations of various media streams and semantic objects. A multimedia presentation consists of a sequence

of media streams displaying together or separately across the time. The arcs in an ATN represent the time flow from one state to another. ATN differs from a finite state automata in that it permits recursion. Each nonterminal symbol consists of a subnetwork which can be used to model the temporal and spatial information of semantic objects for images and video frames. In addition, a subnetwork can represent another existing presentation. Any change in one of the subnetworks will automatically change the presentation which includes these subnetworks. To design a multimedia presentation from scratch is a painful process in today's authoring environment. The subnetworks in ATN allow the designers to use the existing presentation sequence in the archives, which makes ATN a powerful model to create a new presentation. This is similar to the *class* in the object-oriented paradigm. Also, subnetworks can model keywords in a text media stream so that database queries relative to the keywords in the text can be answered.

Conditions and actions in the arcs in ATNs maintain the synchronization and quality of service (QoS) of a multimedia presentation. In an interactive multimedia presentation, users may want to see different presentation sequences from the originally specified sequence. Therefore, in our design, when a user issues a database query, the specification in the query tries to match the conditions in the arcs. If a condition is matched then the corresponding action is invoked. Different actions can generate different presentation sequences which are different from the original sequence.

When an ATN is used for language understanding, the input for an ATN is a sentence which consists of a sequence of words with linear order. In a multimedia presentation, when user interactions such as user selections and loops are allowed, then we cannot use sentences to be inputs for an ATN. In our design, each arc in an ATN is a string containing one or more media streams displayed at the same time. A media stream is represented by a letter subscripted by some digits. This single letter represents the media stream type and the digits are used to denote various media streams of the same media stream type. For example, T_1 means a text media stream with identification number one. A regular expression

consists of one or more media streams and is used as an input for an ATN. Regular expressions also have the power to model the "or" conditions and the iterative conditions. Since the heart of an ATN is a finite state automata, any regular expression can be represented by an ATN.

The organization of this paper is as follows. Section 2 discusses how to use an ATN to model a multimedia presentation and to incorporate with multimedia database searching. The input for an ATN which is modeled by regular expressions is illustrated in section 3. Section 4 shows the examples which use ATNs and regular expressions to model multimedia presentations and database queries. Related work are in section 5. Conclusions are in section 6.

2. THE AUGMENTED TRANSITION NETWORK (ATN)

A multimedia environment should not only display media streams to users but also allow two-way communication between users and the multimedia system. The multimedia environment consists of a multimedia presentation system and a multimedia database system. If a multimedia environment has only a presentation system but without a multimedia database system then it is like a VCR or a TV without feedback from user. A multimedia database system allows users to specify queries for information. The information may be relative to text data as well as image or video content. By combining multimedia presentation and multimedia database system, users can specify queries which reflect what they want to see or know. A semantic model that models the presentation has the ability to check the features specified by users in the queries, and maintains the synchronization and QoS desired.

A finite state machine (FSM) consists of a network of nodes and directed arcs connecting them. The FSM is a simple transition network. Every language that can be described by an FSM can be described by a regular grammar, and vice versa. The nodes correspond to states and the arcs represent the transitions from state to state. Each arc is labeled with a symbol whose input can cause a transition from the state at the tail of the arc to the state at its head. This feature makes FSM have the ability to model a

presentation from the initial state to some final states or to let users watch the presentation fast forward or reverse. However, users may want to watch part of a presentation by specifying some features relative to image or video contents prior to a multimedia presentation, and a designer may want to include other presentations in a presentation. These two features require a pushdown mechanism that permits one to suspend the current process and go to another state to analyze a query that involves temporal, spatial, or spatio-temporal relationships. Since FSM does not have the mechanism to build up the hierarchical structure, it cannot satisfy these two features.

This weakness can be eliminated by adding a recursive control mechanism to the FSM to form a *recursive transition network* (RTN). A recursive transition network is similar to an FSM with the modifications as follows: all states are given names which are then allowed as part of labels on arcs in addition to the normal input symbols. Based on these labels, subnetworks may be created. Three situations can generate subnetworks. In the first situation, when an input symbol contains an image or a video frame, a subnetwork is generated. A new state is created for the subnetwork if there is any change of the number of semantic objects or any change of the relative position. Therefore, the temporal, spatial, or spatio-temporal relations of the semantic objects are modeled in this subnetwork. In other words, users can choose the scenarios relative to the temporal, spatial, or spatio-temporal relations of the video or image contents that they want to watch via queries. Second, if an input symbol contains a text media stream, the keywords in the text media stream become the input symbols of a subnetwork. A keyword can be a word or a sentence. A new state of the subnetwork is created for each keyword. Keywords are the labels on the arcs. The input symbols of the subnetwork have the same order as the keywords appear in the text. Users can specify the criteria based on a keyword or a combination of keywords in the queries. In addition, the information of other databases can be accessed by keywords via the text subnetworks. For example, if a text subnetwork contains the keyword "Purdue University Library" then the Purdue University library database is linked via a query with this keyword. In this

design, an ATN can connect multiple existing database systems by passing the control to them. After exiting the linked database system, the control is back to the ATN. Third, if an ATN wants to include another existing presentation (ATN) as a subnetwork, the initial state name of the existing presentation (ATN) is put as the arc label of the ATN. This allows any existing presentations to be embedded in the current ATN to make a new design easier. The advantage is that the other presentation structure is independent of the current presentation structure. This makes both the designer and users have a clear view of the presentation. Any change in the shared presentation is done in the shared presentation itself. There is no need to modify those presentations which use it as a subnetwork.

Before the control is passed to the subnetwork, the state name at the head of the arc is pushed into the push-down store. The analysis then goes to the subnetwork whose initial state name is part of the arc label. When a final state of the subnetwork is reached, a pop occurs and the control goes back to the state removed from the top of the push-down store. Examples to illustrate the process will be demonstrated in Section 4.

However, the FSM with recursion cannot describe cross-serial dependencies. For example, network delays may cause some media streams not to be displayed to users at the tentative start time and the preparation time for users to make decisions is unknown when user interactions are provided. In both situations, there is a period of delay which should be propagated to the later presentations. Also, users may specify queries related to semantic objects across several subnetworks. The information in each subnetwork should be kept so that the analysis across multiple subnetworks can be done. For example, the temporal, spatial, or spatio-temporal relations among semantic objects may involve several video subnetworks. The cross-serial dependencies can be obtained by specifying conditions and actions in each arc. The arrangement of states and arcs represents the surface structure of a multimedia presentation sequence. If a user wants to specify a presentation which may be quite different from the surface structure then the actions permit rearrangements and embeddings, and control the synchronization and quality

of service of the original presentation sequence. The cross-serial dependencies are achieved by using *variables* and they can be used in later actions or subsequent input symbols to refer to their values. The actions determine additions, subtractions, and changes to the values of *variables* in terms of the current input symbol and conditions. Conditions provide more sensitive controls on the transitions in ATNs. A condition is a combination of checkings involving the feature elements of media streams such as the start time, end time, etc. An action cannot be taken if its condition turns out to be false. Thus more elaborate restrictions can be imposed on the current input symbol for synchronization and quality of service controls. Also, information can be passed along in an ATN to determine future transitions. The recursive FSM with these additions forms an *augmented recursive transition network* (ATN).

3. FORMULATION OF INPUT SYMBOLS USING REGULAR EXPRESSIONS

Originally, an ATN is used for the analysis of natural language sentences. Its input is a sentence composed of words. This input format is not suitable to represent a multimedia presentation since several media streams need to be displayed at the same time, to be overlapped, to be seen repeatedly, etc.

Regular expressions [12] are useful descriptors of patterns such as tokens used in a programming language. Regular expressions provide convenient ways of specifying a certain set of strings. In this study, these strings are used to represent the presentation sequences of the temporal media streams, spatio-temporal relations of semantic objects, and keyword compositions. Information can be obtained with low time complexity by analyzing these strings. Regular expression goes from the left to right which can represent the time sequence of a multimedia presentation but it cannot represent concurrent appearance and spatial location of media streams and semantic objects. In order to let regular expressions have these two abilities, several modifications are needed. There are two levels need to be represented by regular expressions. At the coarse-grained level, the main presentation which involves media streams is modeled. At the fine-grained level, the semantic objects in image or video frames and the keywords in a

text media stream are modeled at subnetworks. Each keyword in a text media stream is the arc label at subnetworks. New states and arcs are created to model each keyword. The details to model each level are discussed in the follows subsections.

3.1. Using Regular Expression to Model Media Streams

Two notations $\mathcal{L}$ and $\mathcal{D}$ are used to define regular expressions and are defined as follows:

$\mathcal{L}$ = {A, I, T, V} is the set whose members represent the media type, where A, I, T, V denote audio, image, text, and video, respectively.

$\mathcal{D}$ = {0, 1, ..., 9} is the set consisting of the set of the ten decimal digits.

Definition 1: Each input symbol of a regular expression contains one or more media streams which are enclosed by a parentheses and are displayed at the same time interval. A media stream is a string which begins with a letter in $\mathcal{L}$ subscripted by a string of digits in $\mathcal{D}$. For example, V_1 represents video media stream and its identification number is one. The same video or audio media stream may appear in more than one consecutive input symbol and a superscripted string of digits is used to distinguish them such as V_1^1, V_1^2, and so on. The following situations can be modeled by a regular expression.

- **Concurrent:** The symbol "&" between two media streams indicates these two media streams are displayed concurrently. For example, $(T_1 \& V_1)$ represents T_1 and V_1 to be displayed concurrently.

- **Looping:** $m^+ = \bigcup_{i=1}^{\infty} m^i$ is the regular expression of positive closure of m to denote m occurring one or more times. We use the "+" symbol to model loops in a multimedia presentation to let some part of the presentation be displayed more than once.

- **Optional:** In a multimedia presentation, when the network becomes congested the original specified media streams which are stored in the remote server might not be able to arrive on time. The designer can use "*" symbol to indicate the media streams which can be dropped in the on-line

presentation. For example, $(T_1 \& V_1^*)$ means T_1 and V_1 will be displayed but V_1 can be dropped if some criteria cannot be met.

- **Contiguous:** Input symbols which are concatenated together are used to represent a multimedia presentation sequence and to form a regular expression. Input symbols are displayed from left to right across time sequentially. ab is the regular expression of a concatenated with b such that b will be displayed after a is displayed. For example, $(A_1 \& T_1)(A_2 \& T_2)$ consists of two input symbols $(A_1 \& T_1)$ and $(A_2 \& T_2)$. These two input symbols are concatenated together to show that the first input symbol $(A_1 \& T_1)$ is displayed before the second input symbol $(A_2 \& T_2)$.

- **Alternative:** A regular expression can model user selections by separating input symbols with the "|" symbol. So, $(a|b)$ is the regular expression of a *or* b. For example, $((A_1 \& T_1) \mid (A_2 \& T_2))$ denotes either the input symbol $(A_1 \& T_1)$ or the input symbol $(A_2 \& T_2)$ to be displayed.

3.2. Using Regular Expression to Model Semantic Objects in the Subnetworks of ATN

The temporal and spatial relations of semantic objects in video frames are modeled in the subnetworks in an ATN. Regular expression is a left to right model which can model the temporal relations of semantic objects. The semantic objects in the left input symbol appear earlier than those in the right input symbol in a video sequence. The spatial locations of semantic objects also need to be defined so that the queries relative to spatial locations can be answered. In our design, the temporal and spatial relations of semantic objects of a video stream in each input symbol can be modeled by a regular expression. User queries can be answered by analyzing the regular expression (for example, the movement, the relative spatial location, the appearing sequence, etc. of semantic objects). Spatial location of each semantic object needs to be represented by a symbolic form in order to use regular expressions to represent it.

Spatial data objects often cover multi-dimensional spaces and are not well represented by point

locations. It is very important for a database system to have an index mechanism to handle spatial data efficiently, as required in computer aided design, geo-data applications, and multimedia applications. R-tree [10] was proposed as a natural extension of B-trees [2] [7] and combines the nice features of both B-trees and quadtrees. R-tree is a height-balanced tree similar to a B-tree. The spatial objects are stored in the leaf level and are not further decomposed into their pictorial primitives, i.e., into quadrants, line streams, or pixels. We call this spatial object a *"semantic object".* We adopt the minimal bounding rectangle (MBR) concept in R-tree so that each semantic object is covered by a rectangle. Three types of topological relations between the MBRs can be identified [5]:

(1) nonoverlapping rectangles;

(2) partly overlapping rectangles;

(3) completely overlapping rectangles.

For the second and the third alternatives, *orthogonal relations* proposed by Chang et al. [5] can be used to find the *relation objects.* In this section, we consider only the first alternative, the nonoverlapping rectangles.

Automatic segmentation and recognition of semantic objects are outside the scope of this paper. We assume that each image or video frame has been segmented automatically or identified manually. The main focus in this paper is how to model the semantic objects so that the queries related to these semantic objects can be answered quickly.

Definition 2: Let O be a set of n semantic objects, O $= (o_1, o_2, \ldots, o_n)$. Associated with each o_i, $\forall i, (1 \leq i \leq n)$, is an MBR_i which is a minimal bounding rectangle containing the semantic object. In a 3-D space, an entry MBR_i is a rectangle between points $(x_{low}, y_{low}, z_{low})$ and $(x_{high}, y_{high}, z_{high})$. The centroid is used to be a reference point for the spatial reasoning.

As mentioned in the previous section, regular expressions are used to represent the temporal relations

Table 1: Three dimensional relative positions for semantic objects: The first and the third columns indicate the relative position numbers while the second and the fourth columns are the relative coordinates. (x_t, y_t, z_t) and (x_s, y_s, z_s) represent the X-, Y-, and Z-coordinates of the target and any semantic object, respectively. The "$\approx$" symbol means the difference between two coordinates is within a threshold value.

Number	**Relative Coordinates**	**Number**	**Relative Coordinates**
1	$x_s \approx x_t, y_s \approx y_t, z_s \approx z_t$	15	$x_s < x_t, y_s < y_t, z_s > z_t$
2	$x_s \approx x_t, y_s \approx y_t, z_s < z_t$	16	$x_s < x_t, y_s > y_t, z_s \approx z_t$
3	$x_s \approx x_t, y_s \approx y_t, z_s > z_t$	17	$x_s < x_t, y_s > y_t, z_s < z_t$
4	$x_s \approx x_t, y_s < y_t, z_s \approx z_t$	18	$x_s < x_t, y_s > y_t, z_s > z_t$
5	$x_s \approx x_t, y_s < y_t, z_s < z_t$	19	$x_s > x_t, y_s \approx y_t, z_s \approx z_t$
6	$x_s \approx x_t, y_s < y_t, z_s > z_t$	20	$x_s > x_t, y_s \approx y_t, z_s < z_t$
7	$x_s \approx x_t, y_s > y_t, z_s \approx z_t$	21	$x_s > x_t, y_s \approx y_t, z_s > z_t$
8	$x_s \approx x_t, y_s > y_t, z_s < z_t$	22	$x_s > x_t, y_s < y_t, z_s \approx z_t$
9	$x_s \approx x_t, y_s > y_t, z_s > z_t$	23	$x_s > x_t, y_s < y_t, z_s < z_t$
10	$x_s < x_t, y_s \approx y_t, z_s \approx z_t$	24	$x_s > x_t, y_s < y_t, z_s > z_t$
11	$x_s < x_t, y_s \approx y_t, z_s < z_t$	25	$x_s > x_t, y_s > y_t, z_s \approx z_t$
12	$x_s < x_t, y_s \approx y_t, z_s > z_t$	26	$x_s > x_t, y_s > y_t, z_s < z_t$
13	$x_s < x_t, y_s < y_t, z_s \approx z_t$	27	$x_s > x_t, y_s > y_t, z_s > z_t$
14	$x_s < x_t, y_s < y_t, z_s < z_t$		

among media streams and semantic objects. In this section, the use of regular expression to represent the spatial relations of semantic objects is described. The following definition shows the notation for the relative positions in regular expressions.

Definition 3: Each input symbol of a regular expression contains one or more semantic objects which are enclosed by parentheses and appear in the same image or video frame. Each semantic object has a unique name which consists of some letters. The relative positions of the semantic objects relative to the target semantic object are represented by subscripted numbers. A superscripted string of digits is used to represent different subcomponents of *relation objects* if partial or complete overlapping of MBR occurs. The "&" symbol between two semantic objects is used to denote that the two semantic objects appear in the same image or video frame.

This representation is similar to the temporal regular expression. One semantic object is chosen to be the target semantic object in each image or video frame. In order to distinguish the relative positions, the three dimensional spatial relations are developed (as shown in Table 1). In this table, twenty-seven numbers are used to distinguish the relative positions of each semantic object relative to the target semantic object. Value 1 is reserved for the target semantic object with (x_t, y_t, z_t) coordinates. Let (x_s, y_s, z_s) represent the coordinates of any semantic object. The relative position of a semantic object with respect to the target semantic object is determined by the X-, Y-, and Z-coordinate relations. The "$\approx$" symbol means the difference between two coordinates is within a threshold value. For example, relative position number 10 means a semantic object's X-coordinate (x_s) is less than the X-coordinate (x_t) of the target semantic object, while Y- and Z-coordinates are approximately the same. In other words, the semantic object is on the left of the target semantic object. More or fewer numbers may be used to divide an image or a video frame into subregions to allow more fuzzy or more precise queries as necessary. The centroid point of each semantic object is used for space reasoning so that any semantic

object is mapped to a point object. Therefore, the relative position between the target semantic object and a semantic object can be derived based on these centroid points. A regular expression then can be formed after relative positions are obtained.

Each new input symbol is created in a regular expression when any relative position of a semantic object changes or the number of semantic objects changes. Based on our design, the temporal and relative positions of semantic objects can be obtained and the moving history of semantic objects in the video frames can be kept. There is no exiting model that models the spatial, temporal, and moving in one framework as ours.

Figures 1 through 3 are three video frames with frame numbers 1, 52, and 70 which contain four semantic objects, *salesman, box, file holder*, and *telephone*. They are represented by symbols S, B, F, and T, respectively. Each semantic object is surrounded by a minimal bounding rectangle. Let *salesman* be the target semantic object. In Figure 1, the relative position numbers of the other three semantic objects with respect to the target semantic object are at 10, 15, and 24, respectively. The semantic object *box* moves from *left* to *front* of the target semantic object *salesman* in Figure 2, and moves back to *left* in Figure 3. The following regular expression can be used to represent Figures 1 through 3 as follows:

$$\text{Regular expression: } \underbrace{(S_1\&B_{10}\&F_{15}\&T_{24})}_{X_1}\underbrace{(S_1\&B_3\&F_{15}\&T_{24})}_{X_2}\underbrace{(S_1\&B_{10}\&F_{15}\&T_{24})}_{X_3} \qquad [3.1]$$

S_1 in symbol X_1 means *salesman* is the target semantic object. B_{10} represents that the semantic object *box* is on the *left* of *salesman*, F_{15} means semantic object *file holder* is below and to the left of *salesman*, and so on. B_3 in symbol X_2 means the relative position of *box* changes from *left* to *front*. Semantic objects *file holder* and *telephone* do not change their positions so that these two semantic objects have the same relative position numbers in X_1, X_2, X_3. As we can see from this example, the regular expression can represent not only the relative positions of the semantic objects but also the moving of the semantic

Figure 1: Vedio frame 1. There are four semantic objects: *salesman, box, file holder*, and *telephone*. *salesman* is the target semantic object. The relative position numbers (as defined in Table 1) of the other three semantic objects are in the 10, 15, and 24, respectively.

Figure 2: Vedio frame 52. Semantic object *box* moves from the left to the front of *salesman*.

Figure 3: Vedio frame 70. Semantic object *box* moves from the front to the left of *salesman*.

objects. For example, the above regular expression shows the semantic object *box* moves from *left* to *front* relative to the target semantic object *salesman*.

Figure 4 is another example showing how to use the regular expression to represent the relative positions. In Figure 4(a), there are five semantic objects P, Q, R, S, and T surrounded by five MBRs. Let T be the target semantic object. The relative positions of the other semantic objects are at 16, 25, 13, and 22, respectively. The numbers are chosen based on the criteria defined in Table 1. In Figure 4(b), the semantic object Q disappears and the semantic object P moves to the above and to the right of T. The regular expression for Figure 4(a) and (b) is as follows:

Regular expression: $\underbrace{(P_{16}\&Q_{25}\&R_{13}\&S_{22}\&T_1)}_{X_1}\underbrace{(P_{25}\&R_{13}\&S_{22}\&T_1)}_{X_2}$

In input symbol X_1, T_1 indicates that T is the target semantic object, P_{16} means P is above and to the left of T, Q_{25} means Q is above and to the right of T, and so on. In input symbol X_2, P_{25} denotes that the relative position of P changes to the above and to the right of T. Q does not appear in input symbol X_2 indicating that Q disappears as shown in Figure 4(b).

4. AN EXAMPLE

Figure 5 shows an example to use a single ATN to model two multimedia presentations which include user interactions, loops, and other presentations. In Figure 5(a), two presentations, P_1 and P_2, start at different starting states. In presentation 1 (P_1), after X_1 and X_2 are displayed, an existing presentation (say P_3) is displayed. Since P_3 is an existing presentation, it becomes a subnetwork of P_1 (as shown in Figure 5(b)). Since P_3 is embedded in P_1, X_4 and X_5 are displayed after P_3 is finished. When arc 9 is reached, the input symbol X_8 which contains two selection buttons(B_1 and B_2) is displayed to let users make choices. If B_1 is selected, X_9 and X_{10} are displayed and the presentation stops. If B_2 is selected, X_{11} and X_{12} are displayed. After X_{12} is displayed, a loop goes back to let user make the choice again.

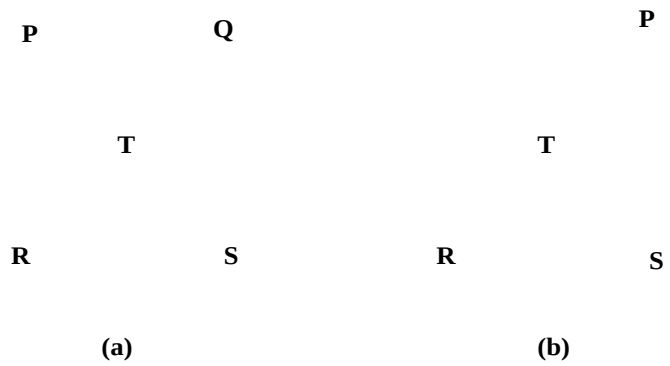


Figure 4: In (a), there are five semantic objects P, Q, R, S and T. In (b), the semantic object Q disappears and the semantic object P moves to the above and to the right of T.

Presentation 2 (P_2) is similar to P_1 except that X_3 is displayed before P_3, and X_6 and X_7 are displayed after P_3. That is, presentations 1 and 2 share arc 4 and arcs 9 through 14.

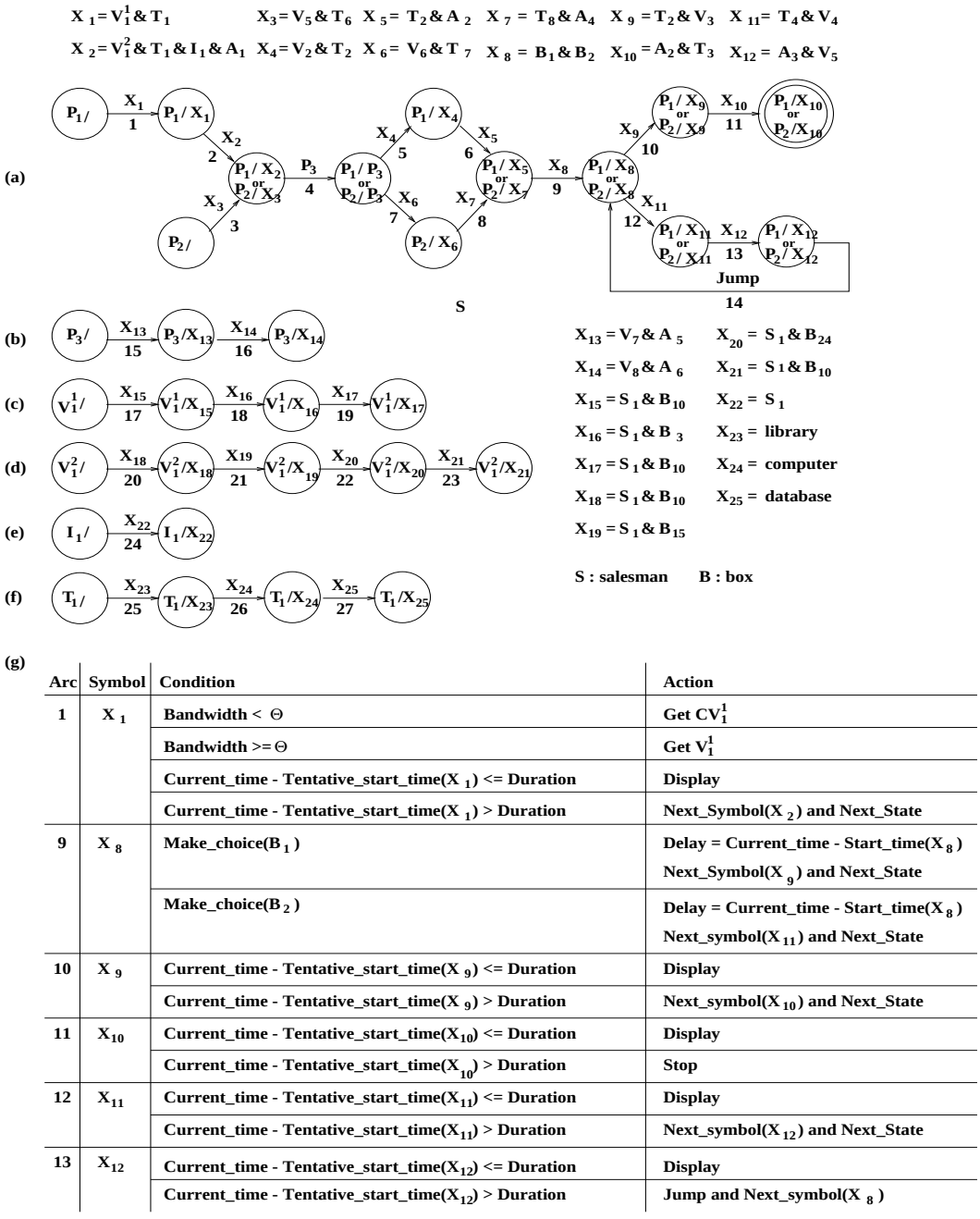
In presentation 1, the regular expression is:

$$\underbrace{(V_1^1 \& T_1)}_{X_1}\underbrace{(V_1^2 \& T_1 \& I_1 \& A_1)}_{X_2}\underbrace{(P_3)}_{P_3}\underbrace{(V_2 \& T_2)}_{X_4}\underbrace{(T_2 \& A_2)}_{X_5}(\underbrace{(B_1 \& B_2)}_{X_8}(\underbrace{(T_2 \& V_3)}_{X_9}\underbrace{(A_2 \& T_3)}_{X_{10}} \mid \underbrace{(T_4 \& V_4)}_{X_{11}}\underbrace{(A_3 \& V_5)}_{X_{12}}))^+ \quad [4.1]$$

In presentation 2, the regular expression is:

$$\underbrace{(V_5 \& T_6)}_{X_3}\underbrace{(P_3)}_{P_3}\underbrace{(V_6 \& T_7)}_{X_6}\underbrace{(T_8 \& A_4)}_{X_7}(\underbrace{(B_1 \& B_2)}_{X_8}(\underbrace{(T_2 \& V_3)}_{X_9}\underbrace{(A_2 \& T_3)}_{X_{10}} \mid \underbrace{(T_4 \& V_4)}_{X_{11}}\underbrace{(A_3 \& V_5)}_{X_{12}}))^+ \quad [4.2]$$

As mentioned earlier, a regular expression is used to represent the presentation sequence. In presentation 1, the input symbol X_1 contains V_1^1 (video stream 1) and T_1 (text 1) which start at the same time and play concurrently. Later, I_1 (Image 1) and A_1 (Audio 1) begin and overlap with V_1^2 and T_1. Therefore, the input symbol X_2 contains the media streams V_1^2, T_1, I_1, and A_1. Each media stream has its own regular expression and is a subnetwork of P_1 (as shown in Figures 5(c) to 5(f)). V_1^1 and V_1^2 are the first and the second parts of V_1, respectively. The delay time for I_1 and A_1 to display needs not to be specified in regular expression explicitly since the regular expression is read from left to right so that the time needed to process X_1 is the same as the delay time for I_1 and A_1. The presentation continues until the final state is reached. In Figure 5(e), there is only one input symbol for the subnetwork modeling I_1.



(g)

Arc	Symbol	Condition	Action
1	X_1	Bandwidth < Θ	Get CV_1^1
		Bandwidth >= Θ	Get V_1^1
		Current_time - Tentative_start_time(X_1) <= Duration	Display
		Current_time - Tentative_start_time(X_1) > Duration	Next_Symbol(X_2) and Next_State
9	X_8	Make_choice(B_1)	Delay = Current_time - Start_time(X_8) Next_Symbol(X_9) and Next_State
		Make_choice(B_2)	Delay = Current_time - Start_time(X_8) Next_symbol(X_{11}) and Next_State
10	X_9	Current_time - Tentative_start_time(X_9) <= Duration	Display
		Current_time - Tentative_start_time(X_9) > Duration	Next_symbol(X_{10}) and Next_State
11	X_{10}	Current_time - Tentative_start_time(X_{10}) <= Duration	Display
		Current_time - Tentative_start_time(X_{10}) > Duration	Stop
12	X_{11}	Current_time - Tentative_start_time(X_{11}) <= Duration	Display
		Current_time - Tentative_start_time(X_{11}) > Duration	Next_symbol(X_{12}) and Next_State
13	X_{12}	Current_time - Tentative_start_time(X_{12}) <= Duration	Display
		Current_time - Tentative_start_time(X_{12}) > Duration	Jump and Next_symbol(X_8)

Figure 5: Augumented Transition Network: (a) is the ATN network for two multimedia presentations which start at the states $P_1/$ and $P_2/$, respectively. (b)-(f) are part of the subnetworks of (a). (b) is an embedded presentation. (c) and (d) model the semantic objects in video media stream V_1, (e) models the semantic objects in image media stream I_1, and (f) models the keywords in text media stream T_1. In (g), CV_1^1 stands for the compressed version of the video media stream V_1^1. The "Get" procedure is to access an individual media stream. "Display" procedure is to display the media streams. "Next_Symbol(X_i)" reads the input symbol X_i. "Next_State" is a procedure to advance to the next state. "Make_choice(B_i)" is a procedure to let users make the choice which is B_i. User thinking time is kept by **delay** variable. θ is a parameter.

The input symbol is X_{22} which contains the semantic object *salesman*. Figure 5(f) is the subnetwork for T_1. The input symbols for T_1 consist of three keywords with appearing sequence *library*, *computer*, and *database*.

The details of how to use ATNs and regular expressions to model embedded presentations, user interactions, loops, and user specified presentations are discussed in the next four subsections.

4.1. Embedded Presentations

The production of high-quality multimedia data such as images and video clips takes a lot of time and is expensive. It is a good way to store these data into large shared databases. Any multimedia application that uses these databases can be greatly facilitated by an underlying model that supports the development process [19]. The main focus of this underlying model is that the data and structure can be shared, portable, and reused by the designer to create a new multimedia application. Under this design, a multimedia application (presentation) can be separated into several modules based on its conceptual structure. A finite state machine (FSM) does not have a pushdown mechanism which suspends a current process and goes to another state. Hence, it is not capable of modeling embedded presentations. While in ATNs, the recursive control mechanisms can be used to model embedded presentations.

As shown in Figure 5(a), the arc label for arc 4 is P_3 which points to an existing presentation modeled by ATN. After P_3 is displayed, the control returns to state labeled P_1/P_3 (or P_2/P_3) and display the next input symbol. A state with arc label P_1/P_3 represents reaching a state of presentation 1 (P_1) in which input symbol P_3 is read. As can be seen from Figure 5(b), P_3 has its own regular expression and becomes a subnetwork of P_1. Nested subnetworks are allowed, i.e., a subnetwork can have its own subnetworks. This feature allows embedded presentations which can be shared by many presentations and therefore greatly reduces the design time and complexity.

4.2. User Interactions

In user interactions, user thinking time delays need to be kept so that later presentations can be shifted. The cross-serial dependencies cannot be handled using FSM. However, the conditions and actions in ATNs have the ability to model user interactions.

In Figure 5(a), after the state P_1/X_5 (or P_2/X_7) the input symbol X_8 with two selection buttons B_1 and B_2 is displayed to users. Before users make a choice, a thinking time should be kept. A **delay** variable is used to represent the delay of presentation. Figure 5(g) shows the detailed condition column and action column for user interactions. As illustrated in Figure 5(a), two choices B_1 and B_2 are provided to let users make their selections. Either Make_choice(B_1) or Make_choice(B_2) will be matched in the condition column in Figure 5(g). The corresponding action to calculate the delay time is activated in the action column. The calculated delay time is used to shift the start time of any media stream displayed after the selection buttons. If users choose B_1 then input symbol X_9 is read and its tentative start time is added by the delay time to form the new start time for displaying. If the difference between the current time and the new start time is less than the duration, then it continues to display until the difference reaches the duration. The process is repeated until the final state P_1/X_{10} (or P_2/X_{10}) is reached.

The following equations calculate the total delay and the new start time for any media stream displayed after a user interaction occurs.

- kth user interaction delay time = δ_k = Current_time - Start_time(k); where Start_time(k) is the start time of the kth user interaction.
- New_start_time(X_i) after kth user interaction = Tentative_start_time(X_i) + δ_k; where the new start time for input symbol X_i is the sum of the original start time of X_i and the kth user interaction delay time δ_k.
- Total delay = $\mathcal{D} = \sum_{i=1}^{m} \delta_k$ where m is the number of user selections.

- Total Presentation time $\mathcal{P} = \mathcal{E}$ - $\mathcal{S}$ + $\mathcal{D}$ where $\mathcal{S}$ and $\mathcal{E}$ are the tentative start time and end time of the multimedia presentation.

The Start_time is defined to be the time that the selection buttons are shown to let the user make the choice. The Current_time is defined to be the time that the user makes the choice. The difference between them is the delay time for the corresponding user interaction. After the delay is obtained, then the start time for the later presentation (New_start_time) is the sum of the tentative start time plus this delay time.

4.3. Loops

When B_2 is chosen after the state P_1/X_5 (or P_2/X_7) in Figure 5(a), input symbols X_{11} and X_{12} are read. The corresponding media streams for X_{11} and X_{12} are displayed until the total display time exceed the durations. When the display time for X_{12} is greater than the duration, an action "Jump" lets the control go back to state P_1/X_8 (or P_2/X_8) and the input symbol X_8 is read to let users make choices again. If users choose B_2 then the loop is executed again. The presentation ends only after users choose B_1 and media streams in X_9 and X_{10} are displayed.

The above process models a loop scenario which is represented by a "+" symbol in the regular expressions [4.1] and [4.2]. The "Jump" action does not advance the input symbol but let the control goes to the pointing state. That means the "Jump" itself is not an input symbol in regular expressions. This feature is crucial for the designers who may want some part of the presentation to be seen over and over again. For example, in a computer-aided instruction (CAI) presentation, the teacher may want the students to view some part of the presentation until they become familiar with this presentation part.

4.4. User Specified Presentations

In a multimedia environment, users should be allowed to issue queries to get the information and display to them. How to combine multimedia presentations with multimedia database systems is a big challenge today. ATNs together with regular expressions have the ability notively to model the multimedia presentation but also to answer the multimedia database queries. In the following subsections, how to use ATNs and regular expressions to answer user queries relative to temporal, spatial, spatio-temporal, recursive, and unordered aspects are discussed. The most important thing is that users can specify the criteria relative to the contents of images or video frames.

The presentations in Figure 5 are used in the following discussions unless specified otherwise.

4.4.1. The Formalization of Searching Strategies

In ATNs, the complexity of a query depends crucially on the order in which the network is searched for a successful path. If the states are in linear order, then the traversing goes from left to right. However, when two or more arcs leaving out of a state, the traversing order needs to be specified. Our searching strategy for this situation is to go from the topmost path to the bottommost path. For example, arcs 10 and 11 will be tried prior to arcs 12 and 13 in Figure 5. When traversing an attempted arc, the remaining untried arcs leaving the state are temporarily held in a list. After this attempted arc is traversed, one alternative is removed from the front of this list and tried. The process continue until no alternatives left in this list. As a result of this *depth-first* search, the information which satisfies the criteria in the queries are obtained. The corresponding actions can build up a presentation tree so that the presentation follows the order specified in this tree. There are three different nodes in the presentation tree which are root, input symbol, and media stream nodes. The input symbol nodes show the input symbols to be displayed. Each input symbol node connects to several media stream nodes that composed of this input symbol. There is only one root node. In the later presentation, *depth-first* search is used to traverse the

presentation tree so that the corresponding media streams can be displayed concurrently and sequentially. In this manner, users can choose part of a presentation to watch and the order of the presentation can be different from the original specification.

4.4.2. Temporal Database Queries

- **Query 1: Find and display the video clips beginning with a person who is alone and ending with the same person with a cat.**

In this query, we are interested in the temporal relation of two semantic objects (a *person* (R) and a *cat* (C)). This query can be translated into two regular expressions (R) and ($R\&C$). Relative position number is not specified for each semantic object to indicate that the spatial location is not our concern so that it will match the semantic object at any location. For example, R matches with R_1, R_2, and so on. (R) must appear earlier than ($R\&C$) but not necessarily immediately after (R). After translating this query into a regular expression, this query then becomes a substring matching problem. Each subnetwork which models video media streams is checked to see whether this regular expression can be identified. These two input symbols may be found in two separate subnetworks. The conditions and actions are used to check which criteria are satisfied and to connect the required presentation parts.

4.4.3. Spatial Database Queries

In the following two subsections, database queries use the situation from Figures 1 through 3. Assume there are several video media streams and V_s is one of them. The regular expression for the subnetwork which models V_s is as follows:

$$\text{Regular expression: } \underbrace{(S_1\&B_{10}\&F_{15}\&T_{24})}_{X_1}\underbrace{(S_1\&B_3\&F_{15}\&T_{24})}_{X_2}\underbrace{(S_1\&B_{10}\&F_{15}\&T_{24})}_{X_3} \qquad [4.3]$$

In this regular expression, it contains semantic objects *salesman* (S), *box* (B), *file folder* (F) and *telephone* (T).

- **Query 2: Display the multimedia presentation beginning with a salesman with a box on his right.**

In this query, only the spatial locations of two semantic objects *salesman* and *box* are checked. The subnetworks of an ATN are traversed and the corresponding regular expressions are analyzed. Suppose the regular expression in [4.3] is to model the subnetwork for V_s. The input symbol X_1 in V_s contains semantic objects *salesman* (S), and *box* (B). Let the *salesman* be the target semantic object. The relative position of the *box* is to the left of *salesman* in the way that users see the video clip. Based on the information, we know that V_s is the starting video clip of the query. When the control is passed back from the subnetwork, then the rest of the multimedia presentation begins to display.

4.4.4. Spatio-Temporal Database Queries

- **Query 3: Find the video clips beginning with a salesman holding a box on his right, moving the box from the right to his front, and ending with moving the box back to his right.**

This query involves both the temporal and spatial aspects of two semantic objects: *salesman* and *box*. Again, each of the subnetwork needs to be checked one by one. The same as in the previous query, the relative positions to be matched are based on the view that users see the video clip. Using the same regular expression in [4.3], the first condition asks to match the relative position of *box* is to the left of *salesman*. When the subnetwork of V_s is traversed, S_1 and B_{10} tell us that the input symbol X_1 satisfies the first condition which *box* is to the left of *salesman*. Next, the relative position of *box* is moved from the left to the front of *salesman*. This is satisfied by the input symbol X_2 since B_3 indicates that *box* is

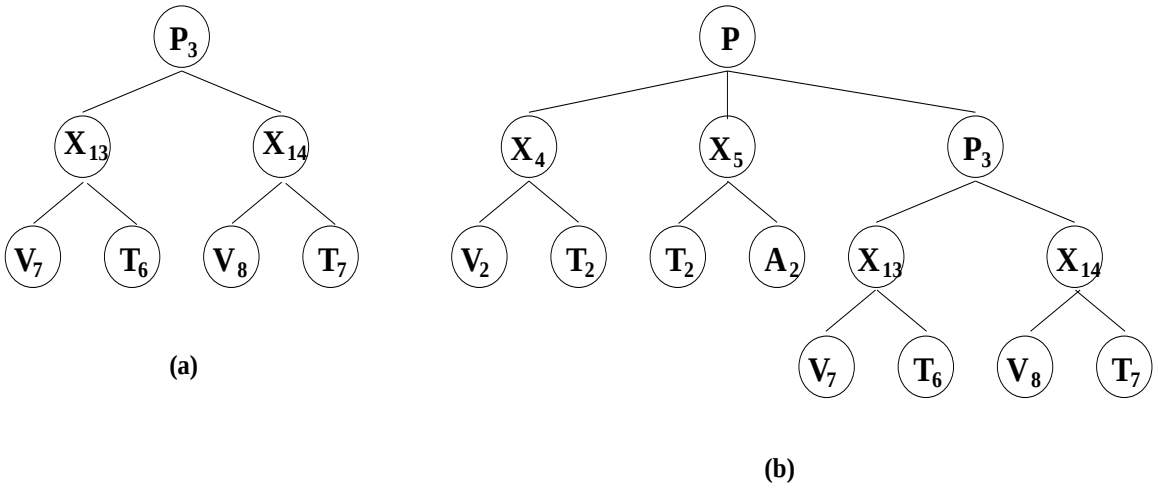


Figure 6: In (a), a struture of P_3 is constructed when P_3 satisfies the query condition. In (b), X_4 and X_5 need to be displayed before P_3. The actions can generate a tree which puts X_4 and X_5 before P_3 so that X_4 and X_5 can be displayed before P_3.

to the *front* of *salesman*. Finally, it needs to match the relative position of *box* to be back to the left of *salesman*. This condition is exactly the same as the first condition and should be satisfied by the input symbol X_3. In this query, the semantic object *salesman* is the target semantic object and his position remains the same without any change.

4.4.5. Unordered Queries

In a multimedia database query, user may want to see the presentation with different orders as prespecified. Conditions are used to match the arc labels with the regular expressions of a query, and actions are used to generate a presentation tree with the structure specified in the query. After this tree is constructed, the presentation then can be displayed in that structure to the users. This feature provides users a great flexibility since they can view the presentation with different orders. Figure 6 demonstrates an example of unordered presentation in such a scenario that user wants to display X_4 and X_5 before P_3. At first, the embedded presentation P_3 satisfies the condition and the action to generate a tree

structure for P_3 is invoked. Since the query specifies that X_4 and X_5 should be displayed prior to P_3, a corresponding action which makes X_4 and X_5 the left child nodes and P_3 the right child node is invoked to form a tree for the query. Based on the depth-first search property of this tree, a new presentation sequence which allows X_4 and X_5 to be displayed before P_3 is formed.

4.5. Synchronization and Quality-Of-Service (QoS) Maintenance

Figure 5(g) shows an example of how to use conditions and actions to maintain synchronization and QoS for the input symbol X_1. In presentation 1, when the current input symbol X_1 (V_1^1&T_1) is read, the bandwidth condition is first checked to see whether the bandwidth is enough to transmit these two media streams. If it is not enough then the compressed version of V_1^1 (CV_1^1) will be transmitted instead V_1^1. Then the condition whether the pre-specified duration to display V_1^1 and T_1 is reached is checked. If it is not, the display continues. The start time is defined to be the time when the displaying of the media streams starts. The difference between the current time and the start time is the total display time so far. The last condition is met when the total display time reaches the pre-specified duration. In this case, a next input symbol X_2 is read. The same conditions are checked for X_2, too. The process continues until the final state is reached.

5. RELATED WORK

Little and Ghafoor [14] [15] proposed an Object Composition Petri Net (OCPN) model based on the logic of temporal intervals and Timed Petri Nets. OCPN was proposed to store, retrieve, and communicate between multimedia objects. This model is a modification of earlier Petri net models and consists of a set of transitions (bars), a set of places (circles), and a set of directed arcs. OCPN is a network model and a good data structure for controlling the synchronization of the multimedia presentation. A network model can easily show the time flow of a presentation. Therefore, OCPN can serve as a visualization structure

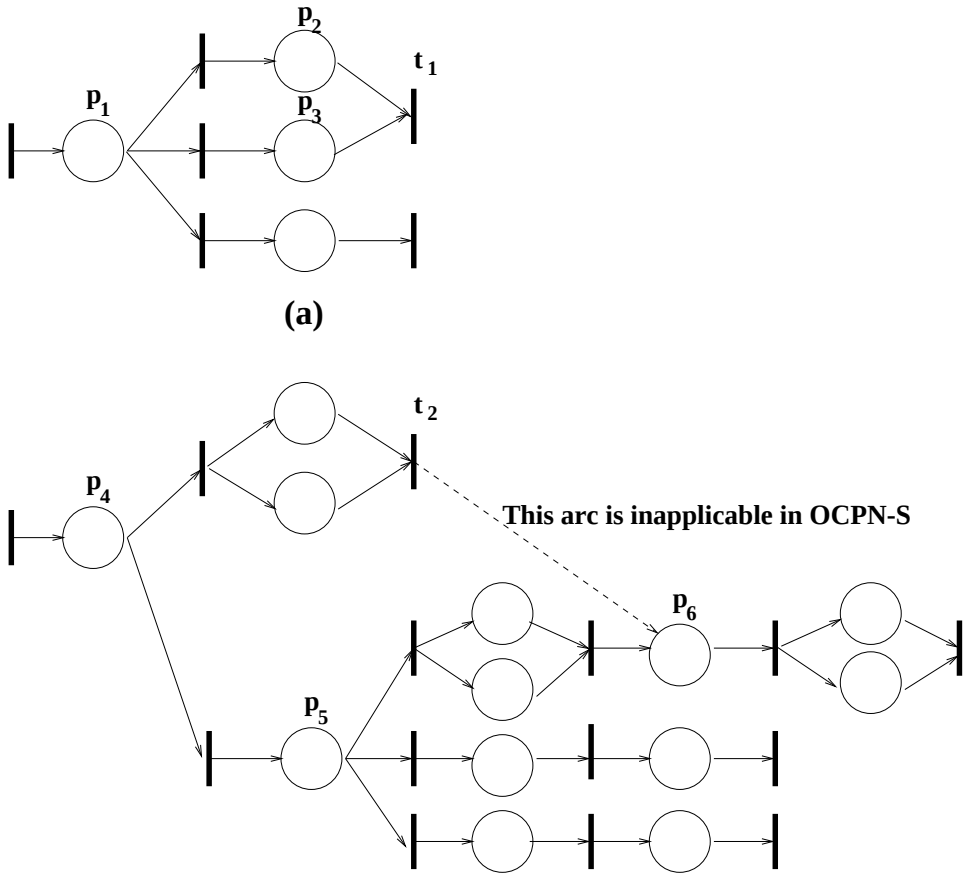


Figure 7: In (a), p_1 is the selection place. The transition t_1 is impossible to be fired since only one selection path can be chosen in OCPN-S and therefore, either p_2 or p_3 can get the token. One of t_1's input places does not have an *unlocked* token. In (b), there are two selection places p_4 and p_5 whereas p_5 belongs to one selection path of p_4. As shown in (a), two paths belonging to different selections cannot merge together, so the arc from transition t_2 to place p_6 is inapplicable in OCPN-S.

for users to understand the presentation sequence of media streams. Many later abstract semantic models are based on petri-net or OCPN [6] [13] [1] [20].

OCPN is a form of *marked graph* so that each place in an OCPN has exactly one incoming arc and one outgoing arc. *Marked graph* can only model those systems whose control flow has no branch. Hence, it can model the parallel activities but not alternative activities [18] such as user interactions. In order to model user interactions, Day [8] proposed an OCPN-S with user selection functions supplemented in OCPN. In OCPN-S, a selection place which may have more than one outgoing arcs to model the alternative activities is defined. Two paths belonging to different selections cannot merge together to avoid the situation as shown in Figure 7(a). In Figure 7(a), if two selection paths p_2 and p_3 merge at transition t_1 then transition t_1 cannot be fired. The reason is that either place p_2 or p_3 can have token since those places belonging to the unselected paths are excluded from the presentation. By having this restriction, the situation in Figure 7(b) is not applicable in OCPN-S. In Figure 7(b), there are two selection places, p_4 and p_5, where p_5 belongs to one selection path of p_4. The arc originating from transition t_2 to place p_6 is inapplicable to OCPN-S since two paths belonging to different selections cannot merge together and each place cannot have more than one incoming arc. In applications such as computer-aided instruction (CAI), a situation as in Figure 7(b) can happen since CAI applications may involve a lot of two-way communications and different selections will merge together occasionally. On the other hand, in ATNs, based on the selections of users, the corresponding input symbol is read, and the conditions and actions are used to pass the control to the desired state to let the presentation continue. Therefore, ATN allows nested selections and merges for the scenario in Figure 7(b). In addition, the shared presentation in Figure 7(b) can be modeled by using embedded presentation in ATNs as shown in section 4.1.

Oomoto and Tanaka [16] developed a video-object database system named OVID. A video-object data model provides interval-inclusion based inheritance and composite video-object operations. A video object is a video frame sequence (a meaningful scene) and it is an independent object itself. Each video

object has its own attributes and attribute values to describe the content. Since this model only uses frame sequences to represent the interval, the start time and end time of each interval are not included. Their model is designed to help browsing and database queries related to video objects. The subnetworks for video media streams in ATNs can provide the similar functionality as OVID does. These subnetworks can be used to answer the queries involving the temporal and spatial relations of semantic objects. Moreover, ATNs can model text and image media streams using subnetworks. Hence, database queries can access text and image information too.

Hirzalla et al. [11] developed a graphical temporal model for interactive multimedia documents to expand the traditional timeline models such as Blakowski et al. [3] to include temporal inequalities between events. The main contribution of this model is to include user actions in their model. A new type of media object called *choice* is included in the vertical axis of the timeline. This new object is associated with a data structure that contains user actions, regions, and destination scenario pointers. In order to have the same functionality as this model, the *delay* variables in conditions and actions are used in ATNs to keep track the user delays in the user selections so that the later presentations can be shifted.

6. CONCLUSIONS

In this paper, we describe an ATN based model together with regular expressions for multimedia presentations and multimedia database applications. ATNs provide two major capabilities: *presentation* and *querying*. ATNs are left to right models which are used to model a presentation sequence from the beginning to the end. Each arc label is a string which consists of those media streams to be displayed. Each media stream consists of a subnetwork which allows querying capabilities in ATNs. Querying capabilities allow users to retrieve information related to media streams or semantic objects in a specific presentation directly. Subnetworks are used to model the temporal and spatial information of semantic

objects for image and video frames. Keywords sequence in text media streams can also be included in the subnetworks. Since a lot of research has already shown how to use ATNs to model a sentence, we do not discuss how to use ATNs to model audio media streams in this paper. Embedded presentations allow the reusing of existing presentation sequences. It emphasizes the modularity and reuses existing media streams and presentation structures. Under this design, the storage intensive multimedia data can be stored into large shared databases. This feature greatly reduces the design complexity and makes the design easier. This can be modeled by using subnetworks too, via putting a presentation name as an arc label such as "P2" in Figure 5.1. Further investigation on how to solve the heterogeneity to allow the searching via the embedded presentations can be conducted. User interactions are included in ATNs since ATNs provide branching for the alternative choices. User interaction features allow two-way communication between users and multimedia information systems. The user thinking times in the users decision processes can be handled by conditions and actions in ATNs. By using a "variable" to keep track of the time duration for the decisions, the latter presentations can be shifted by this time duration since this "variable" can be passed to the later conditions to decide the adjusted start time. Another major difference among ATNs and other semantic models is that users can get a presentation order which is different from the original presentation sequence. This feature gives users a great flexibility to compose a presentation from an existing presentation. Also, ATNs allow loops in a presentation. Loops can be used to let some part of a presentation be watched more than once. Unlike the existing semantic models which model either presentations or queries, our ATN model provides these two capabilities in one framework.

Acknowledgements

This work has been partially supported by National Science Foundation under contract IRI 9619812 and the office of Naval Research under contract N00014-91-J-4126.

References

[1] Y.Y. Al-Salqan and C.K. Chang, "Temporal Relations and Synchronization Agents," *IEEE Multimedia*, pp. 30-39, Summer 1996.

[2] R. Bayer and E. McCreight, "Organization and Maintenance of Large Ordered Indices," *Proc. 1970 ACM-SIGFIDENT Workshop on Data Description and Access*, Houston, Texas, pp. 107-141, Nov. 1970.

[3] G. Blakowski, J. Huebel, and U. Langrehr, "Tools for Specifying and Executing Synchronized Multimedia Presentations," *Proc. 2nd Int'l Workshop on Network and Operating System Support for Digital Audio and Video*, pp. 271-279, 1991.

[4] M. Buchanan and P. Zellweger, "Automatically Generating Consistent Schedules for Multimedia Documents," *ACM Multimedia Systems Journal*, 1(2), Springer-Verlag, pp. 55-67, 1993.

[5] S.K. Chang, C.W. Yan, D.C. Dimitroff, and T. Arndt, "An Intelligent Image database System," *IEEE Trans. on Software Engineering*, vol 14, no. 5, pp. 681-688, May 1988.

[6] H.J. Chang, T.Y. Hou, S.K. Chang, "The Management and Application of Teleaction Objects," *ACM Multimedia Systems Journal*, vol. 3, pp. 228-237, November 1995.

[7] D. Comer, "The Ubiquitous B-tree," *Computing Surveys*, 11:2, pp. 121-138, June 1979.

[8] Young Francis Day, "Semantic Modeling and Management of Multimedia Data," Ph.D Thesis, Purdue University, Auguest 1996.

[9] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, P. Yanker, "Query by Image and Video Content: The QBIC System," *IEEE Computer*, vol. 28, no. 9, pp. 23-31, September 1995.

[10] A. Guttman, "R-tree: A Dynamic Index Structure for Spatial Search," *in Proc. ACM SIGMOD*, pp. 47-57, June 1984.

[11] N. Hirzalla, B. Falchuk, and A. Karmouch, "A Temporal Model for Interactive Multimedia Scenarios," *IEEE Multimedia*, pp. 24-31, Fall 1995.

[12] S.C. Kleene, "Representation of Events in Nerve Nets and Finite Automata, Automata Studies," *Princeton University Press*, Princeton, N.J., pp. 3-41, 1956.

[13] C.C. Lin, J.X., S.K. Chang, "Transformation and Exchange of Multimedia Objects in Distributed Multimedia Systems," *ACM Multimedia Systems Journal*, vol. 4, pp. 12-29, February 1996.

[14] T.D.C. Little and A. Ghafoor, "Synchronization and Storage Models for Multimedia Objects," *IEEE J. Selected Areas in Commun.*, vol. 9, pp. 413-427, Apr. 1990.

[15] T.D.C. Little and A. Ghafoor, "Interval-Based Conceptual Models for Time-Dependent Multimedia Data," *IEEE Trans. On Knowledge and Data Engineering*, vol. 5, no 4, pp. 551-563, Aug. 1993.

[16] E. Oomoto, and K. Tanaka, "OVID: Design and Implementation of a Video Object Database System," *IEEE Trans. on Knowledge and Data Engineering*, vol. 5, no. 4, pp. 629-643, August 1993.

[17] M.T. Özsu, D. Duane, G. El-Medani, C. Vittal, "An object-oriented multimedia database system for a news-on-demand application," *ACM Multimedia Systems Journal*, vol. 3, pp. 182-203, November 1995.

[18] J.L. Peterson, "Petri nets," *ACM Comput. Surveys*, vol. 9, pp. 223-252, Sept. 1977.

[19] G.A. Schloss and M.J. Wynblatt, "Providing definition and temporal structure for multimedia data," *ACM Multimedia Systems Journal*, vol. 3, pp. 264-277, November 1995.

[20] Heiko Thimm and Wolfgang Klas, "δ-Sets for Optimized Relative Adaptive Playout Management in Distributed Multimedia Database Systems," *IEEE 12th International Conference on Data Engineering*, New Orleans, Louisiana, pp. 584-592, 1996.

[21] W. Woods, "Transition Network Grammars for Natural Language Analysis," *Comm. ACM*, **13**, pp. 591-602, October 1970.