Enhancing Multimedia Data Fragmentation

Richard Chbeir[1], Dominique Laurent[2]
[1] LE2I - Univiversity of Bourgogne - CNRS
France
[2] ETIS - ENSEA - University of Cergy-Pontoise - CNRS
France
richard.chbeir@u-bourgogne.fr, dominique.laurent@u-cergy.fr

ABSTRACT: *Distributed multimedia applications have emerged at an increasing rate during the last decade in several domains (video conferencing, e-health, virtual meeting rooms, etc). This has created several new challenging problems related to data integration and fragmentation, user-oriented and adaptive interfaces, real time and network performances, etc. In this article, we focus on the problem of data(base) fragmentation, initially consisting of reducing irrelevant data accesses by grouping data frequently accessed together in dedicated segments, in a multimedia context. We mainly address the issue of query and predicate implication required in current fragmentation algorithms, and provide a formal approach to identify such implications, in order to partition multimedia data efficiently. It is worthy to note that our approach is capable of considering multimedia-based as well as semantic comparisons, based on a generalized notion of functional dependencies, which are called multimedia functional dependencies.*

Keywords: Multimedia Fragmentation, Pre-ordering, Data Partition, Query Implication, Multimedia Distance, Multimedia Functional Dependency.

Received: 11 March 2010, Revised 12 April 2010, Accepted 18 April 2010

1. Introduction

In the past few years, distributed multimedia applications and data have become available at an increasing rate ranging from video-on-demand and video conferencing, to medical imaging, cartography, meteorology, surveillance, etc. This has created several new challenging problems related to data integration and fragmentation, user-oriented and adaptive interfaces, real time and network performances, etc.

While several studies aim at building distributed MultiMedia DataBase Management Systems (MMDBMS) [Braunmuller et al. 2000], most of existing systems lack an appropriate framework to adequately provide full-fledge multimedia operations. Particularly, data fragmentation (or partitioning) techniques need to be adapted in a multimedia context so to properly achieve high resource utilization and increased concurrency and parallelism.

We recall that fragmentation consists of dividing the database objects and/or entities into fragments, on the basis of common query accesses, in order to distribute them over several distant sites [Chinchwadkar and Goh 1999] so to:

— Reduce the amount of irrelevant data accessed by applications, because applications usually access portions of entities and objects.
— Allow parallel execution of a single query, dividing it into a set of sub-queries that operate on segments of an entity/class.

— Reduce the quantity of data transferred when migration is required.

— Decrease data update cost and storage space.

Though fragmentation of traditional (relational or object oriented) databases has been thoroughly studied in the literature [Özsu and Valduriez 1999; Baiao and Mattoso 1998; Ezeife and Barker 1998; Navathe et al. 1995; Bellatreche et al. 1997], fragmentation of multimedia data has not yet received strong attention [Saad et al. 2006; Getahun et al. 2007a; 2007b], mainly because of the following two issues:

— Multimedia data structure: while current fragmentation algorithms require as input the database conceptual schema [Chinchwadkar and Goh 1999], this requirement is not always fulfilled in multimedia databases due to the unstructured and complex nature of multimedia data.

— Multimedia features: the wide variety of features used to describe multimedia data is out of the scope of traditional partitioning techniques. These features can be mainly categorized as low-level features (such as color, texture, shape, layout, etc.) and semantic-based or meta-data features (such as event, main topic, place, time, etc).

In this article, we provide a formal approach dedicated to multimedia query and predicate implication, required in current fragmentation algorithms. To do so, we consider both multimedia and semantic features, and we define a pre-ordering between queries, making an explicit use of a generalization of functional dependencies, which we call multimedia functional dependencies. Based on this preordering, given a set of frequently asked queries (FQ), we characterize the set of queries whose answers constitute a minimal data set that has to be stored so as to optimize the computation of the answers to queries that are comparable to some queries in FQ.

The rest of the article is organized as follows. The next section is dedicated to provide a motivating scenario explaining the requirements to be considered when fragmenting multimedia data. In Section 3, we review related work concerning multimedia data fragmentation. In Section 4, we recall the main concepts and formalism necessary to understand our approach and we define a new type of multimedia data dependencies, called multimedia functional dependencies. Then, in Section 5, we define the type of multimedia queries considered in our approach, which are basically projection-selection queries taking into account the specific features of multimedia data, and then, we study the problem of implication of selections. In Section 6, we define a pre-ordering for multimedia queries, and the induced equivalence relation. In Section 7, we introduce our data fragmentation strategy and discuss its main implementation issues. Section 8 concludes the article and draws some of our future work.

2. Motivating Scenario

To illustrate and motivate our study throughout the article, we consider the following scenario of a simple multimedia database used to manage singer records in a production company.

Example 1. We consider a table named Albums defined over the attribute set U = {name, birth, place, genre, picture, song, clip}, where each tuple describes information about a specific song by a singer.

More precisely, attributes name, birth, place and picture refer respectively to the name, birth date, birth place and pictures of a specific singer, genre refers to the type of songs of this singer, whereas song and clip stand respectively for the title and the associated clip of a given song.

In this context, let us consider the following queries:

— Q_1 : find all songs of hiphop singers

— Q_2 : find all songs of popular singers

— Q_3 : find the singer pictures appearing in Figure 1

— Q_4 : find all singer pictures of singers appearing in Figure 2

— Q_5 : find all clips and pictures of singers from Paris

— Q_6 : find all clips and pictures of all French singers

With current fragmentation approaches, these queries are considered different and analyzed separately. However, they embed several implications:

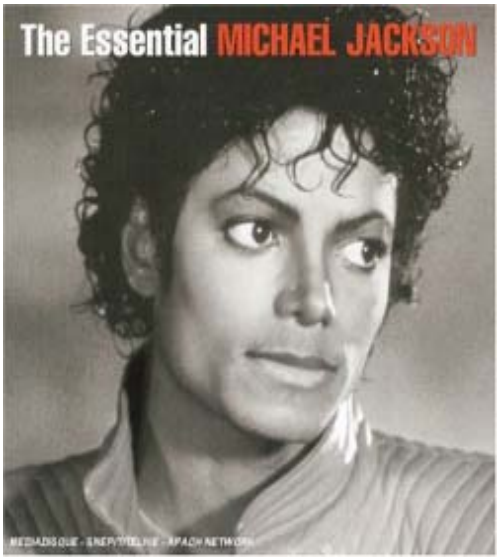

Figure 1. Sample photo of M. Jackson

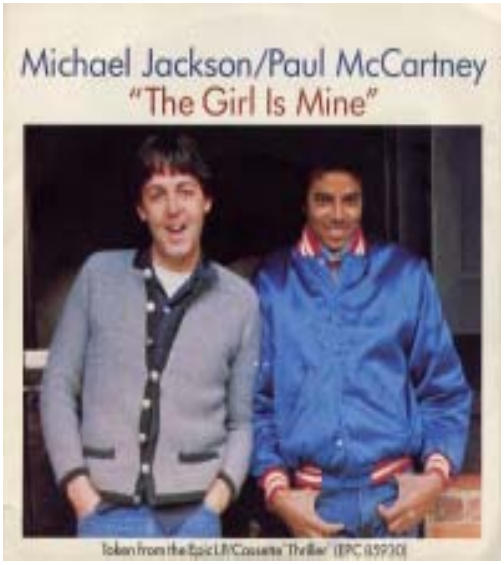

Figure 2. Sample photo of M. Jackson and P. McCartney

—The result of Q_1 is included in that of Q_2 (since hiphop music is popular).

—The result of Q_3 is included in that of Q_4 (since M. Jackson in Figure 1 appears also with P. McCartney in Figure 2).

—The result of Q_5 is included in that of Q_6 (since singers from Paris are also French).

This means that answering queries Q_1-Q_6 without accessing the whole table Albums, can be done using only the answers to Q_2, Q_4 and Q_6.

Ignoring such implications between queries (and consequently predicates) would lead, in multimedia applications, to:

(1) High computation cost when creating fragments

(2) Creation of large fragments, which is very restrictive for multimedia storage, migration, and retrieval

(3) Data duplication on several sites, when the storage is distributed. In essence, there are two types of dependencies or implications that need to be considered when multimedia data come to play:

In essence, there are two types of dependencies or implications that need to be considered when multimedia data come to play:

— Multimedia inter-attributes (or functional) dependency: To illustrate the traditional inter-attribute dependency in our motivating example, let us consider the functional dependencies name→ birth (meaning that a given singer has a unique birth date) and name song → clip (stating that a song has a single clip for each of its singers). However, although carrying useful semantics, such dependencies are not enough in our context, since several attributes can be complex/multimedia and be the source of dependency. For instance, name→ picture cannot be expected to hold in the table Albums since singer's pictures may differ from one song to another. However, it is likely that two pictures, associated with two distinct songs by the same singer, have strong similarities. Such a constraint on pictures is not taken into account by traditional functional dependencies.

— Intra-attribute dependency: Intra-attribute dependency can be illustrated here through the attribute genre where a hierarchy over song types is used, stating for instance that hiphop music is a particular type of popular music, or that parisian albums are also french, etc.

3. Related Work

As mentioned earlier, fragmentation (or partition) techniques used in distributed database systems aim at removing irrelevant data access and reducing data exchange among related sites [Baiao and Mattoso 1998]. The accuracy of existing fragmentation

algorithms is closely related to their query/predicate implication policy since most of them use the set of predicates to determine the minterm fragments.

We recall that a minterm is a conjunction of simple predicates [Bellatreche et al. 1997] associated to a fragment and that three fundamental fragmentation strategies have been defined so far:

— Horizontal Fragmentation (HF): This technique underlines the partitioning of an entity/class in segments of tuples/ objects verifying certain criteria. The generated horizontal fragments have the same structure as the original entity/class. HF can be of two types:

— Primary or PHF: partitioning of an entity based on the values of its attributes [Özsu and Valduriez 1999].

— Derived or DHF: partitioning of an entity based on inter-links with other entities [Baiao and Mattoso 1998].

— Vertical Fragmentation (VF): This technique breaks down the logical structure of an entity/class by distributing its attributes/methods over vertical fragments, which would contain the same tuples/ objects with different attributes [Baiao and Mattoso 1998] (except the unique tuple/object identifier kept in all vertical fragments [Ezeife and Barker 1998] so as to link related segments).

— Mixed Fragmentation (MF): This technique is a hybrid partitioning technique where horizontal and vertical fragmentations are simultaneously applied on an entity/class [Navathe et al. 1995].

In the literature, two main PHF algorithms have been provided for relational DBMS:

— Com-Min algorithm [Özsu and Valduriez 1999]: This algorithm generates, from a set of simple predicates applied to a certain entity, a complete and minimal set of predicates used to determine the minterm fragments corresponding to that entity.

— Make-Partition graphical algorithm [Navathe and Ra 1989]: This algorithm generates minterm fragments by grouping predicates having high affinity towards one another.

It is worthy to note that the number of minterm fragments generated by Make-Partition is relatively smaller than the number of Com-Min minterm fragments [Navathe et al. 1995], due to the fact that the number of minterm fragments generated by Com-Min is exponential to the number of simple predicates considered.

For object oriented DBMS also, two main PHF algorithms have been studied in the literature: one developed by Ezeife and Barker using Com-Min [Özsu and Valduriez 1999], and the other developed by Bellatreche et al. [Bellatreche et al. 1997] on the basis of Make-Partition [Navathe and Ra 1989]. The use of Com-Min or Make-Partition is the major difference between them.

In parallel, and as a consequence of the various XML-oriented formats available on the web, generally utilized for multimedia data representation (such as SVG[1], SMIL[2], MPEG-7[3], etc.), several recent approaches have been provided for XML fragmentation [Mahboubi and Darmont 2009], [Süß 2001].

As XML-based files are usually verbose (and can thus become extremely large in size), and have a huge number of users accessing them, XML fragmentation becomes prominent. The usage of XPath and XML predicates forms the common basis of all these studies. Nevertheless, XML fragmentation approaches are very specific and hardly applicable to multimedia databases.

For multimedia data fragmentation, only two approaches have been recently provided in the literature where the authors focus on the implication problem between predicates and try to improve and adapt existing algorithms:

— One provided by Saad et al. in [Saad et al. 2006], where the authors discuss multimedia primary horizontal fragmentation. Implications between low-level multimedia predicates are identified and utilized as input of the traditional fragmentation algorithms (Com-Min [Özsu and Valduriez 1999] and Make-Partition [Navathe and Ra 1989]), in order to efficiently partition the multimedia database.

[1]http://www.w3.org/Graphics/SVG/
[2]http://www.w3.org/TR/REC-smil/
[3]http://www.chiariglione.org/mpeg/standards/mpeg-7/

— One provided by Getahun et al. in [Getahun et al. 2007a; 2007b], where the authors discuss the issue of identifying semantic implications between textual-based multimedia predicates, and propose to integrate knowledge bases as a framework for assessing the semantic relatedness between predicate values and operators.

Although these two approaches are interesting, they do not fully consider characteristics related to multimedia attributes when computing implication between predicates, and more precisely they do not address multimedia inter-attributes (or functional) dependencies. In addition, they only consider the primary horizontal fragmentation type. It is to be noted that several dedicated studies have been provided in the literature aiming at studying the functional dependency problem in a multimedia context [Chang et al. 2007; Polese and Chang 2001] (without addressing specifically the fragmentation problem). However, these approaches did not address the issues of predicate implication and query pre-ordering, as we do in this article.

4. Back Ground

4.1 Preliminaries

In this work, we consider that the data are stored in a table defined over two kinds of attributes, namely atomic attributes and multimedia attributes. More precisely, we assume a fixed attribute set $U = A \cup M$ where:

— $A = \{A_1,...,A_p\}$ and each A_i ($i = 1,...,p$) is an atomic attribute associated with a set of atomic values (such as strings, numbers, etc.) called the *domain* of A_i and denoted by *dom* (A_i).

— $M = \{M_1,...,M_q\}$ and each M_j (j = 1,...,q) is a multimedia attribute, associated with a set of complex values (usually represented as sets of values or vectors) commonly called multimedia features (such as color, texture, shape, loudness, pitch, brightness, etc.). The domain of M_j is denoted by *dom* (M_j).

Thus, given a table Δ defined over U, tuples t in Δ are denoted as $\langle a_1,...,a_p,m_1,...,m_q\rangle$ where a_i is in *dom* (A_i) ($1 \leq i \leq p$) and m_j is in *dom* (M_j)($1 \leq i \leq q$). Moreover, every a_i is denoted by $t.A_i$ and every mj is denoted by $t.M_j$.

4.2 Distance and Similarity

Since we consider two kinds of attributes, comparing subtuples of tuples in Δ comes down to comparing atomic values, multimedia values, or both.

On one hand, when multimedia objects come to play, comparing two atomic attribute values using standard operators (<, >, Like, =, etc.) can be inefficient and inaccurate. More particularly, when multimedia objects are associated with textual or spatial attributes (e.g. genre and place attributes in our running example, respectively), the use of such operators becomes inappropriate since they do not take into account the related semantics (e.g., comparing the strings hiphop to popular music, Paris to France, etc. would not provide any usable result in the context of our running example).

In essence, associating semantics to multimedia objects (for description and retrieval purposes for instance) is a complex task, because (i) the description of a multimedia object is subjective and depends on each user, and (*ii*) the content of a multimedia object cannot always be defined by a set of terms (words and/or expressions).

To overcome these limitations and to consider the semantics behind terms and values, semanticbased operators (and distances) need to be defined and used. This has been done so far in the literature using knowledge bases (KB).We recall that knowledge bases (thesauri, taxonomies, semantic networks, and/or ontologies) are utilized in the fields of Natural Language Processing (NLP) and Information Retrieval (IR) to compare/match the considered entities (words or expressions [Smeaton and Quigley 1996; Lin 1998], generic concepts [Rodriguez and Egenhofer 2003; Ehrig and Sure 2004], web pages [Maguitman et al. 2005], etc.) with respect to their corresponding relevance degrees with one another. This comes down to a hierarchical structure with a set of concepts (representing or subsuming groups of words, expressions or terms), and a set of relations connecting them. Several relations are commonly used in the literature to measure similarity such as *Synonymy*, *Hyponymy* (or IsA), *Hypernymy* (or HasA), *Meronymy* (or ParOf), and *Holonymy* (or HasPart). Figure 3 shows an extract of a sample KB used in Geographical Information Systems in France.

Current techniques for computing semantic-based distance between words/expressions can be classified as edge-based and node-based. Methods of the former group are straightforward and generally estimate similarity as the shortest path (in edges, edge weights, or nodes) between the two concepts being compared [Rada et al. 1989]. With node-based approaches, the

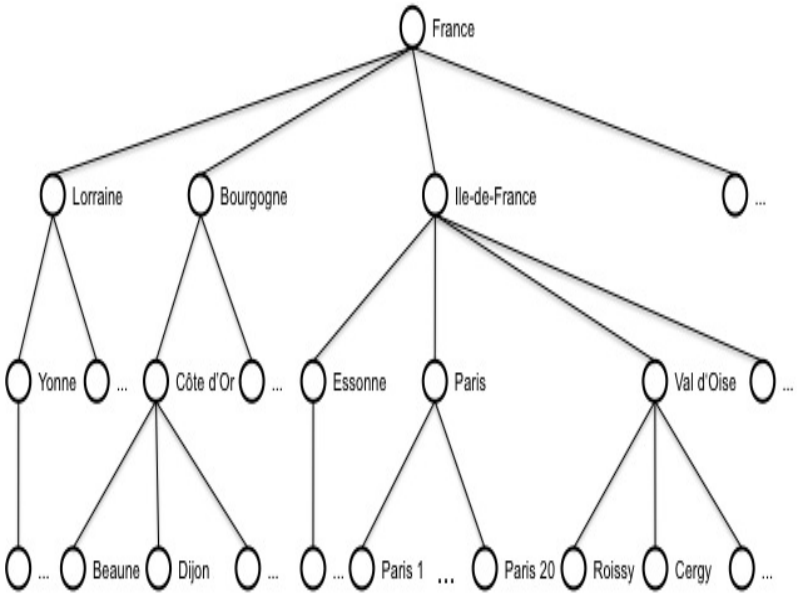

Figure 3. A sample *KB* with one relationship *belongs*

definition of similarity is more sophisticated and is estimated as the maximum amount of information content they share in common. With the help of KB, this common information carrier can be identified as the most specific common ancestor (also known as Lowest Common Ancestor or LCA) that subsumes both concepts being compared [Resnik 1995]. It is important to note that although for every A in *A*, *dom*(*A*) can have its own knowledge base, it is commonly assumed that one global KB is shared among all atomic attribute domains (with if needed one virtual root being the ancestor of all the roots of each specific domain KB).

On the other hand, similarly of atomic attributes, comparing two values from the domain of a given multimedia attribute cannot be done using traditional operators. For example, in Figure 4, we can hardly guess which operator would best correspond with the comparison status of the two images.

Consequently, similarity-based operators (and distances) have been explored in the literature sincethe last two decades to provide more adaptable multimedia comparison [Atnafu et al. ; Adali et al. 1998]. The similarity between two values over a multimedia attribute in *M* can be computed using various distance measures defined on feature spaces (color, texture, etc.) [Androutsos et al. 1999; Jeong et al.].

In our approach, we take into account these important features of atomic and multimedia attributes by assuming that, given an attribute A in U, either atomic or multimedia, several distances can be defined over *dom*(A). Consequently, the similarity of two values *a* and *a'* in *dom*(A) is defined according to a function that aggregates the results of distances defined over *dom*(A).

Figure 4. How to compare these two images?

Definition 1. Let A be an attribute in U (atomic or multimedia) over which n distance functions $d^1_A,...,d^n_A$ are used. We associate A with a global distance function, denoted by w_a such that for all a_1 and a_2 in *dom*(A) $\omega_A(a_1, a_2) = g_A(d^1_A(a_1, a_2),...,d^n_A(a_1, a_2))$ where g_A is an aggregation function assumed to satisfy the following triangular co-norm properties:

— $g_A(0,...,0; x_i, 0,..., 0) = x_i$ (zero identity),

— if for every i = 1,..., n, $x^1_i \leq x^2_i$, then $g_A(x^1_1,..., x^1_n) \leq g_A(x^2_1,..., x^2_n)$ (monotonicity),

— for every permutation ω of {1,..., n}, $g_A(x_1,...,x_n) = g_A(x_{\omega(1)},..., x_{\omega(n)})$ (commutativity),

— $g_A(x^1_1,..., x^1_{i-1}, g_A(x^2_1,..., x^2_n), x^1_{i+1},..., x^1_n) = g_A(x^1_1,..., g_A(x^1_{i-1}, x^2_1,..., x^2_{n-1}), x^2_n, x^1_{i+1},..., x^1_n)$ (associativity).

We note that if n = 1, then g_A is the identity function and so, $\omega_A(a_1, a_2) = d^1_A(a_1, a_2)$, for all a_1 and a_2 in *dom*(A).

When a distance comes to play, two main similarity operators can be commonly employed: range and k-nearest neighbor. The Similarity Range query retrieves the objects in the database that are dissimilar from a given reference up to a given threshold; and the *k*-Nearest Neighbor query (also known as a top-*k* query) retrieves the k objects most similar to a given reference. A formal definition for each one of these query types can be found in [Böhm et al. 2001; Chávez et al. 2001]. In what follows, given an attribute A in U and a real number $\in$, when writing expressions such as $\omega_A(a_1, a_2) \leq \in$, $\in$ refers to the radius of the range operator or to the number of *k* neighbors to be returned, depending on the similarity operator being considered.

Based on the previous definition, when considering an attribute A, either atomic or multimedia, we define a similarity measure between two values in *dom*(A) as follows.

Definition 2. Let A be an attribute in U (atomic or multimedia) and $\in$ a positive real number. For all a_1 and a_2 in dom(A), a_1 and a_2 are said to be similar within , denoted by $a_1 \cong_{\varepsilon} a_2$, if

To illustrate this, let us consider again our motivating scenario of Example 1. We assume that the multimedia attributes Picture and Clip are associated to global distance functions, respectively denoted by $\omega_{picture}$ and ω_{clip}. Using these measures, the distance between two pictures or two clips can then be computed in order to assess their similarities. Similarly for name, birth and song attributes, we assume that they are compared according to the trivial distance where the distance from an element to itself is 0 and the distance between two distinct elements is ∞ . In this way, for every $\in$, $a_1 \cong_{\varepsilon} a_2$ holds if and only if $a_1 = a_2$. Thus, assuming such a distance over *dom*(name), *dom*(birth) and *dom*(song) means that two names, two birth dates, or two song titles are similar within any if and only if they are equal. Regarding the other attributes (e.g., place and genre), they include some semantics. Since their related domains can be defined with respect to a predefined knowledge base, the distances between attribute values can be easily computed.

For instance, the distance between two places p1 and p2, denoted as ω_{place}(p1, p2), can be computed as follows, according to the predefined KB shown in Figure 4.2: ω_{place}(p1, p2) is the height of the minimal subtree of KB whose leaves are p1 and p2. In this case, it is easy to see that ω_{place} is a distance for which ω_{place}(Dijon , Beaune) = ω_{place}(Cergy , Roissy) = 1 and ω_{place}(Pari ,Bourgogne) = 2. Therefore, Dijon $\cong_1$ Beaune and Cergy $\cong_1$ Roissy both hold, whereas Paris $\cong_1$ Bourgogne does not.

4.3 Multimedia Functional Dependencies

In order to provide relevant data fragmentation approach, we extend the standard notion of functional dependency to the multimedia context. To this end, we assume that for every A in U, ω_A is associated to a given threshold $\in_A$. Two elements *a* and *a'* are said to be similar if $\omega_A(a, a') \leq \in_A$ holds. More generally, the notion of similarity over an attribute set X is defined as follows.

Definition 3. Let X be an attribute set and *t* and *t'* two tuples over U. *t* and *t'* are said to be similar over X, denoted $t \sim_X t'$, if for every A in *X* , $t.A \cong_{\varepsilon A} t'.A$ holds.

A multimedia functional dependency is an expression of the form X $\rightsquigarrow$ Y where X and Y are two nonempty sets of attributes (atomic or multimedia). The notion of multimedia functional dependency satisfaction is defined as follows.

A multimedia functional dependency is an expression of the form X $\rightsquigarrow$ Y where X and Y are two nonempty sets of attributes (atomic or multimedia). The notion of multimedia functional dependency satisfaction is defined as follows.

Definition 4. Let X and Y be two nonempty subsets of U. The table Δ is said to satisfy the multimedia functional dependency from X to Y , denoted by $\Delta \models X \rightsquigarrow Y$, if for all t and t' in , $(t \sim_X t') \Rightarrow (t \sim_y t')$.

We point out that multimedia functional dependencies are a generalization of standard functional dependencies, in the sense that for every table Δ over U, if $\Delta \models X \rightarrow Y$, then $\Delta \models X \rightsquigarrow Y$. Roughly speaking, a standard functional dependency $X \rightarrow Y$ can be seen as a multimedia functional dependency $X \rightsquigarrow Y$ in which all distance thresholds of attributes in XUY have been set to 0.

Example 2. In the context of our motivating scenario of Example 1, we assume the following set MD of multimedia functional dependencies:

— nam;birth place picture
— song ; genre
— name song ; clip.

where the attributes are associated with the following similarity thresholds:

— $\in_{name} = \in_{birth} = \in_{song} = 0$.

As the trivial distance has been assumed on attributes name, birth and song, we recall that names, birth dates and song titles are similar if and only if they are equal, whatever the similarity threshold.

— $\in_{place} = \in_{clip} = 0$.

Contrary to the previous item, the considered distances over *dom*(place) and *dom*(clip) are not the trivial distance. Choosing such similarity thresholds means that, in the table Albums:

—using the first multimedia functional dependency shown above, two tuples with the same singer name cannot be associated with two distinct birth places, and

—using the third multimedia functional dependency shown above, two tuples with the same singer name and song title cannot be associated with two distinct clips (even if these clips are close to each other according to ω_{clip}).

— $\in_{genre} = 1$.

In this case, according to the definition of ω_{genre}, two genres whose distance is at most 1 are considered similar. Based on the second multimedia functional dependency shown above, this implies that tuples with the same song but with distinct genres can be in Albums. For example, assuming that ω_{genre}(hiphop; popular) = 1, tuples t and t' such that *t.song = t'.song*, *t.genre = hiphop* and *t.genre = popular* can be in Albums.

— $\in_{picture}$ is chosen so as two pictures p1 and p2 such that $\omega_{picture}(p1 , p2) \leq$ picture concern the same singer. By doing so, considering the first multimedia functional dependency shown above, we allow the table Albums to contain distinct tuples with the same singer name associated with different but similar pictures, as stated in our motivating scenario. As a consequence, we assume that:

(1) $\in \leq \in_{picture}$,because in Example 1, Q_3 is assumed to retrieve all singer pictures representing M. Jackson, that is, all pictures similar to Figure 1.

(2) $\omega_{picture}$(Figure 1; Figure 2) $\omega >$ picture since Figure 2 represents not only M. Jackson, but also P. McCartney. By doing so, we consider that Figure 1 and Figure 2 are not similar.

Now, similarly to standard functional dependencies ([Ullman 1988]), it can be shown that Armstrong axioms are sound and complete for multimedia functional dependencies. To see this, we introduce the following notation: given a set MD of multimedia functional dependencies and two nonempty subsets of U, X and Y , we denote by $MD \models X \rightsquigarrow Y$ the fact that, for every table Δ over U, if $\Delta \models$ MD then $\Delta \models X \rightsquigarrow Y$.

We show that, as for standard functional dependencies, $MD \models X \rightsquigarrow Y$ if and only if $X \rightsquigarrow Y$ can be deduced from MD by applying repeatedly the following three inference rules:

A1- Pseudo-refexivity: If $Y \subseteq X$ then infer $X \rightsquigarrow Y$

A2- Augmentation: If $X \rightsquigarrow Y$ then, for every $Z \subseteq U$, infer $XZ \rightsquigarrow YZ$

A3- Transitivity: If $X \rightsquigarrow Y$ and $Y \rightsquigarrow Z$ then infer $X \rightsquigarrow Z$

Denoting by $MD \vdash X \rightsquigarrow Y$ the fact that $X \rightsquigarrow Y$ can be inferred from MD using axioms A1-A3 above, the following theorem states that these axioms are sound and complete.

Theorem 1. Let MD be a set of multimedia functional dependencies over U. Then for every relation Δ over U satisfying MD and for every $X \rightsquigarrow Y$ over U, we have $\Delta \models X \rightsquigarrow Y$ if and only if $MD \vdash X \rightsquigarrow Y$.

Proof. Soundness: It is easy to see that every axiom is sound in the sense that, for every axiom, every satisfying the premisses also satisfies the inferred dependency.

Completeness: Let us assume that $X \rightsquigarrow Y$ cannot be inferred from MD using the axioms. Denoting by X^+ the set of all attributes A in U such that $X \rightsquigarrow A$ is inferred from MD using the axioms, let $Z = U \setminus X^+$. Consider now the relation $\Delta = \{u_1 , u_2\}$ over U such that, for every $A \in X^+, \omega_A(u_1.A , u_2. A) = \in_A$, and for every B in $Z, \omega_B(u_1.B , u_2.B) > \in_B$.

We first prove by contraposition that satisfies MD. Let $X' \rightsquigarrow Y'$ in MD not satisfied by Δ. Then, for every $A \in X', \omega_A(u_1.A , u_2. A) \leq \in_A$, and there exists B in Y', $\omega_B((u_1.B , u_2.B) > \in_B$. Thus,

$X' \subseteq X^+$, which, by definition of X^+, shows that for every $A \in X'$, $X \rightsquigarrow A$ can be inferred. Thus, using A2 and A3, $X \rightsquigarrow X'$ can be inferred, which, using A3, implies that $X \rightsquigarrow Y'$ can be inferred from the axioms. Consequently, $Y' \subseteq X+$, which is a contradiction with the fact that there exists B in Y 0 such that $\omega_B(u_1.B , u_2.B) > \in_B$. Therefore, satisfies MD.

We now prove that does not satisfy $X \rightsquigarrow Y$. Indeed, if we assume that satisfies $X \rightsquigarrow Y$ then, either $X \not\subseteq X'$, or $Y \subseteq X^+$. However, axiom A1 shows that $X \subseteq X^+$ holds for every X, therefore, $Y \subseteq X^+$ holds. By definition of X^+, this implies that, for every $A \in X^+$, X ; A can be inferred from MD using the axioms. Thus, it can be seen that $X \rightsquigarrow Y$ can be inferred from MD using the axioms, which is a contradiction. Therefore, the proof is complete.

In what follows, we assume that satisfies a given set of multimedia functional dependencies. As in [Ullman 1988], we denote by MD^+ the set of all multimedia functional dependencies that can be derived from MD based on the Armstrong's axioms. Moreover, as specified in the proof just above, given an attribute set X , X^+ denotes the set of all attributes A such that $X \rightsquigarrow A$ is in MD^+. Then, X^+ is called the closure of X (with respect to MD).

To illustrate this, let us recall from Example 2 that the set of considered multimedia functional dependencies in our motivating scenario of Example 1 contains name $\rightsquigarrow$ birth place picture, song $\rightsquigarrow$ genre and name song $\rightsquigarrow$ clip.

Using axioms A1-A3 and Theorem 1, it can be seen that the first dependency can be equivalently replaced by the set of the following three multimedia functional dependencies: name $\rightsquigarrow$ birth, name $\rightsquigarrow$ place and name $\rightsquigarrow$ picture. Using this "decomposed" form, it can be seen, based on the similarity thresholds given here, that name $\rightsquigarrow$ birth, name $\rightsquigarrow$ place and name song $\rightsquigarrow$ clip are standard functional dependencies, whereas name $\rightsquigarrow$ picture and song $\rightsquigarrow$ genre are not.

It is easy to see that $name^+$ = name birth place picture, $song^+$ = song genre, and $(name\ song)^+$ =name birth place picture song genre cilp.

To end the section, we notice that, as in [Jen et al. 2010], multimedia functional dependencies can be extended to the empty attribute set as follows: for every attribute set X, $X \rightsquigarrow \emptyset$ always holds, whereas dependencies of the form $\emptyset \rightsquigarrow X$ are not considered. Consequently, this implies that$^+ \emptyset = \emptyset$, for every set of multimedia functional dependencies MD.

5. Multimedia Queries

5.1 Basic Definitions

The multimedia queries considered in our approach are conjunctive projection-selection queries over, that is queries of the

form $\Pi_X \sigma_C(\Delta)$ such that X is a non empty subset of U and C is a conjunction of atomic selection predicates, defined as follows.

Definition 5. An atomic selection predicate P is an expression of the form (A $\tilde{=}_\varepsilon a$), where A$\in$ U, a$\in$ *dom*(A) and is a positive real number. The attribute A is called the schema of P, and is denoted by sch(P).

Given an atomic selection predicate P : (A $\tilde{=}_\varepsilon a$), sat(P) is the set of all in dom(A) such that $\alpha \cong_\varepsilon a$. A tuple t over U is said to satisfy an atomic selection predicate P, denoted by $t \models$P, if t:sch(P) $\in$ sat(P).

If C is a conjunction of atomic selection predicates, C : $P_1 \wedge \ldots \wedge P_p$, then a tuple t over U is said to satisfy C, denoted by t$\models$C, if t$\models P_i$ for every i = 1,...,p. The schema of C, denoted by sch(C), is the set of all attributes sch(P_i), for i = 1,...,p.

A query of the form $\Pi_X \sigma_C(\Delta)$ where X is a subset of U and C is a conjunction of atomic selection predicates over pairwise distinct attributes, is referred to as a multimedia query over Δ. The set of all these queries is denoted by MQ(Δ).

Now given a multimedia query q, the answer to q in Δ is defined as follows.

Definition 6. Let q = $\Pi_X \sigma_C(\Delta)$ be in MQ(Δ). The answer to q in , denoted by q(Δ), is the set of all X-values of all tuples t in Δ such that $t \models$C, that is, q(Δ) = {t.X t$\in \wedge$ t$\models$ C}.

Example 3. Referring back to Example 1, the queries Q_1-Q_6 can be written according to Definition 5 as follows:

Example 3. Referring back to Example 1, the queries Q_1-Q_6 can be written according to Definition 5 as follows:

—Q_1 : $\Pi_{song} \sigma_{C1}$ (Albums), where C_1 : genre $\tilde{=}_0$ hiphop

—Q_2 : $\Pi_{song} \sigma_{C2}$ (Albums), where C_2 : genre $\tilde{=}_1$ popular

—Q_3 : $\Pi_{picture} \sigma_{C3}$ (Albums), where C_3 : picture $\tilde{=}_\varepsilon$ Figure 1

—Q_4 : $\Pi_{picture} \sigma_{C4}$ (Albums), where C_4 : picture $\tilde{=}_{\varepsilon'}$ Figure 2

—Q_5 : $\Pi_{clip\ picture} \sigma_{C5}$ (Albums), where C_5 : place $\tilde{=}_0$ Paris

—Q_6 : $\Pi_{clip\ picture} \sigma_{C6}$ (Albums), where C_6 : place $\tilde{=}_1$ France

It should be noticed that the values in every C_i (i = 1 ,..., 6) are chosen so as to fit the comments made in the introductory section. That is:

— Choosing 0 in C_1 means that the queries are meant to select respectively song titles by hiphop singers, only. Similarly, choosing 0 in C_5 means that only the clips and pictures of singers born in Paris are selected.

— Assuming that the genre-values in the table Albums are only leaf values of the adopted KB, choosing 0 in C_2 would lead to an empty answer. On the other hand, choosing 1 in C_2, allows to return all song titles whose genre g is such that ω_{genre}(g , popular) = 1, which includes hiphop.

A similar remark holds for C_6, and thus, choosing 1 in C_6, allows to return all clips and pictures of singers whose birth place p is such that ω_{place}(p , Paris) = 1, which includes Paris.

—The value of $\in$ in C_3 is such that all pictures in Album of the singer in Figure 1 can be retrieved.

The value of $\in'$ in C4 must be chosen so as all singers appearing in Figure 2 can be matched. Thus, it is likely that $\in'$ has to be chosen greater than $\in$.

On the other hand, making sure that the pictures returned by Q_3 are among those returned by Q_4 depends on the contents of Figure 1 and Figure 2, that is on the value of $\omega_{picture}$ (Figure 1 , Figure 2). This point is discussed in the next section.

5.2 Predicate Entailment

We define the notion of selection predicate entailment as follows.

Definition 7. Let C and C' be two selection predicates. Then, C is said to entail C', denoted by C ▷ C', if for every tuple t over U, (t ⊨ C)) (t ⊨ C') holds.

Clearly, for all selection predicates C and C', if C ▷ C' then $\sigma_{C}(\Delta) \subseteq \sigma_{C'}(\Delta)$. Selection predicate entailment is characterized in our approach according to the following proposition.

Proposition 1. Let C : $P_1 \wedge \ldots \wedge P_p$ and C' : $P'_1 \wedge \ldots \wedge P'_{p'}$ be two selection predicates such that for every *i* in {1 ,..., p}, sat(P_i) ≠ ∅ and sat(P_i) ≠ *dom*(sch(P_i)), and for every *i'* in {1,..., p'}, sat($P'_{i'}$) ≠ ∅ and sat($P'_{i'}$) ≠ *dom*(sch($P'_{i'}$)).

Then, C ▷ C' holds if and only if, for every *i'* in {1 ,..., p'}, there exists i in {1,...,p} such that sch(P_i) = sch($P'_{i'}$) and sat(P_i) sat($P'_{i'}$).

Proof. Let us first assume that for every *i'* ∈ {1,..., p'}, there exists i ∈ {1,..., p} such that sch(P_i) = sch($P'_{i'}$) and sat(P_i) ⊆ sat($P'_{i'}$). Let t be such that t ⊨ C, and $P'_{i'}$ in C' such that sch($P'_{i'}$) = A'. Denoting by $P_{i'}$ one of the atomic selection predicates in C such that sat($P_{i'}$) sat($P'_{i'}$), we have sch($P_{i'}$) = sch($P'_{i'}$) = A'. Since t ⊨ C, t.A' is in sat($P_{i'}$) and thus, t.A' is in sat($P'_{i'}$). Consequently, for every *i'* ∈ {1,..., p'}, t:sch($P'_{i'}$) ∈ sat($P'_{i'}$), which shows that t ⊨ C' and thus that C ▷ C'.

Conversely, let *i'* ∈ {1 ,...., p'} be such that for every i ∈ {1,..., p}, either sch(Pi) ≠ sch($P_{i'}$) or sat(Pi) ⊄ sat($P_{i'}$). Denoting by A' the attribute sch($P_{i'}$), given *i* in {1,...,p}, we separately consider the cases where (*i*) A' = sch(Pi) and (*ii*) A' ≠ sch(Pi).

(i) If A' = sch(Pi), then sat(Pi) ⊄ sat($P_{i'}$), and so, there exists $\alpha \in dom(A)$ such that $\alpha \in$ sat(Pi) and $\alpha \notin$ sat($P_{i'}$). Thus, the hypotheses on C show that there exists a tuple t over U such that for every i ∈ {1,..., n}, t.sch(Pi) ∈ sat(Pi) and t.A' = α . Then, t ⊨ C and t 6⊨ C', which implies that C ▷ C' does not hold.

(ii) If sch(Pi) ≠ sch($P_{i'}$), based on case (i) above, we assume that for every i ∈ {1,..., p}, A' ≠ sch(Pi). Then, the hypotheses on C and C' imply that there exists a tuple t over U such that t ⊭ C and t.A' ⊭ $P_{i'}$. Indeed, t can be built up as follows: for every i ∈ {1,..., p} choose t.A_i so as t.A_i ∈ sat(Pi), then choose t.A' so as t.A' ∉ sat($P_{i'}$) and for every remaining attribute A, choose any value in *dom*(A) for t.A. Thus, in this case again, t ⊭ C', and so, C ▷ C' does not hold. Therefore, the proof is complete.

As will be seen next in this section, predicate entailment is a basic issue for query comparison, which in turn, is used extensively in our fragmentation strategy. We notice in this respect that the hypotheses on C and C' in Proposition 1 simply state that the selection predicates under consideration are satisfiable by at least one tuple over U, and that every atomic selection predicate in these selection predicates do not cover the whole domain of the correspondent attribute. As in practice, these hypotheses are generally satisfied by selection predicates, we assume in the remainder of the article that they hold.

Then, it is important to note that Proposition 1 shows that, given two selection predicates C and C', checking whether C ▷ C' comes down to check atomic predicate entailment, which is the subject of the next proposition.

Proposition 2. For all atomic selection predicates P : (A ≷ $_{\varepsilon}$ a) and P' : (A ≷ $_{\varepsilon}$ a') such that sch(P) = sch(P') = A, let μ (P,P') be defined by: μ (P,P') = $\max_{\alpha}$ ({$\omega_A(a', \alpha)$ $\omega_A(a', \alpha) \leq \in$}). Then sat(P) ⊆ sat(P') holds if and only if μ (P,P') ≤ $\in'$.

Proof. For every α in sat(P), we have $\omega_A(a', \alpha) \leq \in$ and is in sat(P') if and only if $\omega_A(a', \alpha) \leq \in'$. Thus, if sat(P) sat(P') holds then μ (P,P') ≤. Conversely, assuming that μ (P,P') ≤ $\in'$, implies that for every α in sat(P), $\omega_A(a', \alpha) \leq \in'$, and thus that sat(P) ⊆ sat(P'). Therefore, the proof is complete.

Proposition 2 shows that, in order to efficiently test whether P ▷ P' holds for atomic selection predicates over the same attribute A, μ (P, P') must be efficiently computed, which depends of the definition of the distance ω_A. We discuss below this important point for various distances.

First, if ω_A is the trivial distance, then we clearly have μ (P,P') = 0 if and only if *a* = *a'* and μ (P,P') = ∞ otherwise. Thus, in

this case P⊳ P' can be easily checked.

If we assume now that ω_A is a distance operating on $\mathbb{R}^n$ for some positive integer n (such as any Minkowski distance or the cosine distance), then *dom*(A) contains an element α_0 such that μ (P, P') = $\omega_A(a', \alpha)$ + $\omega_A(a', \alpha_0)$ and $\omega_A(a', \alpha_0) = \in$. Thus in this case, which generally happens for multimedia attributes, Proposition 2 shows that P⊳ P' holds if and only if ω_A(a', a) + $\in \leq \in'$. We emphasize that, as this inequality depends on the expressions of P and P' only, P⊳ P' is efficiently checked in this case.

As an example, let Q_3 and Q_4 be the queries given in Example 3, in which C_3 : picture ξ_ε Figure 1 and C4 : picture ξ_ε figure 2. Proposition 2 along with the previous remark show that choosing $\in$ and $\in'$ so as $\omega_{picture}$(Figure 1, Figure 2) + $\in \leq \in'$ implies that $C_3 \rhd C_4$ holds. Thus, in this case, as expected in Example 1, Q_3(Albums) is a subset of Q_4(Albums).

Unfortunately, as shown in the following example, in the case of discrete attribute domains, such an easy computation of μ (P, P') does not always hold.

Example 4. Recalling from Example 3 that the selection predicates C_1 and C_2 in the queries Q_1 : $\Pi_{song}\sigma_{C1}$ (Albums) and Q_2 : $\Pi_{song}\sigma_{C2}$ (Albums) are defined by C_1 : genre ξ_0 hiphop and C_2 : genre ξ_1 popular, let C'$_1$ be defined by genre $\cong_{\in'}$ hiphop where $\in'_1$ = 0:5.

Since ω_{genre}(hiphop , popular) = 1, ω_{genre}(hiphop , popular) + $\in'_1 > \in$ (because ω_{genre}(hiphop , popular) + $\in'_1$ = 1:5 and $\in_2$ = 1). But, as for all genre-values g and g', ω_{genre}(g ,g') is an integer, sat(C'$_1$) = sat(C_1) = fhiphopg, and so, C'$_1$ C_2. We note that, in this case, μ (C'$_1$,C_2) = 1, and so, that μ (C'$_1$, C_2) $\leq \in$.

On the other hand, considering C_1 instead of C'$_1$, we have ω_{genre}(hiphop ,popular) = 1, $\in_1$ = 0 and $\in_2$ = 1. Since sat(C_1) contains the single value hiphop, μ (C_1;C_2) = ω_{genre}(hiphop ,popular) + $\in_1$, and so, μ (C_1,C_2) $\leq \in$, which, by Proposition 2, entails that $C_1 \rhd C_2$. Therefore, as expected in Example 1, Q_1(Albums) is a subset of Q_2(Albums).

We note that, regarding the queries Q_5 and Q_6 of Example 3, as for Q_1 and Q_2, we have C_5 C_6. Thus, in this case again and as expected in Example 1, Q_5(Albums) is a subset of Q_6(Albums).

To sum up our discussion on the computation of μ (P, P'), a general and efficient approach to this computation is currently unknown to the authors. However, we notice that for standard distances, such as the trivial distance and all standard distances operating on Rn for some positive integer n, (P; P0) is computed without any access to the data. Of course, the case of discrete distances has to be investigated further, because these distances, among which are those based on a hierarchy, play an important role in our approach.

6. Query Comparison

In this section, we introduce a pre-ordering relation over the set of multimedia queries of MQ(Δ) and we show this pre-ordering allows for a definition of query equivalence that plays a key role in our data fragmentation approach.

6.1 Query Pre-ordering

In order to define our way of comparing queries, we associate every query with a set of attributes for which selection predicate satisfaction implies similarity.

Definition 8. Let q = $\Pi_x\sigma_C(\Delta)$ be in MQ(Δ). The attribute set SIM(q) is the set of all attributes A in sch(C) such that (A ξ_ε a) occurs in C and $\in \leq \in_A$.

Example 5. Referring to the queries of Exemple 3, we have Q_1 : $\Pi_{song}\sigma_{C1}$ (Albums), where C_1 : genre ξ_0 hiphop. Since $\in_{genre}$ = 1 (see Example 2), we have SIM(Q_1) = genre. Similarly, as Q_2 : $\Pi_{song}\sigma_{C2}$ (Albums), where C_2 : genre ξ_1 popular , we also have SIM(Q_2) = $\in_{genre}$.

On the other hand, since $Q_5 : \Pi_{\text{clip picture}} \sigma_{C5}$ (Albums), where C_5 : place $\equiv_0$ Paris and $Q_6 : \Pi_{\text{clip picture}} \sigma_{C6}$ (Albums), where C_6 : place $\equiv_1$ France, we have SIM (Q_5) = place and SIM(Q_6) = $\emptyset$ because $\in_{\text{genre}} = 0$ (see Example 2).

Regarding now the queries $Q_3 : \Pi_{\text{picture}} \sigma_{C3}$ (Albums), where C_3 : picture $\equiv_\varepsilon$ Figure 1 and $Q_4 : \Pi_{\text{picture}} \sigma_{C4}$ (Albums), where C_4 : picture $\equiv_{\varepsilon'}$ Figure 2, we recall that:

(1) In Example 2, it is assumed that $\in \leq \in_{\text{picture}}$ and that ω_{picture}(Figure 1, Figure 2) $> \in_{\text{picture.}}$

(2) In Example 4, it is assumed that ω_{picture}(Figure 1, Figure 2) + $\in \leq \in'$ so as $C_3 \rhd C_4$ holds.

Thus, by item 1, we have $\in \leq \in_{\text{picture}} < \omega_{\text{picture}}$(Figure 1, Figure 2). As item 2 implies that ω_{picture}(Figure 1, Figure 2) $\leq \in'$, we obtain $\in \leq \in_{\text{picture}} < 0$. Therefore, we have SIM(Q_3) = picture and SIM(Q_4) = $\emptyset$.

Based on Definition 8, predicate entailment and multimedia functional dependencies, queries are compared according to the following relation, borrowed from [Jen et al. 2010].

Definition 9. Let $q_1 = \Pi_{x1} \sigma_{C1}(\Delta)$ and $q_2 = \Pi_{x2} \sigma_{C2}(\Delta)$ be in MQ(Δ). Then, q_2 is said to be more specific than q_1, denotedby$q_1 \preceq q_2$, if:

(1) $C_2 \rhd C_1$,

(2) SIM(q_1) SIM(q_2), and

(3) X_1 SIM(q_2) $\rightsquigarrow X_2$ is in MD^+.

Example 6. We recall from Example 2 that the set MD of multimedia functional dependencies of interest in our motivating scenario of Example 1 is

MD ={name $\rightsquigarrow$ birth place picture, song $\rightsquigarrow$ genre,name song $\rightsquigarrow$ clip}

Considering the queries $Q_1 : \Pi_{\text{song}} \sigma_{C1}$ (Albums) and $Q_2 : \Pi_{\text{song}} \sigma_{C2}$ (Albums), where C_1 : genre $\equiv_0$ hiphop and C_2 : genre $\equiv_1$ popular, we have $Q_2 \preceq Q_1$, because:

(1) By Example 4, we have $C_2 \rhd C_1$.

(2) By Example 5, we have SIM(Q_1) = SIM(Q_2) = genre.

(3) song genre $\rightsquigarrow$ song is a trivial multimedia dependency that can be inferred from MD.

It can be seen in a similar way that we also have $Q_4 \preceq Q_3$, $Q_6 \preceq Q_5$. As a more sophisticated example of query comparison, let q1 and q2 be the following queries:

— $q_1 = \Pi_{\text{name birth}} \sigma_{C1}$ (Albums), where C_1 is defined by C_1 : genre $\equiv_1$ popular. This query asks for all names and birth dates of all singers of popular songs.

— $q_2 = \Pi_{\text{clip picture}} \sigma_{C2}$ (Albums), where C_2 is defined by C_2 : (genre $\equiv_0$ hiphop) $\wedge$ (song $\equiv_0$mysong). This query asks for the clips and pictures of hiphop singers singing the song entitled mysong.

We have$q_1 \preceq q_2$since it is easy to see that $C_2 \rhd C_1$, SIM(q_1) $\subseteq$ SIM(q_2) (because SIM(q_1) = genre and SIM(q_2) = genre song), and that name birth genre song $\rightsquigarrow$ clip picture can be inferred from MD, using axioms A1-A3.

As in [Jen et al. 2010], the relation $\preceq$ is shown to be a pre-ordering over MQ(Δ), that is is reflexive and transitive.

Proposition 3. The relation is a pre-ordering over the set of all queries in MQ(Δ).

Proof. As reflexivity is trivial, we only prove the transitivity of the relation. Let $q_1 = \Pi_{x1} \sigma_{C1}(\Delta)$, $q_2 = \Pi_{x2} \sigma_{C2}(\Delta)$, $q_3 = \Pi_{x3} \sigma_{C3}(\Delta)$ be in MQ such that $q_1 \preceq q_2$ and $q_1 \preceq q_2$ hold.

• As by Definition 9(1), we have $C_2 \rhd C_1$and $C_3 \rhd C_2$, this implies by Proposition 1 that, for every P_i : (A $\equiv_e a_i$) in C_3, there exists P_j : (A $\equiv_e a_j$) in C_2 such that sat(P_i) $\subseteq$ sat(P_j), and that there exists P_k : (A $\equiv_e a_k$) in C1 such that sat(P_j) $\subseteq$ sat(P_k). Thus, we have sat(P_i) $\subseteq$ sat(P_k), which shows that $C_3 \rhd C_1$.

• By Definition 9(2), we have $SIM(q_1) \subseteq SIM(q_2)$ and $SIM(q_2) \subseteq SIM(q_3)$, and thus, $SIM(q_1) \subseteq SIM(q_3)$ holds.

• Definition 9(3) shows that $X_1 SIM(q_2) \rightsquigarrow X_2$ and $X_2 SIM(q_3) \rightsquigarrow X_3$ are in MD^+. In order to prove that $X_1 SIM(q_3) \rightsquigarrow X_3$ is also in MD^+, we first note that since $SIM(q_1) \subseteq SIM(q_2)$, $SIM(q_2) \subseteq SIM(q_3)$ and $SIM(q_1) \subseteq SIM(q_3)$ hold, by axiom A_1, $SIM(q_2) \rightsquigarrow SIM(q_1)$, $SIM(q_3) \rightsquigarrow SIM(q_2)$ and $SIM(q_3) \rightsquigarrow SIM(q_1)$ are in MD^+. Moreover, as $SIM(q_3) \rightsquigarrow SIM(q_2)$ is in MD+, axiom A_2 implies that $X_1 SIM(q_3) \rightsquigarrow X_1 SIM(q_2)$ is in MD^+. As $X_1 SIM(q_2) \rightsquigarrow X_2$ is assumed to be in MD^+, by axioms A_2 and A_3, we obtain that $X_1 SIM(q_3) \rightsquigarrow X_2 SIM(q_3)$ is also in MD^+. Using the fact that $X_2 SIM(q_3) \rightsquigarrow X_3$ is in MD^+, by axiom A_3, $X_1 SIM(q_3) \rightsquigarrow X_3$ is in MD^+, and the proof is complete.

6.2 Query Equivalence

We notice $\preceq$ that is not an ordering, because, as shown in the following example, there exist distinct queries q_1 and q_2 such that $q_1 \preceq q_2$ and $q_1 \preceq q_2$ hold.

Example 7. In the context of our motivating scenario of Example 1, let us consider the queries q_1 of Example 6 defined by $q_1 = \Pi_{\text{name birth}} \sigma_{C1}$ (Albums), where C_1 is defined by C_1 : genre $\equiv$ popular, along with $q'_1 = \Pi_{\text{name birth}} \sigma_{C'1}$(Albums), where $C'_1 = C_1$ Since $C'_1 = C_1$, $C_1 \rhd C'_1$ and $C'_1 \rhd C_1$ trivially hold. Furthermore, as $SIM(q_1)$ = genre, we have $SIM(q_1) = SIM(q'_1)$ = genre. Using now the multimedia functional dependencies of MD given in Example 2, it is easy to see that name birth genre $\rightsquigarrow$ name place genre and name place genre ; name birth genre can be inferred from MD. Hence, q_1 and q'_1 are distinct queries such that $q_1 \preceq q'_1$ and $q'_1 \preceq q_1$ both hold.

Queries q_1 and q_2 in MQ(Δ) such that $q_1 \preceq q_2$ and $q_2 \preceq q_1$ are said to be equivalent, which we denote by $q_1 \equiv q_2$. It should be clear that the relation is an equivalence relation over MQ(Δ). The following proposition characterizes equivalent queries.

Proposition 4. For all queries $q_1 = \Pi_{x1} \sigma_{C1}(\Delta)$, $q_2 = \Pi_{x2} \sigma_{C2}(\Delta)$ in MQ(Δ), $q_1 \equiv q_2$ if and only if

(1) $C_1 \rhd C_2$ and $C_2 \rhd C_1$ hold.

(2) $SIM(q_1) = SIM(q_2)$.

(3) $(X_1 SIM(q_1))+ = (X_2 SIM(q_2))+$.

Proof. Let us first assume that $q_1 \equiv q_2$. Thus, we have $q_1 \preceq q_2$ and $q_2 \preceq q_1$, which, by Definition 9, entails that $C_1 \rhd C_2$ and $C_2 \rhd C_1$, and that $SIM(q_1) = SIM(q_2)$. Moreover, we also havem that $X_1 SIM(q_2) \rightsquigarrow X_2$ and $X_2 SIM(q_1) \rightsquigarrow X_1$ are in MD+, which combined with $SIM(q_1) = SIM(q_2)$, entails that $X_i SIM(q_i) \rightsquigarrow X_j SIM(q_j)$ are in MD+ for i,j = 1,2. Thus, we obtain $(X_1 SIM(q_1))+ = (X_2 SIM(q_2))+$.

Conversely, if we assume that $C_1 \rhd C_2$, $C_2 \rhd C_1$, $SIM(q_1) = SIM(q_2)$, and $(X_1 SIM(q_1))+ = (X_2 SIM(q_2))+$, it is easy to see that $q_1 \preceq q_2$ and $q_2 \preceq q_1$ both hold, which completes the proof.

Example 8. Referring back to the queries q_1 and q'_1 in Example 7, Proposition 4 implies that $q_1 \equiv q'_1$. Moreover, Proposition 4 also shows that these queries are equivalent to the query $q^*_1 = \Pi_{X^*1} \sigma_{C1}(\Delta)$ (Albums) where X^*_1 = name birth place picture genre. This is so because:

— $C_1 \rhd C'_1$ and $C'_1 \rhd C_1$ hold,

— $SIM(q_1) = SIM(q'_1) = SIM(q^*_1)$ = genre, and

— (name birth genre)$^+$ = (name place genre)$^+ = X^*_1$.

7. Multimedia Data Fragmentation

Based on the formalism introduced so far, we now explain how our approach can be used for multimedia data fragmentation. We recall in this respect that we assume that we are given:

— A table Δ satisfying a set MD of multimedia functional dependencies.

— A set of frequently asked queries, which we denote by FQ.

7.1 Fragment Definition and Properties

Our goal is to define a fragmentation of Δ allowing for efficiently answering the queries in FQ, and possibly other queries that are characterized below. Roughly, our fragmentation approach works as follows:

(1) Every query q in FQ is replaced with an equivalent query q^* which allows to answer efficiently all queries equivalent to q. We denote by FQ^* the obtained set of queries.

(2) The set FQ^*_{min} of all queries in FQ^* that are minimal with respect to $\preceq$ is identified.

(3) For every query q^* in FQ^*_{min}, a fragment, denoted by $\varphi(q^*)$ is created as a set of tables containing

(a) The answer $q^*(\Delta)$ to q^*

(b) An auxiliary table associated to every q^*_i of FQ^* such that $q^* \preceq q^*_i$, so as to efficiently answer every query equivalent to q^*_i, without accessing Δ.

In order to formalize our fragmentation approach, we first define the queries of FQ^* as follows: Given $q = \Pi_X \sigma_C(\Delta)$ in $MQ(\Delta)$, the query q^* is defined as $q^* = \Pi_{X^*} \sigma_C(\Delta)$ where $X = (X\ SIM(q))^+$.

Therefore, we have $FQ^* = \{q^*\ q \in FQ\}$. We note that the size of FQ^* is smaller than or equal to that of FQ. In particular, if FQ contains two equivalent queries q_1 and q_2, then $q^*_1 = q^*_2$ and thus, only one query is considered in our fragmentation approach.

Moreover, the following proposition shows that the answers to all queries equivalent to q^* are simply projections of $q^*(\Delta)$.

Proposition 5. For all queries $q = \Pi_X \sigma_C(\Delta)$ and $q' = \Pi_{X'} \sigma_C(\Delta)$ in $MQ(\Delta)$ such that $q \equiv q'$, $q'(\Delta) = \Pi_{X'}(q^*(\Delta))$.

Proof. By Proposition 4, we have $q \equiv q' \equiv q^*$ and $(X\ SIM(q)) = (X'\ SIM(q')) = X$. Thus, $X' \subseteq X$, showing that the expression $\Pi_{X'}(q^*(\Delta))$ makes sense. Moreover, if t' is in $q'(\Delta)$, then there exists t in such that $t \models C'$ and $t' = t.X'$. As $C' \rhd C$, $t \models C$, and thus, $t.X^* \in q^*(\Delta)$. Since $X' \subseteq X$, $t.X' = (t.X^*).X'$, and so, $t' \in \Pi_{X'}(q^*(\Delta))$. Hence, $q^*(\Delta) \subseteq \Pi_{X'}(q^*(\Delta))$.

Conversely, let t' be in $\Pi_{X'}(q^*(\Delta))$. Then, there exists $t^* \in q^*(\Delta)$ such that $t' = t^*.X'$, which implies that there exists $t \in \Delta$ such that $t \models C$ and $t.X^* = t^*$. Thus, $t' = (t.X^*):X0 = t.X' = t.X'$. Moreover, as $C \rhd C'$, $t \models C'$, and thus, $t' \in q^*(\Delta)$. Hence, $\Pi_{X'}(q^*(\Delta)) \subseteq q'(\Delta)$, which implies that $q'(\Delta) = \Pi_{X'}(q^*(\Delta))$.

Example 9. Let us assume that, in the context of our motivating scenario of Example 1, FQ is the set of queries Q_1-Q_6. Since it has been seen in Example 6 that $Q_2 \preceq Q_1$, $Q_4 \preceq Q_3$ and $Q_6 \preceq Q_5$, we obtain $FQ^*_{min} = \{Q^*_2, Q^*_4, Q^*_6\}$.

Considering that (song genre)+ = song genre, (picture)+ = picture and (clip picture place)+ = clip picture place, the different answers to store are

— $Q^*_2(Albums) = \Pi_{song\ genre} \sigma_{C2}(Albums)$, where C_2 : genre = popular,
— $Q^*_4(Albums) = \Pi_{picture} \sigma_{C4}(Albums)$, where C_4 : picture = Figure 2, and
— $Q^*_6(Albums) = \Pi_{clip\ picture\ place} \sigma_{C6}(Albums)$, where C_6 : place = France.

It should be clear that, instead of storing all six answers to Q_1-Q_6, we only store three answers, which avoids storing redundancies, because as will be seen later, based on these three answers, those of Q_1, Q_3 and Q_5 can be efficiently computed.

Notice that, although $Q_4 = Q_4$, storing $Q_4(Albums)$ allows to answer Q_3 because, intuitively, all pictures returned by Q_3 are among those returned by Q_4. In the case of Q_6, storing $Q^*_6(Albums)$ allows to answer all queries $q = \Pi_X \sigma_C(Albums)$ such that $X \subseteq$ clip picture place, $C \rhd C_6$ and $sch(C) \subseteq sch(C_6)$. We emphasize that Q_5 and, for instance, $\Pi_{clip} \sigma_{place} = {}_{0Dijon}(Albums)$ are such queries. It should be clear that similar remarks hold for Q_2, and thus, storing $Q_2(Albums)$ allows to efficiently answer to Q_1, as well as to other queries, such as the query $\Pi_{song} \sigma_{genre} = {}_{0rock'n\ roll}(Albums)$.

In order to show that the general case is more sophisticated than the previous examples, let us now assume that $FQ = \{q_1, q_2\}$, where are the two queries given in Example 6, that is:

— $q_1 = q^*_1$ (Albums) with C_1 : genre Ξ_1 popular and

— $q_2 = \Pi_{\text{clip picture}} \sigma_{C2}$ (Albums) with C_2 : (genre Ξ_0 hiphop) $\wedge$ (song Ξ_0 mysong).

Since it has been shown that $q_1 \preceq q_2$, we have $FQ^*_{min} = \{q^*_1\}$, and thus q^*_1(Albums) is stored according to our approach. However, recalling from Example 8 that q^*_1 is defined by $\Pi_{X^*_1} \sigma_{C1}$ (Albums) where X^*_1 = name birth place picture genre, no song-values and no clip-values are stored in the corresponding fragment, and thus, q^*_1(Albums) does not allow to compute the answer to q_2.

In order to cope with the problem mentioned just above, given $q^*_0 = \Pi_{X^*_0} \sigma_{C^*_0}(\Delta)$ in FQ^*_{min}, for every $q^* = \Pi_{X^*} \sigma_{C^*}(\Delta)$ in FQ^* such that $q^*_0 \preceq q^*$ an additional table, denoted by $\Delta(q^*_0, q^*)$, is built up as follows:

— We associate every tuple v of q^*_0 with a unique identifier, denoted by Id(v), and stored in $q^*_0(\Delta)$ as an extra column called Id.

— $\Delta(q^*_0, q^*)$ is a table defined over $\{Id\} \cup (X^* \setminus X^*_0)$, containing the following tuples:
For every tuple v in $q^*_0(\Delta)$, if Δ contains a tuple t such that $t.X^*_0 = v$ and $t \models C^*$, then the tuple w defined by w:Id = Id(v) and $w.(X^* \setminus X^*_0) = t.(X^* \setminus X^*_0)$ belongs to $\Delta(q^*_0, q^*)$.

Referring back to Example 9, in the case of the fragment corresponding to Q_6, i.e., $\wp(Q^*_6)$, as $Q_6 \prec Q_5$, the table Albums $(Q^*_6 Q^*_5)$ is defined over the single attribute Id (since Q^*_6 and Q^*_5 are defined over the same schema clip picture place) and contains all tuple identifiers of tuples in Q^*_6 (Albums) associated in Albums with a singer born in Paris.

On the other hand, in the case of the fragment $\wp(q^*_1)$, additionally to the table q_1(Albums), we consider the table Albums (q^*_1, q^*_2) over Id clip song, because in this case, $X^*_2 \setminus X^*_1$ = clip song. Moreover, this table contains the triples (Id(v),c, s), where v is in q_1(Albums) and is associated in Albums with the song whose clip is c and whose title s is such that $s \Xi_0$ mysong (that is, such that s = mysong).

The following proposition states that every query in $FQ^* \setminus FQ^*_{min}$ can be answered through a projection-join of the tables stored in one segment.

Proposition 6. Let $q^* = \Pi_{X^*} \sigma_{C^*}(\Delta)$ be a query in $FQ^* \setminus FQ^*_{min}$. Denoting by q^*_0 a query in FQ^*_{min} such that $q^*_0 \prec q^*$, then $q^*(\Delta) = \Pi_{X^*}(q^*_0(\Delta) \bowtie \Delta(q^*_0, q^*))$.

Proof. We first note that, by definition of $\Delta(q^*_0, q^*)$, the join is preformed on tuple identifiers and the join is defined over $X^*_0 \cup X^*$ which contains X^*. Moreover, for every tuple τ in the join, there exists $v \in q^*_0(\Delta)$ and $w \in \Delta(q^*_0, q^*)$ such that there exists t in satisfying the following: $t.X^*_0 = v$, $t.X^* = w$ and $t \models C$. Thus, $\tau.X^* \in q^*_0(\Delta)$.

Conversely, for every τ in $q^*(\Delta)$, there exits $t \in \Delta$ such that $t.X^* = \tau$ and $t \models C^*$. As $q^*_0 \prec q^*$, $t \models C^*_0$, and thus $t.X^*_0 \in q^*_0(\Delta)$. Since $t.(X^* \setminus X^*_0)$ appears in $\Delta(q^*_0, q^*)$ associated with the identifier of some $v = t.X^*_0$ in $q^*(\Delta)$, $t.(X^* \cup X^*_0) \in q^*_0(\Delta) \bowtie \Delta(q^*_0, q^*)$. Hence, $\tau \in \Pi_{X^*}(q^*_0(\Delta) \bowtie \Delta(q^*_0, q^*))$, because $\tau = t.(X^* \cup X^*_0)$. Therefore, the proof is complete.

Combining Proposition 5 and Proposition 6, we obtain the following basic corollary showing that all queries equivalent to a query in FQ are answered, based on the tables stored in the fragments only.

Corollary 1. Let $q = \Pi_X \sigma_C(\Delta)$ be in $MQ(\Delta)$ and FQ be a set of frequently asked queries. Then, if there exists q_0 in FQ such that $q \equiv q_0$, $q(\Delta)$ can be computed through a projection or a projection-join of the tables stored in one of the fragments built up from FQ.

Proof. If q^*_0 is in FQ^*_{min}, then Proposition 5 shows that $q(\Delta)$ is a projection of $q^*_0(\Delta)$. Otherwise, FQmin contains a query q1 such that $q^*_1 \preceq |^*_0$. Thus, Proposition 6 shows that $q^*_0(\Delta)$ is obtained by a projection of $q^*_0(\Delta) \bowtie \Delta(q^*_0, q^*)$. Thus, applying Proposition 5 to $q^*_0(\Delta)$ completes the proof.

It is important to note that, based on Corollary 1, additional queries can be answered using the tables stored in the fragments. Namely, every query $q = \Pi_X \sigma_C(\Delta)$ for which there exists $q^*_0 = \Pi_{X^*_0} \sigma_{C^*_0}(\Delta)$ in FQ such that $q \preceq q^*_0$ and $(X \cup sch(C)) \subseteq X^*_0$,

can be answered using the tables stored in the fragment corresponding to q^*_0. This so because Corollary 1 shows that $q^*_0(\Delta)$ can be computed using its corresponding fragment and, moreover, our assumption about schema inclusion implies that all attributes for computing the projection and the selection in $q(\Delta)$ are available.

As an example of such query, let us consider again the query $q = \Pi_{song}\sigma_{genre \cong_0 \text{rock'n roll}}$ (Albums) (mentioned in Example 9). Although, q is not equivalent to any query in FQ, we have $Q_2 \preceq q$ (because genre $\cong_0$ rock'n roll is assumed to entail genre $\cong_1$ popular), and all attributes occurring in q are among those occurring in Q_2. In particular the attribute genre appears in Q_2, and thus, it is possible to know whether a tuple in Q_2 (Albums) satisfies or not the selection predicate genre $\cong_0$ rock'n roll.

To sum up our fragmentation approach, we emphasize that, contrary to standard approaches, the semantics of the data, in terms of multimedia functional dependencies, allows to compare queries and to characterize query equivalence. Based on this basic feature of our work, it turns out that our fragmentation approach offers a trade off between redundancy avoidance and extra storage in order to answer any many queries as possible. Indeed:

—Redundancies are avoided by fully storing the answers of minimal queries, while the answers to non minimal queries are only partially stored (but can be fully recovered).

—On the other hand, considering the queries in FQ^* requires to store more attributes than if queries in FQ were considered. However, this is the price to pay for answering queries other than those in FQ, based on the information stored in the fragments.

In this section, we argue that our approach does not raise computational difficulties, although providing an appropriate technique for storing the answers to given queries (namely, the queries in FQ), while allowing to answer further queries.

Indeed, regarding firstly query answering, it has been seen previously that all answers to queries under consideration are obtained through a projection or a projection-join (see Corollary 1), or a selection-projection-join (see the remark following Corollary 1) of the tables stored in one fragment. Moreover, since joins are performed according to tuple identifiers, standard indexing techniques can be used to optimize join computations.

Regarding now the computation of fragments, we notice that determining whether two queries are comparable according to $\preceq$ only requires to know the similarity thresholds associated to every attribute (so as to compute the set SIM(q) associated to a query q) and to perform predicate entailments and inferences based on axioms A1-A3. Apart from predicate entailment that has to be investigated further, such computations can be efficiently implemented and, in any case, we point out that they do not require to access the table Δ.

Consequently, the computation of the sets FQ^* and FQ^*_{min} does not raise computational difficulties, and consequently, for every fragment $\varphi(q^*_0)$, the schemas of all additional tables $\Delta(q^*_0, q^*)$ in $\varphi(q^*_0)$ are also easily computed.

The last issue to be investigated is the computation of the content of all tables in the fragments. Algorithm 1 shows that all tables of one given fragment are computed through only one scan of the underlying table Δ. Therefore, the complexity of the computation of all fragments can be said to be linear in the number of scans of the table Δ.

8. Conclusion

In this article, we have proposed a novel approach to multimedia data mixed fragmentation, based on query comparison and query equivalence. We emphasize that the main feature of our approach, is that the way queries are compared makes explicit use of the semantics of the data to be fragmented.

This semantics is defined through the novel notion multimedia functional dependencies, which is a generalization of standard functional dependencies. It has been argued that multimedia functional dependencies take into account the specific features of multimedia data, and an important point in this respect is that we have shown that multimedia functional dependencies are axiomatized in the same way as standard functional dependencies.

Given a set of frequently asked queries FD, an important feature of our fragmentation strategy is to store the answers to particular queries in FD, and that these answers not only allow for efficiently computing the answers of all queries in FD, but also of additional queries that have been characterize using the notion of query equivalence.

Algorithm 1 Fragment Computation

Input: The table Δ, and
- A query $q_0^S = \pi_{X_0^S}\sigma_{C_0^S}(\Delta)$ in FQ_{min}^S
- The set $FQ^S = \{q_1^S, \ldots, q_k^S\}$ such that, for $i = 1, \ldots, k$, $q_i^S = \pi_{X_i^S}\sigma_{C_i^S}(\Delta)$ and $q_0^S \prec q_i^S$.

Output: All tables in the fragment $\varphi(q_0^S)$, that is:
- The table $q_0^S(\Delta)$
- All tables $\Delta(q_0^S, q_i^S)$, for $i = 1, \ldots, k$.

```
1: for all t in Δ do
2:   if t ⊨ C_0^S then
3:     if t.X_0^S ∉ q_0^S(Δ) then
4:       insert t.X_0^S into q_0^S(Δ) associated with a new identifier Id(t)
5:     else
6:       let Id(t) be the identifier of t.X_0^S in q_0^S(Δ)
7:   for all i = 1,...,k do
8:     if t ⊨ C_i^S then
9:       if t.(X_i^S \ X_0^S) does not occur in Δ(q_0^S, q_i^S) associated with Id(t) then
10:        insert t.(X_i^S \ X_0^S) into Δ(q_0^S, q_i^S) associated with Id(t)
11: return φ(q_0^S)
```

Based on this work, several issues have to be further investigated. First, we plan to implement our fragmentation method, in order to assess its efficiency and compare it to existing approaches. Second, the problem of testing atomic predicate entailment is a key issue of our approach that has to be studied more deeply. More specifically, a condition on distances allowing to fully characterize atomic predicate entailment is a theoretical issue that will be addressed in the near future. Third, the computation of the answers to queries in which atomic selection predicates are distributed in several queries whose answers are stored, is an important issue that will be addressed in the near future.

References

[1] Adali, S., Bonatti, P., Sapino, M., Subrahmanian, V. S., A multi-similarity algebra. In Proc. ACM SIGMOD98. pp. 402–413.

[2] Androutsos, D., Plataniotis, K. N., Venetsanopoulos, A. N. A novel vector-based approach to color image retrieval using a vector angular-based distance measure. *Computer Vision and Image Understanding*, 75, 46–58.

[3]Atnafu, S., Brunie, L., Kosch, H. Similarity-based operators in image database systems. *In:* Advances in Web-Age Information Management, X. Wang, G. Yu, and H. Lu (Eds.). Lecture Notes in Computer Science, V.. 2118. Springer Berlin / Heidelberg, p. 14–25.

[4] Baiao, F., Mattoso, M., A mixed fragmentation algorithm for distributed object oriented databases. In 9th International Conference on Computing Information. p. 141–148.

[5] Bellatreche, L., Karlapalem, K., Simonet, A., Horizontal class partitioning in object-oriented databases. In Lecture Notes in Computer Science. p. 58–67.

[6] Böhm, C., Berchtold, S., Keim, D. A., Searching in high-dimensional spaces: Index structures for improving the performance of multimedia databases. ACM Comput. Surv. vol. 33, pp. 322–373, September, 2001.

[7] Braunmuller, B., Ester, M., Kriegel, H.-P., Sander, J., S, J. Efficiently supporting multiple similarity queries for mining in metric databases. *In:* ICDE '00: Proceedings of the 16th International Conference on Data Engineering. IEEE Computer Society, Washington, DC, USA, p. 256.

[8] Chang, S.-K., Deufemia, V., Polese, G., Vacca, M., A normalization framework for multimedia databases, *IEEE Transactions*

on Knowledge and Data Engineering, 19, p. 1666–1679.

[9] Chávez, E., Navarro, G., Baeza-Yates, R., Marroquín, J. L. Searching in metric spaces, *ACM Comput. Surv*. 33, p. 273–32.

[10] Chinchwadkar, G. S., Goh, A. , An overview of vertical partitioning in object oriented databases, *The Computer Journal*. 42.

[11] Ehrig, M., Sure, Y. Ontology mapping - an integrated approach. *In:* The Semantic Web: Research and Applications, LNCS. Springer Verlag, p. 76–91.

[12] Ezeife, C. I. and Barker, K. Distributed object based design: Vertical fragmentation of classes. Distrib. Parallel Databases 6 (4) 317–350, 1998.

[13] Getahun, F., Tekli, J., Atnafu, S., Chbeir, R., Towards efficient horizontal multimedia database fragmentation using semantic-based predicates implication. *In* XXII Simpósio Brasileiro de Banco de Dados (SBBD). p. 68–82, 2007a.

[14] Getahun, F., Tekli, J., Atnafu, S., and Chbeir, R. The use of semantic-based predicates implication to improve horizontal multimedia database fragmentation. In Workshop on The Many Faces of Multimedia Semantics (MS). pp. 29–38, 2007b.

[15] Jen, T., Laurent, D., Spyratos, N., Computing supports of conjunctive queries on relational tables with functional dependencies. *Fundamenta Informaticae*, 99 (3) 263–292.

[16] Jeong, S., Kim, S.-W., Kim, K., and Choi, B.-U. An effective method for approximating the euclidean distance in high-dimensional space. In Database and Expert Systems Applications, S. Bressan, J. Kaing, and R. Wagner (Eds.). Lecture Notes in Computer Science, vol. 4080. Springer Berlin / Heidelberg, pp. 863–872.

[17] Lin, D. , An information-theoretic definition of similarity. *In:* 15th International Conference on Machine Learning. Morgan Kaufmann, p. 296–304.

[18] Maguitman, A., Menczer, F., Roinestad, H., Vespignani, A., Algorithmic detection of semantic similarity. In WWW '05: 14th international conference on World Wide Web. p. 107–116.

[19] Mahboubi, H. and Darmont, J. Enhancing xml data warehouse query performance by fragmentation. In SAC '09: Proceedings of the 2009 ACM symposium on Applied Computing. p. 1555–1562, 2009.

[20] Navathe, S. B., Karlapalem, K., Ra, M.(1995). A mixed fragmentation methodology for initial distributed database design. Journal of Computer and Software Engineering 3 (4) 395–426.

[20] Navathe, S. B., Ra, M., Vertical partitioning for database design: a graphical algorithm. SIGMOD Rec.18 (2) 440–450.

[21] Özsu, M. T., Valduriez, P. Principles of distributed database systems (2nd ed.). Prentice-Hall, Inc., 1999.

[22] Polese, G. and Chang, S., Towards a theory of normalization for multimedia databases. Human-Centric Computing Languages and Environments, IEEE CS International Symposium on , p. 406, 2001.

[23] Rada, R., Mili, H., Bicknell, E., Blettner, M., Development and application of a metric on semantic nets, I*EEE Transactions on Systems, Man, and Cybernetics*. 19. p. 17–30.

[24] Resnik, P., Using information content to evaluate semantic similarity in a taxonomy. *In:* International Joint Conference on Artificial Intelligence (IJCAI). 1. 448–453.

[25] Rodriguez, M., Egenhofer, M., Determining semantic similarity among entity classes from different ontologies. *IEEE Transactions on Knowledge and Data Engineering*, 15, 442–456.

[26] Saad, S., Tekli, J., Chbeir, R., Yétongnon, K. , Towards multimedia fragmentation. In Advances in Databases and Information Systems (ADBIS). p. 415–429.

[27] Smeaton, A., Quigley, I. Experiments on using semantic distances between words in image caption retrieval. In . SIGIR '96: 19th ACM SIGIR conference on Research and development in information retrieval. ACM, pp. 174–180, 1996.

[28] Süß, C., An approach to the model-based fragmentation and relational storage of xml-documents. In Grundlagen von Datenbanken. p. 98–102.

[29] Ullman, J., Principles of Databases and Knowledge-Base Systems. Vol. 1. Computer Science Press.

Author Biographies:

Richard Chbeir received his PhD in Computer Science from the University of INSA-FRANCE in 2001and then his Habilitation degree in 2010 from the University of Bourgogne where he is currently an Associate Professor in the Computer Science Department in Dijon-France. His research interests are in the areas of multimedia information retrieval, XML and RSS Similarity, access control models, multimedia document annotation. Dr. Chbeir has published in international journals and books (IEEE Transactions on SMC, Journal of Methods of Information in medicine, JDIM, etc.), conferences (ACM SAC, Visual, IEEE CIT, FLAIRS, PDCS, etc.), and has served on the program committees of several international conferences (IEEE SITIS, ACM SAC, IEEE ISSPIT, EuroPar, etc.). He is currently the Chair of the French Chapter ACM SIGAPP and the vice-chair of ACM SIGAPP.

Home page: http://le2i.cnrs.fr/-Richard-Chbeir

Dominique Laurent received his doctoral degree in 1987 and then his Habilitation degree in 1994 from the University of Orléans (France). In 1988-1996, he was Assistant Professor in the University of Orléans, and then, Professor in the University of Tours (France) from September 1996 until September 2003. Since then, he is Professor at the University of Cergy-Pontoise (France), where lead the Department of Computer Science until 2010. He is currently the head of the Doctoral School of Sciences and Engineering of the University of Cergy-Pontoise. He is also a member of the research laboratory ETIS (a laboratory associated with the French Scientific Research Centre, CNRS).

His research interests include database theory, information systems, deductive databases, data mining, data integration, OLAP techniques and data warehousing.

Home page: http://depinfo.u-cergy.fr/~dlaurent