A Spatio-Temporal Semantic Model for Multimedia Database Systems and Multimedia Information Systems

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

Abstract—As more information sources become available in multimedia systems, the development of abstract semantic models for video, audio, text, and image data becomes very important. An abstract semantic model has two requirements: It should be rich enough to provide a friendly interface of multimedia presentation synchronization schedules to the users and it should be a good programming data structure for implementation in order to control multimedia playback. An abstract semantic model based on an augmented transition network (ATN) is presented. The inputs for ATNs are modeled by multimedia input strings. Multimedia input strings 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 multimedia input string. In this design, a presentation is driven by a multimedia input string. Each subnetwork has its own multimedia input string. Database queries relative to text, image, and video can be answered via substring matching at subnetworks. Multimedia browsing allows users the flexibility to select any part of the presentation they prefer to see. This means that the ATN and its subnetworks can be included in multimedia database systems which are controlled by a database management system (DBMS). User interactions and loops are also provided in an ATN. Therefore, ATNs provide three major capabilities: multimedia presentations, temporal/spatial multimedia database searching, and multimedia browsing.

Index Terms—Augmented Transition Network (ATN), multimedia database systems, multimedia input string, multimedia presentations, semantic object.

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 a computer. Normally, multimedia systems require the management and delivery of extremely large bodies of data at very high rates and may require 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.

- *S.-C. Chen is with the School of Computer Science, Florida International University, Miami, FL 33199. E-mail: chens@cs.fiu.edu.*
- *R.L. Kashyap is with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907-1285. E-mail: kashyap@ecn.purdue.edu.*

Manuscript received 12 Nov. 1997; revised 10 Feb. 1999; accepted 14 Dec. 1999; posted to Digital Library 6 Apr. 2001.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number 111120.

The augmented transition network (ATN), developed by Woods [24], has been used in natural language understanding systems and question answering systems for both text and speech. We use the ATN to model the semantic aspects of a multimedia presentation, browsing, and database searching [9], [10]. The designer can design a general purpose multimedia presentation using an ATN so that users can take their own needs to watch, browse, and search this presentation. Since ATNs can model user interactions and loops, the designer can design a presentation with selections so that users can utilize the choices to browse or watch the same part of the presentation more than once. 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 where user interactions such as user selections and loops are allowed, we cannot use sentences to be inputs for an ATN. In our design, multimedia input strings are used to be the inputs for ATNs. A multimedia input string consists of one or more input symbols. An input symbol represents the media streams which are displayed at the same time within a given duration. These input symbols are also the arc label in an ATN. By using multimedia input strings, one can model user interactions and loops. The details will be discussed in this paper.

Subnetworks in ATNs are used to model detailed information of media streams such as video, image, and text. The inputs of these subnetworks are multimedia input strings, too. We use multimedia input string to model the temporal, spatial, or spatio-temporal relations of semantic objects. A semantic object is the object that appears in a video frame or image such as a "car." Users can issue queries using a high-level language such as SQL. This query then translates to a multimedia input string to match the multimedia input strings in the subnetworks. Therefore, multimedia database query processes become substring matching processes.

This paper addresses three important problems:

1. How to model multimedia presentations,
2. How to model temporal and spatial relations of media streams and semantic objects which multimedia database searchings related to different media streams and semantic objects can answer, and
3. How to model multimedia browsing so that users can select information to watch.

The organization of this paper is as follows: Section 2 discusses how ATNs and multimedia input strings model multimedia presentations. Multimedia database searching and multimedia browsing based on ATNs and multimedia input strings are presented in Section 3 and Section 4. Section 5 shows a complicated example which uses ATNs and multimedia input strings to model user interactions and loops in multimedia information systems. Related work is in Section 6. The paper is concluded in Section 7.

2 Using ATNs and Multimedia Input Strings to Model Multimedia Presentations

2.1 The Augmented Transition Network

A multimedia presentation consists of a sequence of media streams displaying together or separately across 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 that includes these subnetworks. To design a multimedia presentation from scratch is a difficult 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 for creating a new presentation. This is similar to the *class* in an 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 of 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.

2.2 Multimedia Input Strings as Inputs for ATNs

Originally, an ATN was used for the analysis of natural language sentences. Its input is a sentence composed of words. This input format is not suitable for representing a multimedia presentation since several media streams need to be displayed at the same time, to be overlapped, to be seen repeatedly, etc. Multimedia input strings adopt the notations from regular expressions [16]. Regular expressions 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, multimedia input 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. A multimedia input string goes from left to right, which can represent the time sequence of a multimedia presentation but it cannot represent concurrent appearances and spatial locations of media streams and semantic objects. In order to let multimedia input strings have these two abilities, several modifications are needed. Two levels need to be represented by multimedia input strings. 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 for modeling a coarse-grained level are discussed as follows:

Two notations $\mathcal{L}$ and $\mathcal{D}$ are used to define multimedia input strings and are defined as follows:

1. $\mathcal{L} = \{A, I, T, V\}$ is the set whose members represent the media type, where A, I, T, V denote audio, image, text, and video, respectively.
2. $\mathcal{D} = \{0, 1, \ldots, 9\}$ is the set consisting of the set of the ten decimal digits.

Definition 1. *Each input symbol of a multimedia input string contains one or more media streams enclosed by parentheses and 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 following situations can be modeled by a multimedia input string:*

- ***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 being displayed concurrently.*

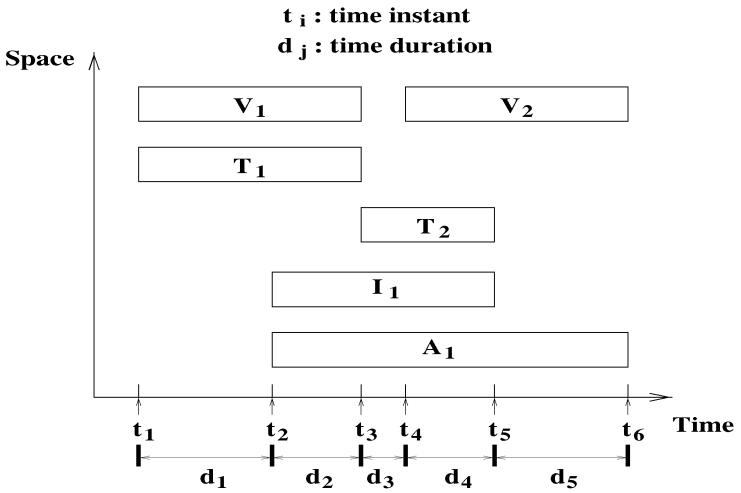


Fig. 1. Timeline for multimedia presentation. t_1 to t_6 are the time instances. d_1 is the time duration between t_1 and t_2, and so on.

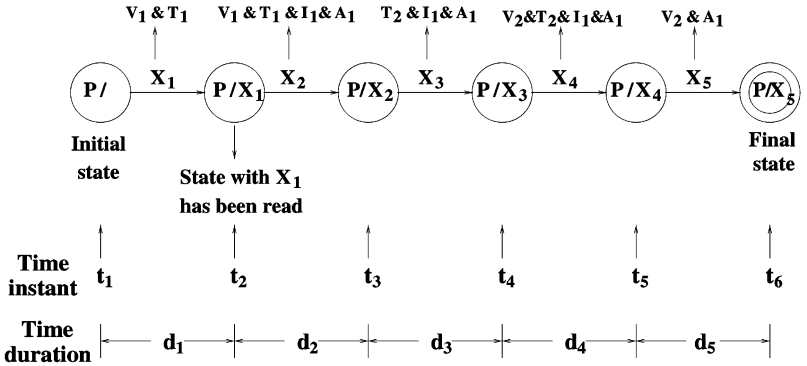


Fig. 2. *Augmenetd transition network* for a multimedia presentation.

- ***Looping.*** *$m^+ = \bigcup_{i=1}^{\infty} m^i$ is the multimedia input string of the 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 the "*" symbol to indicate the media streams which can be dropped in the online 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 multimedia input string. Input symbols are displayed from left to right sequentially across time. ab is the multimedia input string 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 multimedia input string can model user selections by separating input symbols with the "\| " symbol. So, $(a \mid b)$ is the multimedia input string 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.*
- ***Ending:*** *The symbol "$" denotes the end of the presentation.*

2.3 A Multimedia Presentation Example

Fig. 1 is a timeline representing a multimedia presentation. The presentation starts at time t_1 and ends at time t_6. At time t_1, media streams V_1 (Video 1) and T_1 (Text 1) start to play at the same time and continue to play. At time t_2, I_1 (Image 1), and A_1 (Audio 1) begin and overlap with V_1 and T_1. The duration d_1 is the time difference between t_1 and t_2. V_1 and T_1 end at time t_3 at which T_2 starts. The process continues till the end of the presentation. The timeline representation can model the temporal relations of media streams in a multimedia presentation [5]. Every presentation needs to strictly follow the prespecified sequence. However, it has difficulty representing user interactions and will be discussed later.

Fig. 2 is an ATN for Fig. 1. There are six states and five arcs which represent six time instants and five time durations, respectively. State names are in circles to indicate presentation status. State name $P/$ means the beginning of the *transition network* (presentation), and state name P/X_1

TABLE 1
Conditions and Actions Table.

Input Symbols	Condition	Action
$V_1\&T_1$	if bandwidth $< \theta$	Get compressed version V_1
	if bandwidth $\geq \theta$	Get V_1
	if current_time-start_time(X_1) $<$ duration	Display
	if current_time-start_time(X_1) $\geq$ duration	Get_Symbol and Next_State
$V_1\&T_1\&I_1\&A_1$	if bandwidth $< \tau$	Get compressed version I_1
	if bandwidth $\geq \tau$	Get I_1
	if bandwidth $< \rho$	Get compressed version I_1 and A_1
	if bandwidth $\geq \rho$	Get I_1 and A_1
	if current_time-start_time(X_2) $<$ duration	Display
	if current_time-start_time(X_2) $\geq$ duration	Get_Symbol and Next_State
$T_2\&I_1\&A_1$	if current_time-start_time(X_3) $<$ duration	Display
	if current_time-start_time(X_3) $\geq$ duration	Get_Symbol and Next_State
$V_2\&T_2\&I_1\&A_1$	if bandwidth $< \theta$	Get compressed version V_2
	if bandwidth $\geq \theta$	Get V_2
	if bandwidth $< \rho$	Get compressed version I_2 and A_2
	if bandwidth $\geq \rho$	Get I_2 and A_2
	if current_time-start_time(X_4) $<$ duration	Display
	if current_time-start_time(X_4) $\geq$ duration	Get_Symbol and Next_State
$V_2\&A_1$	if current_time-start_time(X_5) $<$ duration	Display
	if current_time-start_time(X_5) $\geq$ duration	Get_Symbol and Next_State

Get procedure is to access an individual media stream. Get_Symbol is a procedure which reads the next input symbol of a multimedia input string. Next_State is a procedure to advance to the next state in ATN. Display procedure displays the media streams. θ, τ, and ρ are the parameters.

denotes the state after X_1 has been read. The reason for using X_i is for convenience. In fact, X_1 can be replaced by V_1 and T_1. State name P/X_5 is the final state of the *transition network* indicating the end of the presentation. There are five occurrences of media stream combinations at each time duration. They are:

1. Duration d_1: V_1 and T_1.
2. Duration d_2: V_1, T_1, I_1, and A_1.
3. Duration d_3: T_2, I_1, and A_1.
4. Duration d_4: V_2, T_2, I_1, and A_1.
5. Duration d_5: V_2 and A_1.

Each arc label X_i in Fig. 2 is created to represent the media stream combination for each duration as above. For example, arc label X_1 represents media streams V_1 and T_1 displayed together at duration d_1. A new arc is created when new media streams I_1 and A_1 overlap with V_1 and T_1 to display. A multimedia input string is used as an input for this *transition network* and the symbol "&" between media streams indicates these two media streams are displayed concurrently. A multimedia input string consists of several input symbols and each of them represents the media streams to be displayed at a time interval.

Table 1 shows a simple example of how to use conditions and actions to control the synchronization and quality of service (QoS). Conditions provide more sensitive controls on the transitions in ATNs. 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 first column contains the input symbols and the second column shows the media streams. The third and the fourth columns show the conditions and the actions, respectively. When the current input symbol X_1 ($V_1\&T_1$) is read, the condition of the bandwidth is first checked to see whether the bandwidth is large enough to transmit media streams V_1 and T_1. If it is not, then the

compressed version of V_1 will be transmitted. Then, the second condition checks whether the prespecified duration to display V_1 and T_1 is reached. If it is not, the display continues. The start time is defined to be the time starting the display of V_1 and T_1, and the difference between the current time and the start time is the total display time so far. The third condition is met when the total display time is equal to the prespecified duration. In that case, a next input symbol X_2 ($V_1\&T_1\&I_1\&A_1$) is read and these four conditions will be checked again. The process continues until the final state (state 5) is reached.

3 Using ATNs and Multimedia Input Strings to Model Multimedia Database Searching

In a multimedia information environment, users may want to watch part of a presentation by specifying some features relative to image or video content prior to a multimedia presentation, and a designer may want to include other presentations in a presentation. In order to meet these two requirements, ATNs use a pushdown mechanism that permits one to suspend the current process and go to another state in the subnetwork to analyze a query that involves temporal, spatial, or spatio-temporal relationships. Subnetworks are separated from the main ATN. Before control is passed to the subnetwork, the state name at the head of the arc is pushed into the push-down store (stack). 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, the control goes back to the state removed from the top of the push-down store.

Two 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 in the number of semantic objects or any change in 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 appearing 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 control to them. After exiting the linked database system, the control is back to the ATN.

This section will focus on how to use multimedia input strings and ATNs to model the temporal and spatial relations of semantic objects. In our design, each image or video frame has a subnetwork which has its own multimedia input string. Subnetworks and their multimedia input strings are created by the designer in advance for a class of applications. Users can issue multimedia database queries using high-level database query languages such as SQL. Each high-level query then translates into a multimedia input string so that it can match with the multimedia input strings for subnetworks. Therefore, database queries become the substring matching processes. A multimedia input string 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 multimedia input string. User queries can be answered by analyzing the multimedia input string (for example, the movement, the relative spatial location, the appearing sequence, etc., of semantic objects). The spatial location of each semantic object needs to be represented by a symbolic form in order to use multimedia input strings to represent it.

3.1 Modeling the Spatial and Temporal Relations of Semantic Objects

Spatial data objects often cover multidimensional 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. An R-tree [14], which was proposed as a natural extension of B-trees [3], [11], combines the nice features of both B-trees and quadtrees. An 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 an R-tree so that each semantic object is covered by a rectangle. Three types of topological relations between the MBRs can be identified [7]:

1. nonoverlapping rectangles,
2. partly overlapping rectangles, and
3. completely overlapping rectangles.

For the second and the third alternatives, *orthogonal relations* proposed by Chang et al. [7] 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

TABLE 2
Three-Dimensional Relative Positions for Semantic Objects.

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	$x_s < x_t, y_s < y_t, z_s < z_t$		

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-corrdinates of the target and any semantic object, respectively. The "≈ " symbol means the difference between two coordinates is within a threshold value.*

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,*

$$\mathrm{O} = (o_1, o_2, \ldots, o_n).$$

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 3D 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 spatial reasoning.*

As mentioned in the previous section, multimedia input strings are used to represent the temporal relations among media streams and semantic objects. In this section, the use of a multimedia input string to represent the spatial relations of semantic objects is described. The following definition shows the notation for the relative positions in multimedia input strings.

Definition 3. *Each input symbol of a multimedia input string 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 numerical subscripts. 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 multimedia input string. 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 2). In this table, 27 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 "≈ " 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.

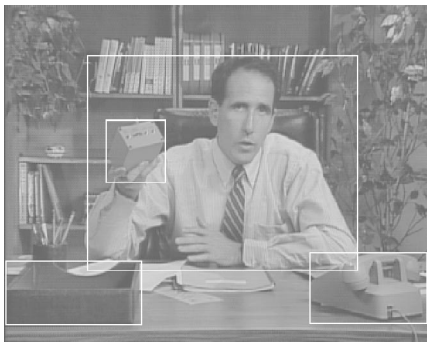


Fig. 3. Video frame 1. There are four semantic objects: *salesman*, *box*, *file holder*, and *telephone*; *salesman* is the target semantic object. The relative position numbers (as defined in Table 1) of the other three semantic objects are 10, 15, and 24, respectively.

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 multimedia input string then can be formed after relative positions are obtained. Del Bimbo et al. [4] proposed a region-based formulation with a rectangular partitioning. Therefore, each object stands over one or more regions. Table 2 follows the same principle but directly captures the relative positions among objects. Relative positions are explicitly indicated by numbers to capture the spatial relations and moving history. Different input symbols in multimedia input strings represent different time durations in a video sequence. These input symbols in multimedia input strings are the arc labels for the subnetworks in ATNs. In ATNs, when an arc contains one or more images, video segments, or texts, then one subnetwork with the media stream as the starting state name is created. A new arc and a new state node in a subnetwork and a new input symbol in a multimedia input string are created when any relative position of a semantic object changes or the number of semantic objects changes. The subnetwork design is similar to the VSDG model [12] which uses transitions to represent the number of semantic object changes.

Fig. 4. Video frame 52. Semantic object *box* moves from the left to the front of *salesman* (from viewer's point of view).

Fig. 5. Video frame 70. Semantic object *box* moves from the front to the left of *salesman* (from viewer's point of view).

However, in the VSDG model, a new transition is created when the number of semantic objects changes and a motion vector is taken with each node. In our design, in addition to the changes of the number of semantic objects, any relative position change among semantic objects is considered and a state node and an arc in the subnetwork and an input symbol in the multimedia input string are created for this situation. Based on our design, the temporal relations and the relative positions of semantic objects can be obtained and the moving histories of the semantic objects in the video sequence can be kept. Therefore, substring matching processes using multimedia input strings in database queries can be conducted.

3.2 Multimedia Database Searching Examples

Figs. 3, 4, and 5 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 Fig. 3, 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 Fig. 4, and moves back to *left* in Fig. 5. The following multimedia input string can be used to represent Figs. 3, 4, and 5 as follows:

Multimedia input string:

$$\underbrace{(S_1\&B_{10}\&F_{15}\&T_{24})}_{X_1}\underbrace{(S_1\&B_3\&F_{15}\&T_{24})}_{X_2}\underbrace{(S_1\&B_{10}\&F_{15}\&T_{24})}_{X_3}. \tag{3.1}$$

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 they have the same relative position numbers in X_1, X_2, X_3. As we can see from this example, the multimedia input string can represent not only the relative positions of the semantic objects but also the motion of the semantic objects. For example, the above multimedia input string shows that the semantic object *box* moves from *left* to *front* relative to the target semantic object *salesman*. Fig. 6 is a

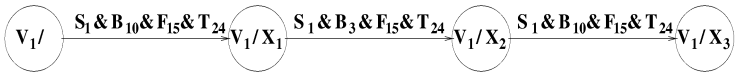


Fig. 6. The corresponding subnetwork for multimedia input string (3.1).

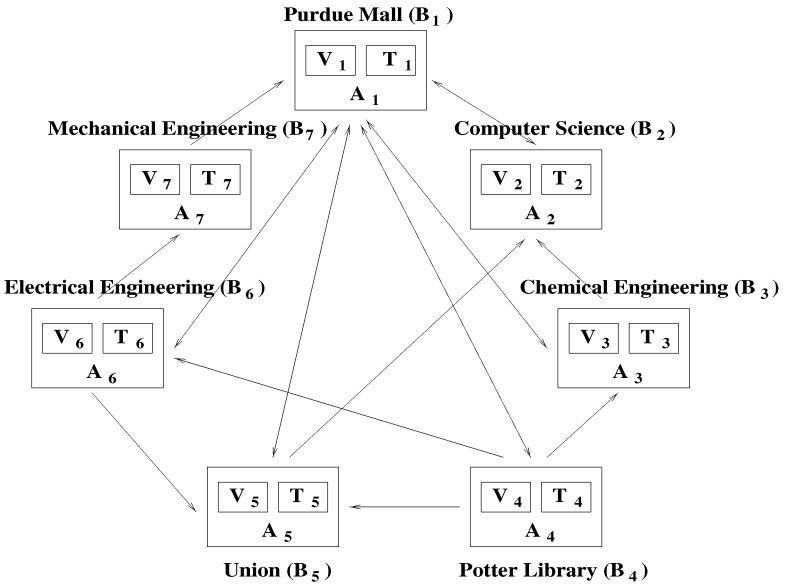


Fig. 7. A browsing graph of a minitour of the Purdue University campus: There are seven sites denoted by B_i, $i = 1 \ldots 7$ that are connected by arcs. A directed arc denotes a one-way selection and a bidirection arc allows two-way selections.

subnetwork for multimedia input string (3.1). We assume this subnetwork models the media stream V_1 in Fig. 2. Therefore, the starting state name for this subnetwork is $V_1/$. As shown in Fig. 6, there are three arcs with arc labels the same as the three input symbols in (3.1).

3.2.1 A Spatial Database Query Example

In a multimedia information system, the designers can design a general purpose presentation for a class of applications so that it allows users to choose what they prefer to see using database queries. Assume that there are several video media streams and V_s is one of them. The multimedia input string for the subnetwork which models V_s is the same as (3.1).

Query 1. 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. A user can issue this query using a high-level language. This query then is translated into a multimedia input string as follows: Multimedia input string:

$$(S_1 \& B_{10}). \tag{3.2}$$

The subnetworks of an ATN are traversed and the corresponding multimedia input strings are analyzed. Suppose the multimedia input string in (3.1) is models 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* from a viewer's perspective. By matching (3.2) with (3.1), 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.

3.2.2 A Spatio-Temporal Database Query Example

Query 2. 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*. This query is translated into a multimedia input string which is the same as (3.1). Again, each of the subnetworks needs to be checked one by one. The same as in the previous query, the relative positions to be matched are based on the views that users see. The first condition in this query asks to match the relative position of the *box* to the left of the *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 in which the *box* is to the left of the *salesman*. Next, the relative position of the *box* is moved from the left to the front of the *salesman*. This is satisfied by the input symbol X_2 since B_3 indicates that the *box* is to the *front* of the *salesman*. Finally, it needs to match the relative position of the *box* to be back to the left of the *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.

From this section, we know that after the subnetworks and their multimedia input strings are constructed by the designer, users can issue database queries related to the temporal and spatial relations of semantic objects using high-level database query languages. These queries are translated into multimedia input strings to match the

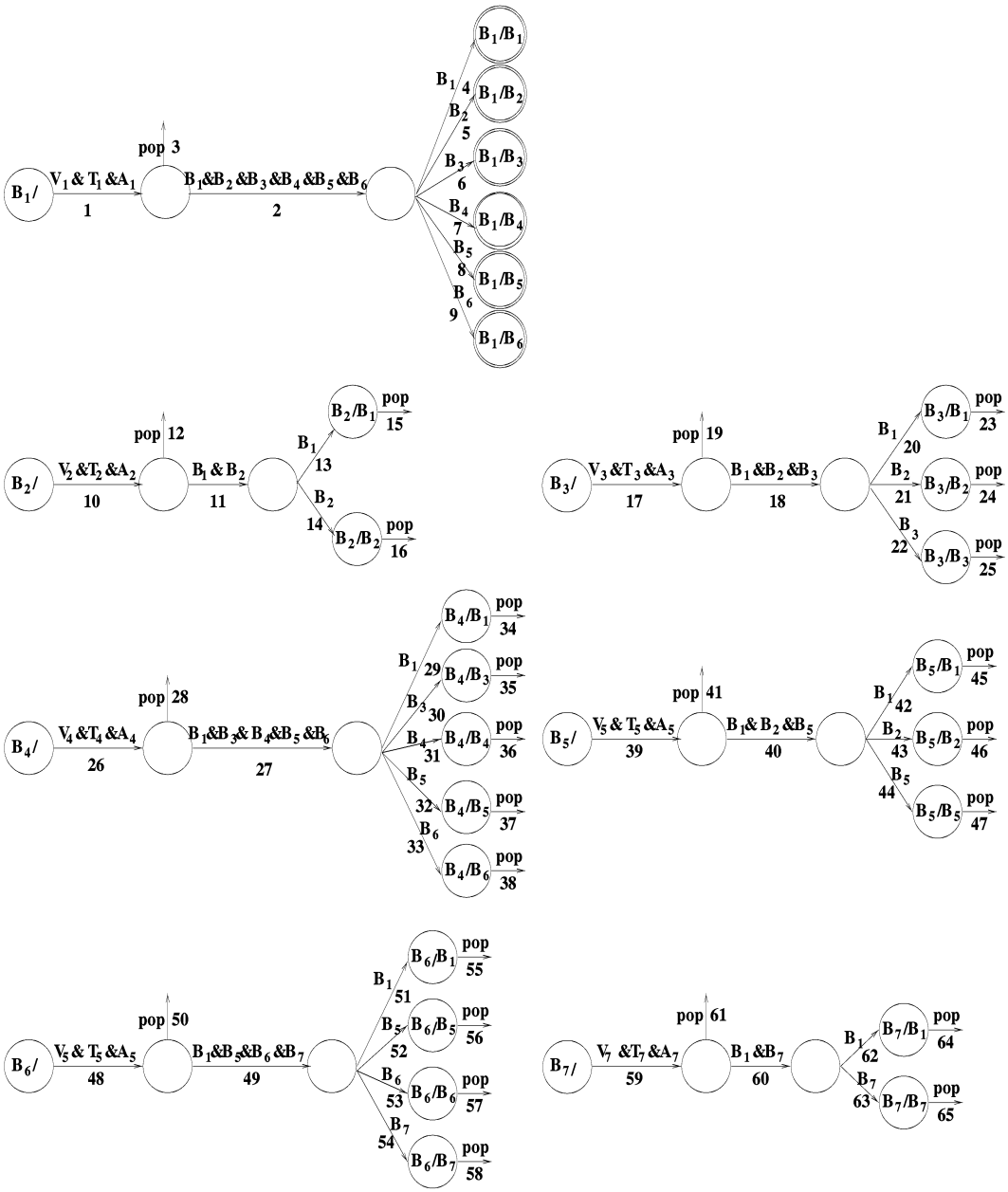


Fig. 8. ATN for the minitour of a campus: Seven networks represent seven sites which users can browse. Networks $B_1/$ through $B_7/$ represent the presentations for sites Purdue Mall, Computer Science Building, Chemical Engineering Building, Potter Library, Union, Electrical Engineering Building, and Mechanical Engineering Building, respectively. Each network begins a presentation with three media streams: a video, a text, and an audio, and is followed by selections. After a user selects a site, the control will pass to the corresponding network so that the user can watch the presentation for that site continuously.

multimedia input strings of the subnetworks that model image and video media streams. Under this design, multimedia database queries related to images or video frames can be answered. The details of the multimedia input strings, the translation from high-level queries to multimedia input strings, and the matching processes are transparent to users. ATNs and multimedia input strings are the internal data structures and representations in DBMS. After users issue queries, the latter processes are handled by DBMS. Separating the detailed internal processes from users can reduce the burden of users so that the multimedia information system is easy to use.

4 Using ATNs and Multimedia Input Strings to Model Multimedia Browsing

Fig. 7 shows a browsing graph of a minitour of a campus. The browsing graph consists of a set of nodes and arcs connecting them. There are seven sites in this browsing graph: Purdue Mall, Computer Science, Chemical Engineering, Potter Library, Union, Electrical Engineering, and Mechanical Engineering. We use B_i, $i = 1 \ldots 7$ to represent these seven sites. For each site, a presentation consists of video, text, and audio media streams which are denoted by V_i, T_i, and A_i. A directed arc denotes a one-way selection. For example, there is a directed arc pointing from the

Mechanical Engineering building to the Purdue Mall. This means that after a user watches the presentation for the Mechanical Engineering building, he/she can immediately watch the Purdue Mall presentation. However, the opposite direction is inapplicable since there is no directed arc pointing from the Purdue Mall to the Mechanical Engineering building. The bidirection arcs allow users to go back and forth between two locations such as the Purdue Mall and the Potter Library. For example, after a user watches the presentation for Purdue Mall, he/she can choose Computer Science, Chemical Engineering, Potter Library, Union, and Electrical Engineering buildings to watch. He/She can also watch the presentation for the Purdue Mall again. Fig. 8 is the ATN for the browsing graph in Fig. 7. For simplicity, some state names are not shown in Fig. 8. Assume the browsing always starts from the Purdue Mall (B_1). There are seven networks in Fig. 8 and each network represents a site. In a user interaction environment, users may start from any site to watch. Before a detailed discussion of how ATN models multimedia browsing, the arc types together with the notation need to be defined. We adopted the following notation and definition as in [1]:

- **Push** arc succeeds only if the named network can be successfully traversed. The state name at the head of the arc will be pushed into a stack and the control will be passed to the named network.
- **Pop** arc succeeds and signals the successful end of the network. The topmost state name will be removed from the stack and become the return point. Therefore, the process can continue from this state node.
- **Jump** arc always succeeds. This arc is useful when passing the control to any state node.

A detailed trace of the following browsing sequence is used to illustrate how the ATN works.

1. Purdue Mall,
2. Chemical Engineering,
3. Computer Science, and
4. Computer Science.

In this browsing example, the Purdue Mall is the first to be visited and followed by the Chemical Engineering building. Then, the Computer Science building is the next one to be watched. The Computer Science building is viewed one more time and then the tour stops. Table 3 shows the trace for this browsing example.

Step 1. The current state is in $B_1/$, where B_1 represents the Purdue Mall. The following arc is arc number 1 and the input symbol is $V_1\&T_1\&A_1$. This input symbol denotes video 1, text 1, and audio 1 which are displayed concurrently.

Step 2. Arc number 2 follows and the input symbol $B_1\&B_2\&B_3\&B_4\&B_5\&B_6$ is read so that users can choose a site from site 1 through 6 to watch.

Step 3. Based on the browsing sequence as specified above, the Chemical Engineering building (B_3) is chosen so that arc number 6 follows and input symbol B_3 is read. Since B_3 is a subnetwork name, the state name pointed by arc number 6 (B_1/B_3) is pushed into a stack. A stack follows the last-in-first-out (LIFO) policy which only allows retrieving the topmost state name first.

Step 4. The control is passed to the subnetwork with starting state name $B_3/$. Arc number 17 follows and the input symbol $V_3\&T_3\&A_3$ is read so that video 3, text 3, and audio 3 are displayed.

Step 5. Arc number 18 follows and the input symbol $B_1\&B_2\&B_3$ is read so that users can choose from site 1 through 3 to watch.

Step 6. As specified above, the Computer Science building (B_2) is the next site to watch so that arc number 21 follows and the input symbol B_2 is read. Since B_2 is a subnetwork name, the state name pointed by this arc is pushed into the stack. Therefore, there are two state names in this stack: B_3/B_2 and B_1/B_3.

Step 7. The control passes to the subnetwork with starting state name $B_2/$. Arc number 10 is followed and the input symbol is $V_2\&T_2\&A_2$ so that video 2, text 2, and audio 2 are displayed.

Step 8. Arc number 11 follows and the input symbol $B_1\&B_2$ is read. Users can choose either site 1 or 2.

Step 9. User interactions allow users to interact with multimedia information systems. Users may want to watch some topic recursively or have user loops so that they have the opportunity to select different contents after viewing a previous selection. ATNs allow recursion, that is, a network might have an arc labeled with its own name. This feature allows ATNs to have the ability to model user loops and recursion easily. As specified above, the Computer Science building (B_2) is watched again. The state name pointed by arc number 14 (B_2/B_2) is pushed into the stack so that there are three state names stored in this stack.

Step 10. The control passes back the same subnetwork and video 2, text 2, and audio 2 are displayed concurrently again.

Step 11. After Step 10, as specified above, the presentation stops so that arc number 12 follows with a pop arc label. The topmost state name in the stack (B_2/B_2) is popped out so that the control passes to the state node with state name B_2/B_2. The stack has two state names now.

Step 12. Arc number 16 follows with a pop arc label. Therefore, the topmost state name B_3/B_2 is popped out and the control is passed to it.

Step 13. The current state name is B_3/B_2 and arc number 24 follows with a pop arc label. The only state name in the stack is popped out and the control is passed to it.

Step 14. The current state is B_1/B_3 which is a final state (no outgoing arc) so that the browsing stops.

ATN and its subnetworks in Fig. 8 depict the structural hierarchy of the browsing graph in Fig. 7. Under ATNs, user interactions are represented by using more than one outgoing arc with different arc labels for a state node. User loops are modeled by using recursions with arcs labeled by network names. By using the recursion, one can avoid many arcs which point back to the previous state nodes. This

TABLE 3
The Trace of ATN for the Specified Browsing Sequence

Step	Current State	Input Symbol	Arc Followed	Backup States
1	$B_1/$	$V_1\&T_1\&A_1$	1	NIL
2	$B_1/V_1\&T_1\&A_1$	$B_1\&B_2\&B_3\&B_4\&B_5\&B_6$	2	NIL
3	$B_1/B_1\&B_2\&B_3\&B_4\&B_5\&B_6$	B_3	6	B_1/B_3
4	$B_3/$	$V_3\&T_3\&A_3$	17	B_1/B_3
5	$B_3/V_3\&T_3\&A_3$	$B_1\&B_2\&B_3$	18	B_1/B_3
6	$B_3/B_1\&B_2\&B_3$	B_2	21	B_3/B_2 B_1/B_3
7	$B_2/$	$V_2\&T_2\&A_2$	10	B_3/B_2 B_1/B_3
8	$B_2/V_2\&T_2\&A_2$	$B_1\&B_2$	11	B_3/B_2 B_1/B_3
9	$B_2/B_1\&B_2$	B_2	14	B_2/B_2 B_3/B_2 B_1/B_3
10	$B_2/$	$V_2\&T_2\&A_2$	10	B_2/B_2 B_3/B_2 B_1/B_3
11	$B_2/V_2\&T_2\&A_2$	pop	12	B_3/B_2 B_1/B_3
12	B_2/B_2	pop	16	B_1/B_3
13	B_3/B_2	pop	24	NIL
14	B_1/B_3 (Finish)			

makes the whole network structure become less complicated. Moreover, the browsing sequences in Fig. 7 are preserved by traversing the ATN.

5 Using ATNs and Multimedia Input Strings to Model User Interactions and Loops

Figs. 9a, 9b, 9c, 9d, 9e, and 9f are the timelines for two multimedia presentations. Figs. 9a and 9b are the starting timelines for presentations P_1 and P_2, respectively. In Fig. 9a, media streams V_1 (video stream 1) and T_1 (text 1) start to display at time t_1 and media streams I_1 (image 1) and A_1 (audio stream 1) join to display at time t_2 in presentation P_1. These four media streams all end at time t_3. In presentation P_2, as shown in Fig. 9b, V_2 and T_2 start at t_1 and end at t_3. Then, presentation P_1 goes to the presentation shown in Fig. 9c and presentation P_2 continues its presentation, as shown in Fig. 9d. Presentations P_1 and P_2 join again at time t_5, where two choices are provided to allow users to choose based on their preference. A timeline model is inapplicable to model the alternative situations so that it is difficult to model this user interaction scenario. As

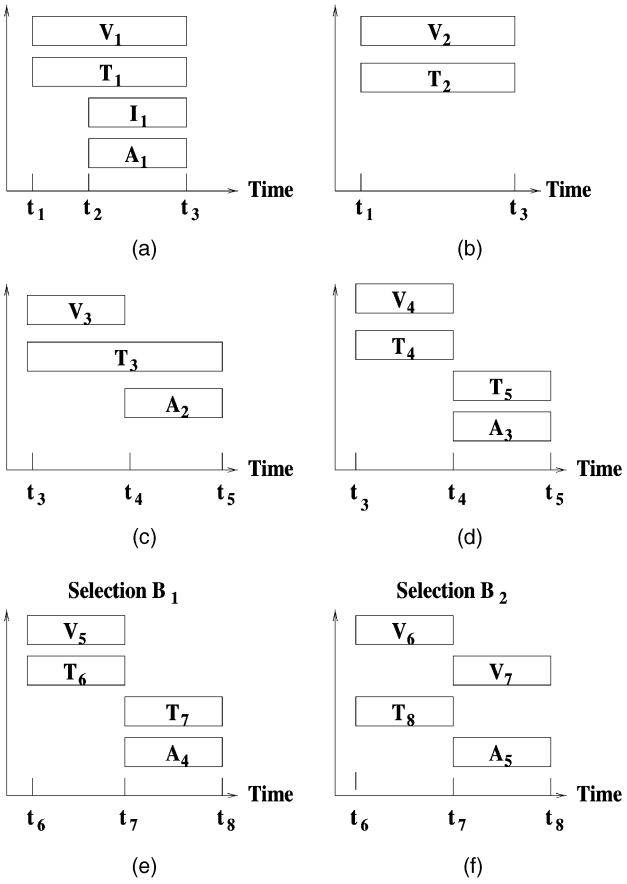


Fig. 9. Timelines for presentation P_1 and P_2: (a), (c), (e), and (f) are the presentation sequences for presentation P_1. (b), (d), (e), and (f) are the presentation sequences for presentation P_2. (e) and (f) are two timelines for selections B_1 and B_2, respectively.

shown in Fig. 9, we cannot directly tell that Figs. 9e and 9f are two timelines for different selections. Since user thinking time is unknown in advance, the starting time for media streams V_5 and T_6 and the starting time for V_6 and T_8 will not be known until a user makes a choice. Let's assume the user makes a choice at time t_6. Figs. 9e and 9f are the timelines for selections B_1 and B_2. Selection B_1 has V_5 and T_6 displayed at time t_6. These two media streams end at time t_7, where T_7 and A_4 begin to display and end at time t_8. Selection B_2 has V_6 and T_8 displayed from time t_6 to time t_7, and V_7 and A_5 start at time t_7 and end at time t_8. If B_1 is chosen, the presentation stops. However, if B_2 is chosen, it allows the user to make the choice again. The timeline representation cannot reuse the same timelines as in Figs. 9e and 9f and, therefore, it needs to create the same information again. Since we don't know how many loops users will go through in this part, it is impractical to use this stand alone timeline representation to model user loops.

When the designer designs the start and end times of media streams as shown in Fig. 9, the multimedia input string can be constructed automatically based on the starting and ending time of media streams. In presentation P_1, the multimedia input string is:

$$\underbrace{(V_1\&T_1)}_{X_1}\underbrace{(V_1\&T_1\&I_1\&A_1)}_{X_2}\underbrace{(V_3\&T_3)}_{X_4}\underbrace{(T_3\&A_2)}_{X_5}(\underbrace{(B_1\&B_2)}_{X_8}$$
$$(\underbrace{(V_5\&T_6)}_{X_9}\underbrace{(T_7\&A_4)}_{X_{10}}\,\$\mid\underbrace{(V_6\&T_8)}_{X_{11}}\underbrace{(V_7\&A_5)}_{X_{12}}))^{+}. \tag{5.1}$$

In presentation P_2, the multimedia input string is:

$$\underbrace{(V_2\&T_2)}_{X_3}\underbrace{(V_4\&T_4)}_{X_6}\underbrace{(T_5\&A_3)}_{X_7}(\underbrace{(B_1\&B_2)}_{X_8}$$
$$(\underbrace{(V_5\&T_6)}_{X_9}\underbrace{(T_7\&A_4)}_{X_{10}}\,\$\mid\underbrace{(V_6\&T_8)}_{X_{11}}\underbrace{(V_7\&A_5)}_{X_{12}}))^{+}. \tag{5.2}$$

As mentioned earlier, a multimedia input string is used to represent the presentation sequence. In presentation P_1, the input symbol X_1 contains V_1 and T_1 which start at the same time and play concurrently. Later, I_1 and A_1 begin and overlap with V_1 and T_1. Therefore, the input symbol X_2 contains the media streams V_1, T_1, I_1, and A_1. Each image, video, or text media stream has its own multimedia input string and a subnetwork is created. Figs. 10b to 10d are part of the subnetworks of P_1 to model V_1, I_1, and T_1, respectively. For simplicity, the subnetworks of other image, video, and text media streams are not shown in Fig. 10.

The delay time for I_1 and A_1 to display does not need to be specified in a multimedia input string explicitly since the multimedia input string 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. The "|" symbol represents the alternatives for different selections. The "$" symbol denotes the end of a presentation.

Fig. 10 shows the use of a single ATN to model presentations P_1 and P_2 which include user interactions and loops. Fig. 10a is an ATN modeling presentations P_1 and P_2. P_1 and P_2 start at different starting states. Table 4 is a trace of ATN for presentation P_1 in Fig. 10 and is used to explain how ATN works.

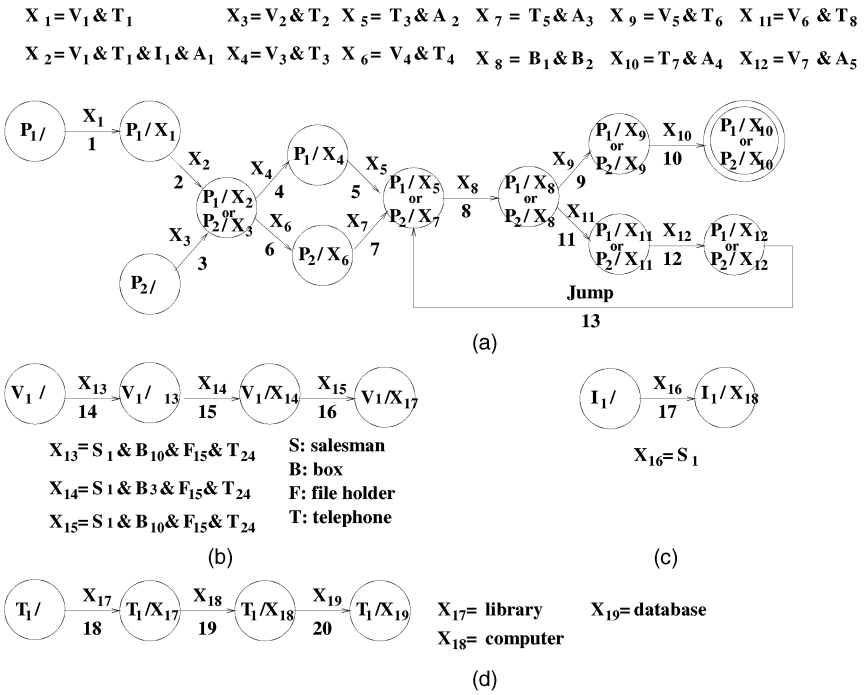
Step 1. The current state is P_1 and the arc to be followed is arc number 1 with arc label X_1. Media streams V_1 and T_1 are displayed. There is no backup state in the stack.

Step 2. The current state is P_1/X_1 which denotes X_1 has been read in presentation P_1. Arc number 2 with arc label X_2 is the arc to be followed. X_2 consists of media streams V_1, T_1, I_1, and A_1.

Step 3. In presentation P_1, after X_1 and X_2 are displayed, the ATN reaches state P_1/X_2 which has two outgoing arcs (arc numbers 4 and 6). The presentation P_1 needs to follow arc 4 so that input symbol X_4 is read. Media streams V_3 and T_3 are displayed at this duration.

Step 4. The current state is P_1/X_4 and arc number 5 is followed. Media streams T_3 and A_2 are displayed.

Step 5. 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 finite state machines. However, the conditions and actions in ATNs have the ability to model user interactions. In Fig. 10a, after the state P_1/X_5 is met, the input symbol X_8



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

(e)

Fig. 10. Augmented Transition Network: (a) is the ATN network for two multimedia presentations which start at the states $P_1/$ and $P_2/$, respectively. (b), (c), and (d) are part of the subnetworks of (a). (b) models the semantic objects in video stream V_1, (c) models the semantic objects in image media stream I_1, and (d) models the keywords in text stream T_1. In (e), the "Get" procedure accesses an individual media stream. "Display" displays the media streams. "$\mathrm{Next\_Symbol}(X_i)$" reads the input symbol X_i. The "Next_State" is a procedure to advance to the next state. "$\mathrm{Start\_time}(X_i)$" gives the prespecified starting time of X_i. User thinking time is accounted for by the **Delay** variable. θ is a parameter.

with two selections B_1 and B_2 is displayed. Before a choice is made, a thinking time should be kept. A **Delay** variable is used to represent the delay of the presentation. The "Start_time(X_i)" procedure gives the prespecified starting time of X_i. The difference between the current time and the prespecified starting time is the total display time so far. Since a delay time occurs after user interactions, the presentation sequence needs to be shifted by the delay. The process continues until the final state is reached. Fig. 10e shows the detailed condition column and action column for user interactions. As illustrated in Fig. 10a, two choices B_1 and B_2 are provided to let users make their selections. The corresponding action calculating the delay time is activated in the action column. The calculated delay time is used to

TABLE 4
The Trace of ATN for Presentation P_1

Step	Current State	Input Symbol	Arc Followed	Backup States
1	$P_1/$	X_1	1	NIL
2	P_1/X_1	X_2	2	NIL
3	P_1/X_2	X_4	4	NIL
4	P_1/X_4	X_5	5	NIL
5	P_1/X_5	X_8	8	NIL

TABLE 5
Continuation of Table 4 if B_1 is Chosen

Step	Current State	Input Symbol	Arc Followed	Backup States
6	P_1/X_8	X_9	9	NIL
7	P_1/X_9	X_{10}	10	NIL
8	P_1/X_{10} (Finish)			

TABLE 6
Continuation of Table 4 if B_2 is Chosen

Step	Current State	Input Symbol	Arc Followed	Backup States
6	P_1/X_8	X_{11}	11	NIL
7	P_1/X_{11}	X_{12}	12	NIL
8	P_1/X_{12}	Jump	13	NIL
9	Go back to Step 5			

shift the starting time of any media stream displayed after the selection.

Step 6. If B_1 is selected, arc number 9 is followed and media streams V_5 and T_6 are displayed (as shown in Table 5). If B_2 is selected, arc number 11 is followed and media streams V_6 and T_8 are displayed (as shown in Table 6).

Step 7. In Table 5, the current state is P_1/X_9, arc number 10 is followed and media streams T_7 and A_4 are displayed since B_1 is selected. In Table 6, the current state is P_1/X_{11}, arc number 12 is followed, and media streams V_7 and A_5 are displayed since B_2 is selected.

Step 8. When B_1 is selected, the current state is P_1/X_{10} and the presentation stops. When B_2 is selected, arc number 13 is followed and a "Jump" arc label is met. The control is passed back to state P_1/X_5.

Step 9. When B_2 is selected, the process goes back to Step 5 to let the user make a choice again. Steps 5 through 9 model a loop scenario which is represented by a "+" symbol in multimedia input strings (5.1) and (5.2). The "Jump" action does not advance the input symbol but lets the control go to the pointing state. That means the "Jump" itself is not an input symbol in multimedia input strings. 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.

In Fig. 10c, there is only one input symbol for the subnetwork modeling I_1. The input symbol is X_{16} which contains the semantic object *salesman*. Fig. 10d is the subnetwork for T_1. The input symbols for T_1 consist of three keywords appearing in the sequence *library*, *computer*, and *database*. Presentation P_2 is similar to P_1 except that X_3 is displayed first, and X_6 and X_7 are displayed after X_3. That is, presentations P_1 and P_2 share arc number 8. Fig. 10e

shows an example of how to use conditions and actions to maintain synchronization and quality-of-service (QoS). In presentation P_1, when the current input symbol X_1 (V_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 will be transmitted instead of V_1. Then, the prespecified duration to display V_1 and T_1 is checked. If the duration is not reached, the display continues. Otherwise, a new input symbol is read and the control is passed to the next state. The same conditions are checked for other input symbols, too.

6 Related Work

Some of the semantic models developed in the recent past are graphical structures for modeling multimedia presentations. Semantic models such as OCPN [18], MDS [8], Firefly [6], and Hirzalla's et al. graphical temporal model [15] all fall into this category. In other multimedia database systems [13], [20], [21], the emphasis is on presenting different media streams to users with query specifications. These models provide searching capabilities allowing users to retrieve information from the database.

Little and Ghafoor [18], [19] 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 for users to understand the presentation sequence of media streams. 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 parallel activities but not alternative activities [22] such as user interactions. Many later abstract semantic models are based on petri-net or OCPN [2], [8], [17], [23].

Oomoto and Tanaka [20] 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.

Hirzalla et al. [15] developed a graphical temporal model for interactive multimedia documents to expand the traditional timeline models such as Blakowski et al. [5] 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 of the user delays in the user selections so that the later presentations can be shifted.

7 Conclusions

In this paper, we describe an ATN-based model together with multimedia input strings for multimedia presentations, multimedia database searching, and multimedia browsing. Therefore, ATNs provide three major capabilities: *presentation*, *querying*, and *browsing*. ATNs are left to right models that are used to model a presentation sequence from the beginning to the end. Each arc label is a string that consists of those media streams to be displayed. A subnetwork and the corresponding multimedia input string are created for an image, a video, or a text media stream to facilitate querying capabilities in ATNs. Subnetworks are used to model the temporal and spatial information of semantic objects for images and video sequences. The keyword sequences in text media streams can also be included in the subnetworks. Querying capabilities allow users to retrieve information related to media streams or semantic objects in a specific presentation directly. Since a great deal 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. 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 user's 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 starting time. Moreover, ATNs allow loops in a presentation. Loops can be used to let some part of a presentation be watched more than once. The browsing capabilities are achieved by using user interactions and user loops. This browsing capability gives users greater flexibility in watching a presentation sequence based on their selections. Unlike the existing semantic models which model presentations, querying, or browsing, our ATN model provides these three capabilities in one framework.

Acknowledgments

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

References

[1] J. Allen, *Natural Language Understanding*. The Benjamin/Cummings Publishing Company, Inc., 1995.
[2] Y.Y. Al-Salqan and C.K. Chang, "Temporal Relations and Synchronization Agents," *IEEE Multimedia*, pp. 30-39, Summer 1996.

[3] R. Bayer and E. McCreight, "Organization and Maintenance of Large Ordered Indices," *Proc. 1970 ACM-SIGFIDENT Workshop Data Description and Access,* pp. 107-141, Nov. 1970.
[4] A.D. Bimbo, E. Vicario, and D. Zingoni, "Symbolic Description and Visual Querying of Image Sequences Using Spatio-Temporal Logic," *IEEE Trans. Software Eng.,* vol. 7, no. 4, pp. 609-621, Aug. 1995.
[5] G. Blakowski, J. Huebel, and U. Langrehr, "Tools for Specifying and Executing Synchronized Multimedia Presentations," *Proc. Second Int'l Workshop Network and Operating System Support for Digital Audio and Video,* pp. 271-279, 1991.
[6] M. Buchanan and P. Zellweger, "Automatically Generating Consistent Schedules for Multimedia Documents," *ACM Multimedia Systems J.,* vol. 1, no. 2, pp. 55-67, 1993.
[7] S.K. Chang, C.W. Yan, D.C. Dimitroff, and T. Arndt, "An Intelligent Image Database System," *IEEE Trans. Software Eng.,* vol. 14, no. 5, pp. 681-688, May 1988.
[8] H.J. Chang, T.Y. Hou, and S.K. Chang, "The Management and Application of Teleaction Objects," *ACM Multimedia Systems J.,* vol. 3, pp. 228-237, Nov. 1995.
[9] S.-C. Chen and R.L. Kashyap, "Temporal and Spatial Semantic Models for Multimedia Presentations," *Proc. 1997 Int'l Symp. Multimedia Information Processing,* pp. 441-446, 1997.
[10] S.-C. Chen and R.L. Kashyap, "Empirical Studies of Multimedia Semantic Models for Multimedia Presentations," *Proc. 13th Int'l Conf. Computer and Their Applications,* pp. 226-229, 1998.
[11] D. Comer, "The Ubiquitous B-tree," *Computing Surveys,* vol. 11, no. 2, pp. 121-138, June 1979.
[12] Y.F. Day, S. Dagtas, M. Iino, A. Khokhar, and A. Ghafoor, "Object-Oriented Conceptual Modeling of Video Data," *Proc. IEEE 11th Int'l Conf. Data Eng.,* pp. 401-408, 1995.
[13] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker, "Query by Image and Video Content: The QBIC System," *Computer,* vol. 28, no. 9, pp. 23-31, Sept. 1995.
[14] A. Guttman, "R-tree: A Dynamic Index Structure for Spatial Search," *Proc. ACM SIGMOD,* pp. 47-57, June 1984.
[15] N. Hirzalla, B. Falchuk, and A. Karmouch, "A Temporal Model for Interactive Multimedia Scenarios," *IEEE Multimedia,* pp. 24-31, Fall 1995.
[16] S.C. Kleene, *Representation of Events in Nerve Nets and Finite Automata, Automata Studies.* Princeton, N.J.: Princeton Univ. Press, pp. 3-41, 1956.
[17] C.C. Lin, J. Xiang, and S.K. Chang, "Transformation and Exchange of Multimedia Objects in Distributed Multimedia Systems," *ACM Multimedia Systems J.,* vol. 4, pp. 12-29, Feb. 1996.
[18] T.D.C. Little and A. Ghafoor, "Synchronization and Storage Models for Multimedia Objects," *IEEE J. Selected Areas in Comm.,* vol. 9, pp. 413-427, Apr. 1990.
[19] T.D.C. Little and A. Ghafoor, "Interval-Based Conceptual Models for Time-Dependent Multimedia Data," *IEEE Trans. Knowledge and Data Eng.,* vol. 5, no. 4, pp. 551-563, Aug. 1993.
[20] E. Oomoto and K. Tanaka, "OVID: Design and Implementation of a Video Object Database System," *IEEE Trans. Knowledge and Data Eng.,* vol. 5, no. 4, pp. 629-643, Aug. 1993.
[21] M.T. Özsu, D. Duane, G. El-Medani, and C. Vittal, "An Object-Oriented Multimedia Database System for a News-on-Demand Application," *ACM Multimedia Systems J.,* vol. 3, pp. 182-203, Nov. 1995.
[22] J.L. Peterson, "Petri Nets," *ACM Comput. Surveys,* vol. 9, pp. 223-252, Sept. 1977.
[23] H. Thimm and W. Klas, "δ-Sets for Optimized Relative Adaptive Playout Management in Distributed Multimedia Database Systems," *Proc. IEEE 12th Int'l Conf. Data Eng.,* pp. 584-592, 1996.
[24] W. Woods, "Transition Network Grammars for Natural Language Analysis," *Comm. ACM,* vol. 13, pp. 591-602, Oct. 1970.

Shu-Ching Chen received the PhD degree from the School of Electrical and Computer Engineering from Purdue University, West Lafayette, Indiana, in 1998. He also received the computer science, electrical engineering, and civil engineering master degrees from Purdue University, West Lafayette, Indiana. He has been an assistant professor in the School of Computer Science, Florida International University (FIU) since August 1999. Before joining FIU, he worked as a R&D software engineer at Micro Data Base Systems (MDBS), Inc., Indiana. His main research interests include distributed multimedia database systems and information systems, information retrieval, object-oriented database systems, data warehousing, data mining, and distributed computing environments for intelligent transportation systems (ITS). He was the program cochair of the Second International Conference on Information Reuse and Integration (IRI-2000). He is a member of the IEEE, the IEEE Computer Society, the ACM, and ITE.

R.L. Kashyap received the PhD degree in 1966 from Harvard University, Cambridge, Massachusetts. He joined the staff of Purdue University in 1966, where he is currently a professor of electrical and computer engineering and the associate director of the US National Science Foundation supported Engineering Research Center Intelligent Manufacturing Systems at Purdue. He is currently working on research projects supported by the US Office of Naval Research, Army Research Office, US National Science Foundation, and several companies like Cummins Engines. He has directed more than 40 PhD dissertations at Purdue. He has authored one book and more than 300 publications, including 120 archival journal papers in areas, such as pattern recognition, random field models, intelligent data bases, and intelligent manufacturing systems. He is a fellow of the IEEE.