INTRODUCTION

Multimedia databases have been used in many application fields. As opposed to traditional alphanumeric databases, they need enhanced data models and DBMSs to enable the modeling and management of complex data types. After an initial anarchy, multimedia DBMSs (MMDBMS) have been classified based on standard issues, such as the supported data model, the indexing techniques to support content-based retrieval, the query language, the support for distributed multimedia information management, and the flexibility of their architecture (Narasimhalu, 1996).

A conspicuous number of MMDBMS products have been developed. Examples include CORE (Wu, Mehtre, Lam, & Gao, 1995), OVID (Oomoto & Tanaka, 1993), VODAK (Löhr & Rakow, 1995), QBIC (Flickner et al., 1995), ATLAS (Sacks-Davis, Ramamohanarao, Thom, & Zobel, 1995), each providing enhanced support for one or more media domains among text, sound, image, and video. Some of these products support specific data models, whereas others support the object-oriented data model or even the canonical relational data model. Moreover, extensible relational DBMSs have been introduced to extend relational DBMSs with objec-oriented features, such as the capability to manage complex data types, including multimedia data. In particular, they implement the concept of the object-relational universal server, providing means to enable the construction of user defined data types (UDTs), and functions for manipulating them (UDFs). In addition, SQL3 has become the standard for relational DBMSs extended with object-oriented capabilities. The standard includes UDTs, UDFs, LOBs (a variant of BLOBs), and type checking on user-defined data types, which are accessed through SQL statements. Early examples of extensible RDBMSs include Postgres, IBM/DB2 version 5, Informix, and ORACLE8.

As MMDBMSs technology has become more mature, the research community has been seeking new methodologies for multimedia software engineering. Independently from the data model underlying the chosen MMDBMS, multimedia software engineering methodologies should include techniques for database design, embedding guidelines and normal forms to prevent anomalies that might arise while manipulating multimedia data.

In this paper, we describe a general-purpose framework to define normal forms in multimedia databases. The framework applies in a seamless way to images as well as to all the other different media types. The semantics of multimedia attributes is defined by means of generalized icons (Chang, 1996), previously used to model multimedia languages in a visual language fashion. In particular, generalized icons are used here to derive extended functional dependencies, which are parameterised upon the similarity measure used to compare multimedia data (Santini & Jain, 1999). Based on these new dependencies, we define three normal forms aiming to reach a suitable partitioning of multimedia data and to derive database schemes that prevent possible manipulation anomalies.

BACKGROUND

The normalization of multimedia databases needs to account for many new issues as opposed to alphanumeric databases. Many different types of complex data need to be analysed. However, in the literature we find many database design techniques focusing on image databases. In particular, a technique for normalizing image databases focuses on the partitioning of images so as to enhance image search and retrieval (Santini & Gupta, 2002). To this end, the technique aims to define dependencies among image features, which suggest to the designer how to efficiently map them into a database schema.

Database designers use their understanding of the semantics of attributes to specify functional dependencies among them (Elmasri & Navathe, 2003). As we know, other than alphanumeric data, multimedia databases can
store complex data types, such as text, sound, image, and video, which were initially modelled as binary large objects (BLOBs) in early MMDBMSs.

In the following we introduce a framework to model the semantics of multimedia attributes aiming to derive functional dependencies between them. To this end, we have exploited the framework of generalized icons (Chang, 1996). Generalized icons are dual objects \((x_m, x_l)\), with a logical part \(x_m\) and a physical part \(x_l\). They can be used to describe multimedia objects such as images, sounds, texts, motions, and videos. A generalized icon for modeling images is like a traditional icon, whereas those for modeling sounds, texts, motions, and videos are earcons, ticons, micons, and vicons, respectively. For all of them we denote the logical part with \(x_m\), whereas the physical parts will be denoted with \(x_l\) for icons, \(x_f\) for earcons, \(x_t\) for ticons, \(x_x\) for micons, and \(x_v\) for vicons. The logical part \(x_m\) always describes semantics, whereas \(x_l\) represents an image, \(x_f\) a sound, \(x_t\) a text, \(x_m\) a motion, and \(x_v\) a video. Furthermore, a multicon is a generalized icon representing composite multimedia objects (Arndt, Cafiero, & Guercio, 1997). Generalized icons can be combined by means of special icon operators. The latter are dual objects themselves, wherein the logical part is used to combine the logical parts \(x_l\) of the operand icons, whereas the physical part is used to combine their physical parts \(x_v\). For instance, by applying a temporal operator to several icons and an earcon, we might obtain a vicon, with the physical part representing a video, and the logical part describing the video semantics.

In our framework we associate a generalized icon to each complex attribute, using the logical part to describe its semantics and the physical part to describe the physical appearance based on a given storage strategy.

The logical parts of generalized icons will have to be expressed through a semantic model. Conceptual graphs are an example of a semantic model that can be used to describe logical parts of generalized icons (Chang, 1996). Alternatively, the designer can use frames, semantic networks, or visual CD forms (Chang, Polese, Orefice, & Tucci, 1994). As an example, choosing a frame-based representation, an image icon representing the face of a person may be described by a frame with attributes describing the name of the person, the colors of the picture, objects appearing in it, including their spatial relationships. A vicon will contain semantic attributes describing the images of the video photograms, the title of the video, the topic, the duration, the temporal relationships, and so forth.

Based on the specific domain of the multimedia database being constructed, the designer will have to specify the semantics of simple and complex attributes according to the chosen semantic model. Once he or she has accomplished this task, the generalized icons for the multimedia database are completely specified, which provides a semantic specification of the tuples in the database.

As an example, to describe semantics in a database of singers we might use the attributes name, birth date, and genre as alphanumeric attributes; picture as an icon representing the singer’s picture; one or more earcons to represent some of the singer’s songs; and one or more vicons to represent the singer’s video clips. A tuple in this database might describe information about a specific singer, including his or her songs and video clips. This provides a complete semantic specification of the tuple.

**NORMAL FORMS IN MULTIMEDIA DATABASES**

In traditional relational databases, a functional dependency is defined as a constraint between two sets of attributes from the database. Given two sets of attributes \(X\) and \(Y\), a functional dependency between them is denoted by \(X \rightarrow Y\). The constraint says that, for any two tuples \(t_1\) and \(t_2\) having \(t_1[X] = t_2[X]\), then \(t_1[Y] = t_2[Y]\). This concept cannot be immediately applied to multimedia databases because there are no similar simple, efficient methods to compare multimedia attributes. In other words, we need a method for defining equalities between groups of attributes involving complex data types.

Generally speaking, the matching of complex attributes needs to be based on an approximate match paradigm, such as those used in content-based retrieval from multimedia databases (Schaubl, 1997). In particular, we extend the definition of functional dependency by selecting a specific similarity function and thresholds to perform approximate comparisons of complex data types. Thus, the functional dependencies change if we use different similarity functions. As a consequence, we enrich the notation used for functional dependencies to include symbols representing the chosen similarity function. In what follows, we introduce some basic concepts of similarity theory (Santini & Jain, 1999).

Tuples of a relation can be compared by means of a set of relevant features \(\Phi\). For instance, images can be compared using attributes such as color, texture, and shape; audio data can be compared using loudness, pitch, brightness, bandwidth, and harmonicity. The values of each feature \(F \in \Phi\) belong to a domain \(D = \text{dom}(F)\).

The similarity between two elements \(x\) and \(y\) in a tuple is based on distance measures in feature spaces (that are assumed to be metric spaces) or, equivalently, on similarity functions. In the following, we will always refer to distance functions, but it should be understood that the same considerations apply to similarity functions, given the symmetry between distance and similarity functions.
3 more pages are available in the full version of this document, which may be purchased using the "Add to Cart" button on the product's webpage:

[www.igi-global.com/chapter/normalizing-multimedia-databases/11181?camid=4v1](www.igi-global.com/chapter/normalizing-multimedia-databases/11181?camid=4v1)


[www.igi-global.com/e-resources/library-recommendation/?id=1](www.igi-global.com/e-resources/library-recommendation/?id=1)

Related Content

Processing Ill-defined Spatial Operators in Geographic Databases: A Common-sense Approach
[www.igi-global.com/article/processing-ill-defined-spatial-operators/51210?camid=4v1a](www.igi-global.com/article/processing-ill-defined-spatial-operators/51210?camid=4v1a)

A Scalable Algorithm for One-to-One, Onto, and Partial Schema Matching with Uninterpreted Column Names and Column Values
Boris Rabinovich and Mark Last (2014). *Journal of Database Management* (pp. 1-16).

Integrity Checking and Maintenance in Relational and Deductive Database and Beyond
[www.igi-global.com/chapter/integrity-checking-maintenance-relational-deductive/24236?camid=4v1a](www.igi-global.com/chapter/integrity-checking-maintenance-relational-deductive/24236?camid=4v1a)

Temporal Data Management and Processing with Column Oriented NoSQL Databases