Journal of Experimental & Theoretical Artificial Intelligence,
Vol. 17, No. 1–2, January–June 2005, 5–18

Taylor & Francis
Taylor & Francis Group

Towards structure discovering in video data

ELISA BERTINO[†]*, MOHAND-SAID HACID[‡]
and FAROUK TOUMANI[§]

[†]Universitá degli Studi di Milano, Via Comelico, 39 20135 Milano, Italy
[‡]University Lyon 1, 43 blvd du 11 novembre 1918, 69622 Villeurbanne, France
[§]ISIMA—Campus des Cezeaux—B.P. 125, 63173 AUBIERE, France

(*Received 1 March 2004; in final form 1 August 2004*)

Digital images and video clips are becoming popular due to the increase in the availability of consumer devices that capture them. Digital content is also growing over the Internet. Applications that benefit from video are education and training, marketing support, medical, etc. The increase of this digital content creates a need for user-friendly tools to browse through large volumes of digital material. However, there are two basic impediments to wider use of digital video. The first is cataloging, which includes video digitization, compression and annotation, and the second is the lack of fast and effective search and browse techniques for this massive video content. The authors are interested in this second problem. One method that they believe is promising is the augmentation of a metadatabase with information on video content so that users can be guided to appropriate data sets. An automated technique is presented that combines manual annotations and knowledge produced by an automatic content characterization technique (i.e. clustering algorithms) to build higher level abstraction of video content.

Keywords: Mining multimedia data; Concepts classification; Description logics

1. Introduction

The availability of a huge amount of multimedia data, images and videos calls for efficient methods and tools to index and retrieve such data types. The interest in this topic is witnessed by several special issues in journals and definitely by the 'work-in-progress' MPEG-7 standard. Images and videos are typical unstructured

*Corresponding author. Email: bertino@dico.unimi.it

Journal of Experimental & Theoretical Artificial Intelligence
ISSN 0952-813X print/ISSN 1362-3079 online © 2005 Taylor & Francis Ltd
http://www.tandf.co.uk/journals
DOI: 10.1080/09528130512331315927

documents widely employed in a variety of communication frameworks such as entertainment, advertising, sport, education, publishing, etc.

Content-based video browsing and retrieval have become important research topics in recent years. Research interest in this field has escalated because of the proliferation of video data in digital form. Content-based retrieval of a video stream, segmented into its constituent elements, requires that content indexes are set. We distinguish indexes on semantic primitives, such as objects and motion, and indexes on the meaning conveyed by visual primitives. Object primitives are usually extracted from key-frames and used for comparison with primitives extracted from a query image.

For databases with large numbers of video sequences, it is not feasible to browse linearly through the videos in the database. A desirable characteristic of a browsing system is to let the user navigate through the database in a structured manner.

We develop an approach to automatically create a hierarchical clustering of the videos in the database and use the resulting tree structure to navigate through the videos. This approach can be applied to key-frames extracted from video segments to enable the users to browse through their video libraries. It should be clear that the usefulness of this approach depends on the robustness of the hierarchical clustering results. If some videos are grouped incorrectly, the user may not be able to find them by browsing through the tree.

To represent the various aspects of a video object's characteristics, we propose a model that consists of two layers: (1) Feature & Content Layer, intended to contain video visual features such as colour, texture and shape; (2) Semantic Layer, which provides the (conceptual) content dimension of video data. Objects of the Semantic Layer make reference to objects of the Feature & Content Layer through a particular attribute called *sequence*. On top of these two layers, we propose to automatically build a Schema Layer, which will contain the structured abstractions of video sequences, i.e. a general schema about the classes of objects represented in the previous layers. This process is shown in figure 1. To do that, we assume video sequences in the Feature & Content Layer are clustered according to some algorithms. Video sequences are grouped by similarity. Each cluster is seen as a class. Given those clusters and data of the Semantic Layer[†], we proceed in two steps for building the Schema Layer: (1) computing the most specific abstraction of each object in the Semantic Layer; and (2) a minimal representation of the (partial) structure of the Semantic Layer is then derived by computing *least common subsumers* of appropriate concepts generated in the first step. To our knowledge, this is the first proposal for automatically building an abstract layer, in the form of structural associations, on top of a video database.

This paper is organized as follows. Section 2 discusses related work in knowledge extraction for video browsing. In section 3, we provide the syntax and semantics of the concept language we use for describing the extracted structural abstractions. Section 4 develops our algorithms for discovering structural associations from video content descriptions. We conclude in section 5.

[†] Data in this layer is expressed in a semistructured data model proposed in Hacid *et al.* (2000).

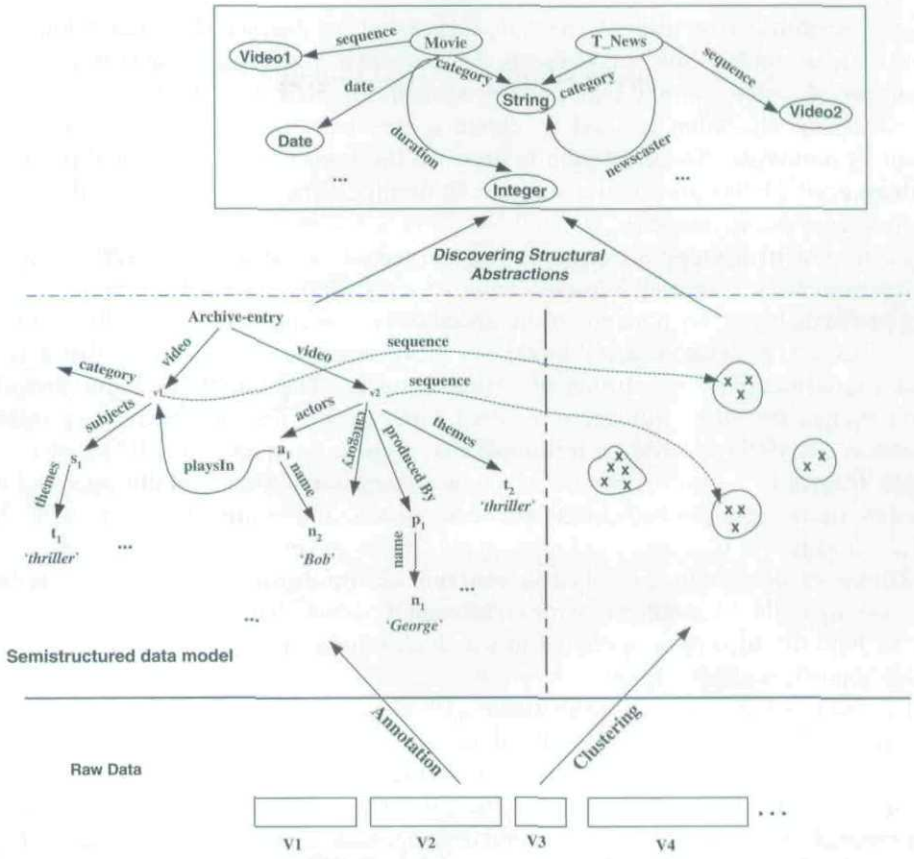

Figure 1. Framework for discovering structural associations.

2. Related work

In the framework of video data access and retrieval, our work relates to browsing. We shortly discuss relevant approaches regarding knowledge extraction and summarization for video browsing. This section is intended to be illustrative. We apologize if we have left out relevant references.

The authors of Hollfelder *et al.* (1998) proposed a retrieval engine for video browsing that offers conceptual, content-based access to videos. Using a small set of annotated stills as a training set, a statistical analysis associates patterns in feature vectors with conceptual classification topics. A video is segmented into single shots by a shot detection algorithm. The video stills indexing (analysis) system then employs a number of feature detection algorithms on selected frames. The result of these algorithms—called the feature extraction values—are used to find rules that map the values to conceptual terms. Generated rules are stored in a metadatabase.

The syntax and expressive power of the class of rules used are not presented. Additionally, the extraction process together with the use of these rules to guide evaluation of a query are not developed. In Meng and Chang (1996), the reflection

of the semantic structure of the video material for further data reduction and clustering of similar shots was the goal. A hierarchical key frame browser is used by a compressed video editing and parsing system to select the shots to be edited. A clustering algorithm is used to create a tree hierarchy of shots by grouping them by similarity. The clustering is done on the basis of visual information only. Arman *et al.* (1994) proposed a solution in which abstractions of each of the video sequences is pre-computed.

Each shot of a video sequence is represented at the abstraction level by using a representative frame (Rframe). Zhang *et al.* (1995) presented an approach to the problem based on parsing. More specifically, parsing will temporally segment and abstract a video source, based on key-frames selected during abstraction and spatial-temporal variations of visual features. The paper does not provide indications on the information used to guide the abstraction process. Kobla *et al.* (1997) described a technique that reduces a sequence of MPEG encoded video frames to a trail of points in a low dimensional space. In this space, they cluster frames, analyse transitions between clusters and compute properties of the resulting trail.

Zhong *et al.* (1996) described a generalized top-down hierarchical clustering process to build hierarchical representations of videos. This work has been done in the field of video data modeling in which defining objects and events in video is given importance.

Krishnamachari and Abdel-Mottaleb (1999) proposed a hierarchical clustering algorithm for images. Starting from a database with n images, the similarity between all pairs of images is pre-computed. The n images in the database are placed in n distinct clusters. These clusters are indexed by $\{C_1, \ldots, C_n\}$. Two clusters C_k and C_l are merged into a new class if the similarity between C_k and C_l is the largest. This process is repeated until only one cluster is left out. Oh and Hua (2000) proposed an approach for organizing and indexing video data. Each video is segmented into shots using a camera-tracking technique. This process allows to extract the feature vector for each shot. By using these shots, an automatic method is applied to build a browsing hierarchy. Finally, an index table is built. Zaiane *et al.* (1998) developed a multimedia data mining prototype, called MultiMediaMiner. The prototype includes the construction of multimedia cubes and the mining of knowledge such as summarization and classification of images and videos.

Zaïane *et al.* (2000) studied a set of methods for mining content-based associations with recurrent items and with spatial relationships from large visual data repositories. A progressive resolution refinement approach was proposed in which frequent item sets at rough resolution levels are mined. An interesting issue here is how to combine this approach and ours in order to deal with both visual and semantic layers of multimedia data mining.

Ordonez and Omiecinski (1999) proposed a data mining algorithm to find association rules in 2-dimensional colour images. The proposed algorithm rests on four steps: feature extraction, object identification, auxiliary image creation and object mining. Auxiliary domain knowledge is not used in this framework.

Faloutsos and Lin (1995) described a fast algorithm to map objects into points in some k-dimensional space (k is user-defined), such that the dis-similarities are preserved. Starting with a multi-dimensional scaling method from pattern recognition, the authors proposed a faster algorithm to solve the problem.

For building video summaries, most, if not all, of the proposed approaches make use of visual features only. No work has been done on the representation of structural associations in video data by combining visual features with semantic associations (By semantic associations, we mean the use of an annotation-based video model (here, a semistructured data model) to capture the logical relationships between the concepts described in the text annotations of the video data.). We take a new look at the problem of abstracting video contents and find that knowledge representation and reasoning techniques for concept languages developed in artificial intelligence combined with database techniques provide an interesting angle to attack such a problem.

3. Representing structural associations

3.1 *Introduction*

The representational formalism we are going to use for describing structural associations belongs to the family of description logics. Description logics (also called concept languages or terminological logics) (Brachman and Schmolze 1985, Nebel 1990, donini *et al.* 1994, Baader *et al.* 2003) are a family of logics designed to represent the taxonomic and conceptual knowledge of a particular application domain on an abstract, logical level. They are equipped with well-defined, set-theoretic semantics. Furthermore, the interesting reasoning problems such as subsumption and satisfiability are, for most description logics, effectively decidable. In the following, we introduce the description logic that will be used to represent the extracted structural associations.

3.2 *Language*

Expressions in this language are built from symbols taken from sets of *concept names* and *role names* and a number of constructors that permit the formation of *concept descriptions* (or simply *concepts*).

The set of constructors allowed by the language are given below.

Definition 1 (Syntax) *Let* $\mathcal{C}$ and $\mathcal{R}$ *be two pairwise disjoint sets of concept names and role names respectively. Let* $A \in \mathcal{C}$ *be a concept name and* $R \in \mathcal{R}$ *be a role name. Concepts* C, D *are defined inductively by the following rules:*

$$C, D \longrightarrow \top \mid \bot \mid A \mid C \sqcap D \mid \exists R.C$$

Here, the concept $\top$ (top) denotes the entire domain of an interpretation, while the concept $\bot$ (bottom) denotes the empty set of objects.

In the following, we call this language $\mathcal{LSA}$ (Language for Structural Associations).

Definition 2 (Semantics) *The semantics is given by an interpretation* $\mathcal{J} = (\Delta^{\mathcal{J}}, \cdot^{\mathcal{J}})$, *which consists of an (abstract) interpretation domain* $\Delta^{\mathcal{J}}$, *and an interpretation function* $\cdot^{\mathcal{J}}$, *which associates each concept* C *with a subset* $C^{\mathcal{J}}$ *of* $\Delta^{\mathcal{J}}$ *and each role* R *with a binary relation* $R^{\mathcal{J}}$ *on* $\Delta^{\mathcal{J}}$. *Additionally,* $\mathcal{J}$ *has to satisfy the following*

equations:

$$\top^{\mathcal{J}} = \Delta^{\mathcal{J}}$$
$$\bot^{\mathcal{J}} = \emptyset$$
$$(C \sqcap D)^{\mathcal{J}} = C^{\mathcal{J}} \cap D^{\mathcal{J}}$$
$$(\exists R.C)^{\mathcal{J}} = \left\{ d_1 \in \Delta^{\mathcal{J}} \mid \exists d_2 \colon (d_1, d_2) \in R^{\mathcal{J}} \wedge d_2 \in C^{\mathcal{J}} \right\}$$

3.3 *Knowledge bases*

A knowledge base $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ built by means of the description logic is formed by two components: a TBox $\mathcal{T}$, the *intensional* one, and an ABox $\mathcal{A}$, the *extensional* one.

Properties of the concepts of interest in a particular application are specified at the intensional level using the so-called *terminological axioms*. Let B and D be a concept name and a concept respectively. Then, $B = D$ is called *concept definition*, where D gives necessary and sufficient conditions for membership in B. A concept name not appearing in the left-hand side of any terminological axiom is called an *atomic concept*.

A TBox (or terminology) $\mathcal{T}$ is a (finite) set of terminological axioms.

An interpretation $\mathcal{J}$ satisfies the axiom $B = D$ iff $B^{\mathcal{J}} = D^{\mathcal{J}}$.

Definition 3 (Valid Interpretation) *An interpretation $\mathcal{J} = (\Delta^{\mathcal{J}}, \cdot^{\mathcal{J}})$ is a valid interpretation, also called a model, of a TBox $\mathcal{T}$ iff it satisfies every axiom in $\mathcal{T}$.*

We say that a concept C is *subsumed* by a concept D, written $C \sqsubseteq D$, iff $C^{\mathcal{J}} \subseteq D^{\mathcal{J}}$ for every valid interpretation $\mathcal{J}$, and C is *equivalent* to D, written $C \equiv D$, iff $C^{\mathcal{J}} = D^{\mathcal{J}}$ for every valid interpretation $\mathcal{J}$.

3.4 *Reasoning services*

In the following, we mainly resort to non standard inference services, namely the computation of the *least common subsumer* (lcs) of a set of descriptions and the *most specific concept* (msc) of an object.

A least common subsumer of a set of descriptions $D_1, \ldots, D_n \in \mathcal{LSA}$ is a most specific description (in the infinite space of all possible descriptions in $\mathcal{LSA}$) that subsumes all the D'_is.

Definition 4 (Least Common Subsumer) (Baader *et al.* 1999) *Let $C_1, \ldots, C_n$ and E be concept descriptions in $\mathcal{LSA}$. The concept description E is a least common subsumer of $C_1, \ldots, C_n$ (noted $E = lcs(C_1, \ldots, C_n)$) iff*

- *$C_i \sqsubseteq E \; \forall i \in [1, n]$, and*
- *E is the least concept description with this property, i.e. if E' is a concept description satisfying $C_i \sqsubseteq E' \; \forall i \in [1, n]$, then $E \sqsubseteq E'$*

The msc is a process that abstracts an object o by constructing a very specific description msc(o) that characterizes o.

Definition 5 (Most Specific Concept) *Let $\mathcal{A}$ be an ABox, o be an object in $\mathcal{A}$, and C be a concept description. C is the most specific concept for o in $\mathcal{A}$, noted $C = msc(o)$, iff $o \in C^{\mathcal{I}}$ and if C' is a concept description satisfying $o \in C'^{\mathcal{I}}$, then $C \sqsubseteq C'$.*

Techniques for computing least common subsumers and most specific concepts are described in Donini and Era (1992) and Baader and Küsters (1998).

Subsumption in $\mathcal{LSA}$ can be decided in polynomial time. Computing the *lcs* of two $\mathcal{LSA}$-descriptions C, D can be done in polynomial time and the size of the *lcs* is polynomial in the size of C and D (Baader *et al.* 1999).

4. Discovering structural associations

In the following, we propose two layers for representing video content (figure 2):

(1) *Feature & Content Layer*: it contains video visual features (e.g. colour, shape, motion). This layer is characterized by a set of techniques and algorithms allowing to retrieve video sequences based on the similarity of visual features.
(2) *Semantic Layer*: this layer contains objects of interest, their descriptions, and relationships among objects based on annotations or extracted features. It constitutes what we call the extensional part of a video database. Objects in a video sequence are represented in the Semantic Layer as visual entities. Instances of visual objects consist of conventional attributes (e.g. name, actorID, date, etc.).

Video data can be captured using a semistructured data model. This one describes data using a graph, called *data graph*, with objects as vertices and labels on the edges. Each object has a unique *identifier* from the type oid.

Definition 6 (Data Graph) *A data graph $DB = (V_a \cup V_c, E, r)$ is a labelled rooted graph, where V_a and V_c are disjoint sets of oid's corresponding respectively to atomic and complex objects. $V_a \cup V_c$ forms a finite set of nodes. $r \in V_c$ is a root node; $E \subseteq V_c \times L \times (V_a \cup V_c)$ is a set of labelled edges, where L is an infinite set of strings*

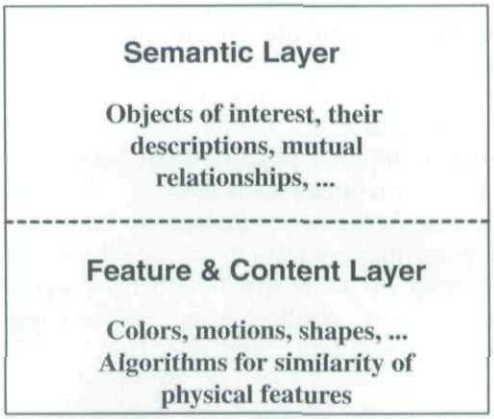

Figure 2. Two layers for video content.

indicating labels. (a, l, b) will denote an edge going from the node a to the node b and labelled l. We assume that all the nodes in $V_a \cup V_c$ are reachable from the root node.

Definition 7 (Simulation/Bisimulation) *Let $G_1=(V_1, E_1)$ and $G_2=(V_2, E_2)$ be two graphs, and L be a prescribed set of edge labels. A relation R on V_1, V_2 is a simulation if it satisfies:*

$$\forall l \in L, \quad \forall x_1, y_1 \in V_1, \quad \forall x_2 \in V_2 \ (x_1[l]y_1 \wedge x_1 R x_2 \Rightarrow \exists y_2 \in V_2 (x_2[l]y_2 \wedge y_1 R y_2)$$

$x[l]y$ stands for an l-labelled edge from x to y.

Let $G_1=(V_1, E_1, r_1)$ and $G_2=(V_2, E_2, r_2)$ be two graphs, where r_1 and r_2 are, respectively, the roots of G_1 and G_2. A relation R on V_1, V_2 is a simulation if it satisfies:

1. $r_1 R r_2$, *and*
2. $\forall x_1, y_1 \in V_1 \forall x_2 \in V_2 (x_1, l, y_1) \in E_1 \wedge x_1 R x_2 \Rightarrow \exists y_2 \in V_2 ((x_2, l, y_2) \in E_2 \wedge (y_1 R y_2))$.

A relation R is a bisimulation if it is a simulation and the reverse R^{-1} is also a simulation.

A simulation R is typed iff whenever xRy, if y is an atomic type, then x must be an atomic node too and have a value of that type.

Two nodes v_1 and v_2 are similar (respectively, bisimilar) if there exists a simulation (respectively, a bisimulation) R such that $v_1 R v_2$.

Informally, a simulation from a graph G_1 to a graph G_2 means that whenever there is an edge in G_1, there is a corresponding edge with the same label in G_2.

A data graph DB conforms to a schema graph S, in notation $DB \leq S$, if there exists a simulation from DB to S. The notion of conformance allows to define an ordering on graph schemas. Informally, a schema graph S is a *refinement* of a schema graph S' iff whenever a data graph DB conforms to S it conforms to S'. Refinement of schema graph can also be defined by means of simulation relation.

4.1 *Discovering structures*

A data graph presents video data as self-describing with no need for any *intentional* description (i.e. an *a priori* schema). In such a *minimalist* approach, the schema is missed even if it is partially and/or implicitly available. Very often, data is not fully unstructured and exploiting, even a partial (Application-dependent) structure, can be very beneficial, e.g. for browsing and querying the data, optimizing query evaluation or improving storage.

We consider the problem of discovering *typical* structural information of video data. This problem, referred to as *schema discovery*, can be formulated as follows: given a data graph DB, find the corresponding TBox (and hence the $\mathcal{LSA}$ graph) that describes the common substructures within DB.

In this section, we propose a *bottom up approach* to deal with the schema discovery problem. Since a data graph may conform to several schema graphs, we first consider the problem of discovering the most specific schema graph of a data graph DB. Then, we show how the most specific schema can be compressed to obtain an approximate typing of the input data graph.

4.1.1 Extracting the most specific schema. We propose an algorithm, called Gen$\mathcal{K}$, that allows one to *abstract* objects of a data graph. Abstraction is a well known

mechanism in reasoning about individuals in description logic systems. It consists of retrieving all the assertions characterizing a given object o and collecting them into a single concept. Such a concept has the property of being the most specific concept expressible in $\mathcal{LSA}$ of which the object o is an instance.

Algorithm 1 (Gen$\mathcal{K}$)
Require: a data graph $DB = (V_a \cup V_c, E, L, v, r)$
Ensure: a knowledge base $\mathcal{K}=(\mathcal{T}_S, \mathcal{A})$.
//**Step 1**: Generating $\mathcal{A}$
1: **for** each $(o_i, l, o_j) \in E$ **do**
2: $\mathcal{A} \leftarrow \mathcal{A} \cup \{l(o_i, o_j)\}$
3: **end for**
//**Step 2**: Generating $\mathcal{T}_S$
//Initialization of specific descriptions and atomic types
4: **for** each o_i in V_a **do**
5: //If an object o_j is atomic we denotes by $type(o_j)$ its atomic type
6: $N_{o_j} \leftarrow type(o_j)$
7: **end for**
8: **for** each o_i in V_c **do**
9: $\delta_{o_i} \leftarrow \emptyset$
10: **end for**
//Computation of the exact structure for each object
11: **for** each $l(o_i, o_j) \in \mathcal{A}$ **do**
12: $\delta_{o_i} \leftarrow \delta_{o_i} \sqcap \exists l.N_{o_j}$
13: **end for**
14: **for** each generated δ_{o_i} **do**
15: $\mathcal{T}_S \leftarrow \mathcal{T}_S \cup \{N_{o_i} = \delta_{o_i}\}$
16: **end for**
//Step 3: Merging equivalent concepts
17: Merge together equivalent concepts in $\mathcal{T}_S$
18: **return** $\mathcal{K} = (\mathcal{T}_S, \mathcal{A})$

The algorithm Gen$\mathcal{K}$ works as follows:

- *Step 1*: it maps a data graph DB to an ABox $\mathcal{A}$. We view a database, and hence the Semantic Layer, as a knowledge base $\mathcal{K} = (\emptyset, \mathcal{A})$ with an empty TBox. The translation of a data graph into an ABox is straightforward.
- *Step 2*: a TBox, called $\mathcal{T}_S$, is derived. For each object o occurring in $\mathcal{A}$, the TBox $\mathcal{T}_S$ contains an axiom of the form $N_o = \delta_o$, where N_o is a concept name and δ_o is a concept term corresponding to $msc(o)$. This assertion reflects the knowledge in the ABox $\mathcal{A}$ concerning the object o. By doing so, we record in a single concept all the information about each object occurring in $\mathcal{A}$ as it can be extracted from all the assertions of the ABox.
- *Step 3*: It merges together equivalent concepts in $\mathcal{T}_S$.

Note that the test of equivalence between two concepts can be reduced to a subsumption test ($C \equiv D$ iff $C \sqsubseteq D$ and $D \sqsubseteq C$). Therefore, *Step 3* can be achieved by first applying the classification process on the concepts of $\mathcal{T}_S$, leading to explicit subsumption relationships between concepts, and then equivalent concepts are identified and merged together.

Proposition 1 (Correctness) *Let $\mathcal{K} = (\mathcal{T}_S, \mathcal{A})$ be the knowledge base generated by applying the algorithm Gen$\mathcal{K}$ on a data graph $DB = (V_a \cup V_c, E, r)$, and $\mathcal{G}_{\mathcal{T}_S} = (\mathcal{V}_{\mathcal{T}_S}, \mathcal{E}_{\mathcal{T}_S})$ be the description graph corresponding to $\mathcal{T}_S$.*

(1) *The data graph DB conforms to $\mathcal{G}_{\mathcal{T}_S}$.*
(2) *The $\mathcal{EL}$ graph $\mathcal{G}_{\mathcal{T}_S} = (\mathcal{V}_{\mathcal{T}_S}, \mathcal{E}_{\mathcal{T}_S})$ is the most specific dual schema graph of DB.*

Proof 1 Let $N_r = \delta_r$ be the most specific concept associated to the root r (N_r is generated by the algorithm Gen$\mathcal{K}$) and let $\mathcal{J}$ be a valid interpretation of $\mathcal{K}$ which associates each object occurring in $\mathcal{A}$ with its most specific concept in $\mathcal{T}_S$.[†]

We define a relation R from the nodes of DB to those of $\mathcal{G}_{\mathcal{T}_S}$ such that:

$\forall o \in V_a \cup V_c, \forall G_N \in \mathcal{V}_{\mathcal{T}_S}, oRG_N$ iff $o \in N^{\mathcal{J}}$.

Let R^{-1} be the inverse relation of R (i.e., $G_N R^{-1} o$ iff oRG_N).

(1) Since $\mathcal{G}_{\mathcal{T}_S}$ is a dual schema graph (because it lists all required edges rather than allowed edges,) we must show that $\mathcal{G}_{\mathcal{T}_S} \leq \mathcal{DB}$.
We have to show that R^{-1} is a simulation, and thus $\mathcal{G}_{\mathcal{T}_S} \leq DB$.
$\forall N_i, N_j \in \mathcal{C}, \forall o_i \in \mathcal{O}$,

- $\forall G_{N_i}, G_{N_j} \in \mathcal{V}_{\mathcal{T}_S}$, if $(G_{N_i}, l, G_{N_j}) \in \mathcal{E}_{\mathcal{T}_S}$ then the description of N_i contains the constraint $\exists l.N_j$. Hence, $\forall o_i \in N_i^{\mathcal{J}}, \exists o_j \in N_j^{\mathcal{J}}$ such that $(o_i, o_j) \in l^{\mathcal{J}}$ (i.e., $l(o_i, o_j) \in \mathcal{A}$. (i)
- If o_i and G_{N_i} are in R-relation then $o_i \in N_i^{\mathcal{J}}$. (ii)

(i) and (ii) lead to

$\forall G_{N_i}, G_{N_j} \in \mathcal{V}_{\mathcal{T}_S}, \forall o_i \in V_a \cup V_c$ if $(G_{N_i}, l, G_{N_j}) \in \mathcal{E}_{\mathcal{T}_S}$ and $o_i R G_{N_i}$ then $\exists o_j \in V_a \cup V_c$ such that $l(o_i, o_j) \in E$ and $o_j R G_{N_j}$. (iii)

In the same way, we can show that:

$\forall o_i, o_j \in V_a \cup V_c, \forall G_{N_i} \in \mathcal{V}_{\mathcal{T}_S}$ if $(o_i, l, o_j) \in E$ and $G_{N_i} R^{-1} o_i$ then $\exists G_{N_j} \in \mathcal{V}_{\mathcal{T}_S}$ such that $(G_{N_i}, l, G_{N_j}) \in \mathcal{E}_{\mathcal{T}_S}$ and $G_{N_j} R^{-1} o_j$ (since $N_i = msc(o_i)$). (iv)

Let us now show that $\mathcal{G}_{\mathcal{T}_S}$ is a rooted graph and its root is in R-relation with the root r of DB.

r is a root of DB implies that each object in DB is reachable from r. (v)

(iv) and (v) implies that each node in $G_{\mathcal{T}_S}$ is reachable from the node G_{N_r}. Hence, G_{N_r} is a root of $\mathcal{G}_{\mathcal{T}_S}$. (vi)

r and G_{N_r} are in R-relation since $r \in N_r^{\mathcal{J}}$. (vii)

(iv), (vi) and (vii) implies that R^{-1} is a simulation from DB to $\mathcal{G}_{\mathcal{T}_S}$. Thus, $\mathcal{G}_{\mathcal{T}_S} \leq DB$.

(2) $\mathcal{G}_{\mathcal{T}_S} = (\mathcal{V}_{\mathcal{T}_S}, \mathcal{E}_{\mathcal{T}_S})$ is the most specific dual schema graph of DB.

(iii), (vi) and (vii) implies that R is a simulation[‡] from $\mathcal{G}_{\mathcal{T}_S}$ to DB. Thus, $DB \leq \mathcal{G}_{\mathcal{T}_S}$.

[†] The interpretation $\mathcal{J}$ can be computed using the well known *relization* reasoning service provided by description logic systems.

[‡] Hence, R is a bisimulation between DB and $\mathcal{G}_{\mathcal{T}_S}$.

The transitivity of simulation ensures that whenever S' is a dual schema graph such that DB conforms to S' (i.e., $S' \leq DB$), then $\mathcal{S}' \leq \mathcal{G}_{\mathcal{T}_S}$. Hence, $\mathcal{G}_{\mathcal{T}_S}$ is the most specific schema graph of DB.

4.2 *Approximate typing*

In the previous section, we provided an algorithm for extracting the most specific schema abstracting a data graph. The data graph conforms to the schema specified as a set of terminological axioms forming a TBox. The resulting TBox exactly matches the data graph (each object is associated with a concept that describes it precisely). Although, the number of concepts (in other words the number of axioms in the TBox) is reduced by grouping together equivalent concepts, the most specific schema may remain too large.

In general, we do not expect to find strict regular structures in video data. Indeed, in a data graph many of the most interesting structures show up in a slightly different forms. So, we can be often satisfied by a compact graph schema that roughly describes the input data graph.

In the following, we propose an approach for reducing the size of the most specific schema by grouping together the concepts that have similar structures. To do that, we replace sets of concept descriptions by their least common subsumer in the TBox.

4.2.1 Computing common subsumers. Given a data graph DB and a threshold k, Comp$\mathcal{T}$ computes frequent substructures in DB. This algorithm is based on the following proposition.

Proposition 2. *Let $DB = (V_a \cup V_c, E, r)$ be a data graph. A description D is verified by k objects in DB iff $D \sqsubseteq lcs(msc(o_{i_1}), \ldots, msc(o_{i_k}))$, for some o_{i_j} in V_c, $j \in [1, k]$.*

So far, the most specific schema (i.e. the content of a TBox) of a data graph is composed of most specific concepts. Therefore, given a most specific schema $\mathcal{T}_S$ of a data graph DB and a threshold k, we can infer the frequent descriptions by computing the least common subsumers of each k concepts in $\mathcal{T}_S$. The computation terminates since $\mathcal{T}_S$ is finite.

4.2.2 The algorithm Comp$\mathcal{T}$. Let $\mathcal{T}$ be a given TBox. We note by $\mathcal{C}_\mathcal{T} = \{N_1, \ldots, N_n\}$ the set of all concept names occurring in $\mathcal{T}$, and we note by $Candidate_i(\mathcal{C}_\mathcal{T})$ the set of subsets of $\mathcal{C}_\mathcal{T}$ of cardinality i (i.e. $Candidate_i\ (\mathcal{C}_\mathcal{T}) = \{C \mid C \subseteq \mathcal{C}_\mathcal{T}$ and $\sharp C = i\}$, where the symbol $\sharp$ stands for cardinality). For a non-empty subset $C = \{N_1, \ldots, N_m\}$ of $\mathcal{C}_\mathcal{T}$, we denote by $lcs[C]$ the concept description computed by $lcs(N_1, \ldots, N_m)$ and we denote by N_C the concept name associated with this description.

In the following, Gen$\mathcal{K}'$ stands for a slightly modified version of the algorithm Gen$\mathcal{K}$ from which we remove Step 3, which consists in merging together equivalent concepts. Gen$\mathcal{K}'$ is obtained by removing line 17 from the algorithm Gen$\mathcal{K}$.

Algorithm 2 (Comp$\mathcal{T}$)
Require: a data graph DB, *a threshold* k
Ensure: a knowledge base $\mathcal{K} = (\mathcal{T}, \mathcal{A})$
1: Gen$\mathcal{K}'(DB)$
2: Let $\mathcal{K} = (\mathcal{T}_S, \mathcal{A})$ be the knowledge base generated by Gen$\mathcal{K}'$.

3: Compute $Candidate_k(C_{T_S})$
4: **for all** $C \in Candidate_k(C_{T_S})$ **do**
5: Compute $lcs[C]$
6: **if** $lcs[C] \not\equiv \top$ **then**
7: $\mathcal{T} \leftarrow \mathcal{T} \cup \{N_C = lcs[C]\}$
8: **end if**
9: **end for**
10: Merge equivalent concepts in $\mathcal{T}$
11: Cleaning($\mathcal{T}$)
12: **return** $\mathcal{K} = (\mathcal{T}, \mathcal{A})$

The input of the algorithm Comp$\mathcal{T}$ is a data graph DB and a threshold k. The first step consists in computing (by a call to Gen$\mathcal{K}'$) the knowledge base $\mathcal{K} = (\mathcal{T}_S, \mathcal{A})$, where $\mathcal{T}_S$ is the most specific schema abstracting the data graph DB. Given this most specific schema and corresponding $\mathcal{C}_{\mathcal{T}_S}$ a most specific TBox $\mathcal{T}$ containing least common subsumers is derived. To reduce the number of concepts in $\mathcal{T}$, equivalent concepts are merged together.

Finally, a supervised cleaning process can be used for removing uninteresting concepts.

Let us briefly describe an interactive cleaning process for removing '*uninteresting*' concepts from $\mathcal{T}$. This process is based on a simple idea that consists in assigning each object to its most specific description in $\mathcal{T}$, and then identifying concepts with less than k instances.

The cleaning step consists in two main tasks:

(1) We assign each object in $\mathcal{A}$ to its most specific concept in $\mathcal{T}$. This is called *realization*. We denote by $\mathcal{J}_{\mathcal{R}}$ the resulting interpretation. The realization ensures that if an object o belongs to a concept named N, then N describes at least a part of o which is not described elsewhere in $\mathcal{T}$. Let $n = \sharp N^{\mathcal{J}_{\mathcal{R}}}$ (i.e. the cardinality of N given $\mathcal{J}_{\mathcal{R}}$). Hence, the *differential* knowledge conveyed by N concerns n objects.
(2) We mark '*Removable*' each concept name N in $\mathcal{T}$ such that $\sharp N^{\mathcal{J}_{\mathcal{R}}} \leq k$. After that, an expert user may decide which concepts have to be removed.

5. Conclusion

Browsing and retrieval require that the source material first be effectively *indexed*. While we tend to think of indexing supporting retrieval, *browsing* is equally significant for video source material. By 'browsing' we mean an informal perusal of content that may lack any specific goal of focus. Since a video database can have more than one type, we considered first the extraction of the schema that best describes the data of the Semantic Layer. This schema is the *most specific* of the input instance and corresponds to perfect typing. For each object in the Semantic Layer, we computed a single type description reflecting relevant assertions about the object in the Semantic Layer. The computed description is the *most specific concept*, expressible in our type description language, of which the object is an instance. The result of this step is a most specific schema abstracting the Semantic Layer content. Second, due to the irregularity of structures in video databases, perfect

typing may lead to large sets of descriptions. We have shown that it is possible to identify similar, and not only identical, objects, and provide an approximate typing. For that, by computing least common subsumers of some appropriate descriptions in the most specific schema generated in the previous step, we built a compressed version of this most specific schema.

Due to the visual nature of video data, the proposed framework should be extended to take into account the visual features. Hybrid representation frameworks can provide a nice basis for such an extension. Cyclic graphs are a natural structure for a full representation of the semantic layer of video data. Dealing with cyclic structures in schema discovery will be an important extension. The preliminary investigations reported in Baader (2003, 2004) can serve as a starting point.

References

F. Arman, R. Depommier, A. Hsu and M.-Y. Chiu, "Content-based browsing of video sequences", in *Proceedings of the Second ACM International Conference on Multimedia*, 1994, pp. 97–103

F. Baader, "The instance problem and the most specific concept in the description logic el w.r.t. terminological cycles with descriptive semantics", Research Report 03-01, Dresden University of Technology, http://lat.inf.tu-dresden.de, 2003.

F. Baader, "A graph-theoretic generalization of the least common subsumer and the most specific concept in the description logic el", Research Report 04-02, Dresden University of Technology, http://lat.inf.tu-dresden.de, 2004.

F. Baader, D. Calvanese, D.L. McGuinness, D. Nardi and P.F. Patel-Schneider, *The Description Logic Handdbook*, Cambridge: Cambridge University Press, 2003.

F. Baader and R. Küsters, "Computing the least common subsumer and the most specific concept in the presence of cyclic $\mathcal{ALN}$-concept descriptions", in *Proceedings of KI-98*, volume 1504 of *LNCS*, 1998, pp. 129–140.

F. Baader, R. Küsters and R. Molitor, "Computing least common subsumer in description logics with existential restrictions", in *Proceedings of IJCAI'99*, T. Dean, Ed., 1999, pp. 96–101.

R. Brachman and J. Schmolze, "An overview of the KL-ONE knowledge representation system", *Cognitive Science*, 1985, 9, pp. 171–216.

F. Donini and A. Era, "Most specific concepts technique for knowledge bases with incomplete information", in *Proceedings of the 1992 International Conference on Information and Knowledge Management (CIKM)*, 1992, pp. 545–551.

F. Donini, M. Lenzerini, D. Nardi and A. Schaerf, "Deduction in concept languages: from subsumption to instance checking", *Journal of Logic and Computation*, 1994, 4, pp. 423–452.

C. Faloutsos and K.-I. Lin, "FastMap: a fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets", in *Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data*, M.J. Carey and D.A. Schneider, Eds, 1995, pp. 163–174.

M.-S. Hacid, C. Decleir and J. Kouloumdjian, "A database approach for modeling and querying video data", *IEEE Transactions on Knowledge and Data Engineering (IEEE TKDE)*, 12, 729–750, 2000.

S. Hollfelder, A. Everts and U. Thiel, "Concept-based browsing in video libraries", in *Proceedings of the IEEE Forum on Research and Technology of Advances in Digital Libraries*, 1998.

V. Kobla, D. Doermann, K. Lin and C. Faloutsos, "Compressed domain video indexing techniques using dct and motion vector information in MPEG video", in *Proceedings of the SPIE Conference on Storage and Retrieval for Still Image and Video Database V*, volume 3022, 1999, pp. 200–211.

S. Krishnamachari and M. Abdel-Mottaleb, "Image browsing using hierarchical clustering", in *Proceedings of The Fourth IEEE Symposium on Computers and Communications*, 1999.

J. Meng and S.-F. Chang, " CVEPS—a conceptual video editing and parsing system", in *Proceedings of the ACM Multimedia*, 1996, pp. 43–53.

B. Nebel, "Reasoning and revision in hybrid representation systems", volume 422 of *LNCS*, Neq York: Springer Verlag, 1990 p. 300.

J. Oh and K.A. Hua, "Efficient and cost-effective techniques for browsing and indexing large video databases", in *Proceedings of the ACM Sigmod International Conference on Management of Data (SIGMOD'2000)*, 2000, pp. 415–426.

C. Ordonez and E. Omiecinski, "Discovering association rules based on image content", in *Proceedings of the IEEE Forum on Research and Technology Advances in Digital Libraries*, I.C. Society, Ed., 1999, pp. 38–49.

O.R. Zaïane, J. Han, Z.-N. Li and J. Hou, "Mining multimedia data.", in *Proceedings of CASCON'98: Meeting of Minds*, November 1998.

O.R. Zaïane, J. Han and H. Zhu, "Mining recurrent items in multimedia with progressive resolution refinement", in *Proceedings of the 16th International Conference on Data Engineering*, I. C. Society, Ed., 2000, pp. 461–470.

H. Zhang, C.Y. Low, S.W. Smoliar and J.H. Wu, "Video parsing, retrieval and browsing: an integrated and content-based solution", in *Proceedings of the ACM International Conference on Multimedia*, 1995, pp. 15–24.

D. Zhong, H. Zhang and S.F. Chang, "Clustering methods for video browsing and annotation", in *Proceedings of the SPIE Conference on Storage and Retrieval for Still Image and Video Database IV*, volume 2670, 1996, pp. 239–246.