A Multi-paradigm Querying Approach for a Generic Multimedia Database Management System

Ji-Rong Wen[*] Qing Li[#] Wei-Ying Ma[*] Hong-Jiang Zhang[*]

[*]Microsoft Research Asia, Beijing, China {jrwen, wyma, hjzhang}@microsoft.com
[#]City University of Hong Kong qing.li@cityu.edu.hk

Abstract

To truly meet the requirements of multimedia database (MMDB) management, an integrated framework for modeling, managing and retrieving various kinds of media data in a uniform way is necessary. MediaLand is an experimental MMDB platform being developed at Microsoft Research Asia *for users with different levels of experiences and expertise to manage and search multimedia repositories easily, efficiently, and cooperatively. Key features of MediaLand include a uniform data model for describing all kinds of media objects and their relationships, and a 4-tier architecture based on this data model. In this paper, a multi-paradigm querying approach of MediaLand is presented, in which multimedia queries are processed based on a seamless integration of various existing search approaches. In doing so, MediaLand also offers the feature of "media independence" which is analogous to the notion of "data independence" from the classic ANSI SPARC standard. By incorporating a rich set of facilities and techniques, MediaLand lays down a good foundation for addressing further research issues, such as multimedia query rewriting, optimization, and presentation.*

Keywords: multimedia database management, multi-paradigm querying, uniform data modeling, media independence.

1. Introduction

An incommensurable amount of digital information is becoming available in digital libraries, on the WWW, and in professional and personal databases, and this amount is only growing. Information may be represented in various forms of media, such as images, graphics, video, audio, and text (collectively termed as *multimedia*). Traditional database systems, especially relational database system, have been very successful at the management of administrative data. However, managing large collections of digitized multimedia data is still a tough challenge, in spite of the fact that users have rapidly growing access to these information resources.

Information management has a long history and many approaches have been invented to manage and query diverse data types in the computer systems. The state-of-the-art approaches being used for information management can be classified into the following categories:

1) Conventional database system – This is the widely-used approach to manage and search for structured data. All data in a database system must conform to some predefined structures and constraints (i.e., schemas). To formulate a database query the user must specify which data objects are to be retrieved, the database tables from which they are to be extracted and predicate on which the retrieval is based. A query language for the database will generally be of the artificial kind, one with restricted syntax and vocabulary, such as SQL.
2) Information retrieval (IR) system – IR system is mainly used to search large text collections, in which the content of the (text) data is described by an indexer using keywords or a textual abstract, and keywords or natural language is used to express query demands.
3) Content based retrieval (CBR) system – This approach is used to retrieve desired multimedia objects from a large collection on the basis of *features* (such as colour, texture and shape, etc.) that can be automatically extracted from the objects themselves. Although keyword can be treated as a "feature" for text data, traditional information retrieval has much more higher performance than content-based retrieval because keyword has the proven ability to represent semantics, while no features have shown convincing semantic describing ability.
4) Graph or tree pattern matching – This approach aims to retrieve object sub-graphs from an object graph according to some denoted patterns.

Originally, the above approaches were designed to provide solutions to specific data types and application scenarios. None of them alone can be taken as the sole strategy in a hybrid environment containing diverse types of data, particularly multimedia data. This situation is worsened by the fact that user queries in such an

environment are often subjectively defined (and hence evaluated). Consider the case that a user is interested in finding a "dream house" in some particular city. While there are certain attributes which he/she can specify clearly (such as the price range, area size, etc.), many aspects can not be specified explicitly (eg, the style, appearance, feng-shui, neighbourhood, etc.) which would be much more effectively described through combinations of "visual" media (eg, photos, videos). In view of such cross-media queries, the main disadvantages of the above existing approaches for developing multimedia databases and applications become evident and striking, as listed below:

- The basic assumption of the traditional database methodology is that structured descriptive data for media data are readily provided. But in many cases the structured data usually are not sufficient or not available.
- The performance of content based retrieval techniques is not good in terms of precision and recall. Moreover, content based retrieval approaches usually are irrelevant to semantics.
- Graph or tree pattern matching approach only focuses on the structural part of media data and pays no or very little attention on the content and semantic part of media data.
- Since the structures of media data are essentially heterogeneous, there still lacks a uniform framework to model media data and, at the same time, provide powerful and flexible ways to manage and retrieve these data.
- Most of the existing retrieval approaches are designed for the professional users who are clear about what they want and how to search. But common users do not know or only know partially the structures of media data and the characteristics of the retrieval approaches. This is an even more noticeable problem when multiple retrieval approaches are applied together. Therefore, a flexible multimedia query language is needed to adequately meet the requirements of different users.

MediaLand is a database system aiming to provide the "true" support for multimedia data management. The objective of MediaLand is to provide an integrated framework for users with different levels of experiences to manage and search multimedia repositories easily, effectively, efficiently and intelligently. A uniform data modeling framework is thus developed for MediaLand to describe all kinds of media objects and the relationships among them. In addition, a novel multimedia database system architecture is designed for MediaLand based on the uniform data model. In this paper, a unified multi-paradigm querying methodology supported by MediaLand is described, which processes multimedia queries based on a seamless integration of various existing search approaches. As a by-product, MediaLand is able to support the notion of *media independence* which is analogous to the concept of "data independence" from the classic ANSI SPARC standard.

The rest of the paper is organized as follows. In section 2 we outline the modeling framework of MediaLand in terms of its logical and conceptual models. In section 3, a multi-paradigm querying methodology based on a 4-tier architecture of MediaLand is presented; we also describe the underlying specific operation constructs, and show how such operations can be used to accommodate powerful multimedia queries. Related work is given in section 4. Finally, section 5 summarizes this paper and offers a few research directions.

2. MediaLand Data Modeling Framework

The data modeling framework of MediaLand comprises the conceptual, logical, and physical models. In this section, we concentrate on the former two (ie, conceptual and logical data models), as the latter is very much implementation dependent.

2.1. Logical Model

2.1.1. Modeling media objects

Multimedia database objects differ from the traditional data items held within a database in that multimedia data objects are real world objects (such as video clips or graphical images), while the more traditional contents of a database are abstract concepts that describe external real world objects. In order to support multiple retrieval approaches in MediaLand, media objects should be described from multiple aspects. In general, a media object can be described by a six-tuple:

Media_Object = <*OID, Type, DB_Attribute, IR_Feature, CBR_Feature, Locator*>

where *OID* is the identifier of an object, *Type* is the type of the object, *DB_Attribute* is the structured data to describe the attributes of the object, *IR_Feature* is the IR related features (usually keywords) extracted from the object, *CBR_Feature* is the content features (such as color, texture, shape, etc.) extracted from the object, and *Locator* is a pointer to locate the media object. Figure 1 shows a diagrammatical description of a sample media object.

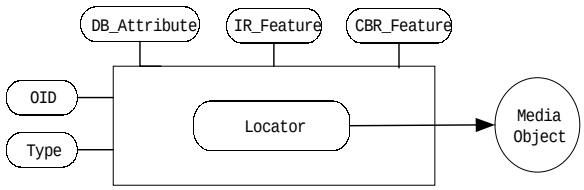

Figure 1. Description of a sample media object

Note that for different media objects the data structures of *DB_Attribute, IR_Feature* and *CBR_Feature* are

different. A universal definition for all media objects is neither feasible nor desirable in practice.

2.1.2. Modeling correlations among media objects

We use links to model the correlations among objects. The links in MediaLand are *typed* and *weighted*. The type information is used to distinguish the different link semantics among objects. And the weight information is used to measure how strong or week of a correlation between objects. In many cases, the links among objects are not static. We should allow that links can be added or deleted dynamically. Therefore, the links should not be modeled within the media objects. A separate mechanism is needed to model the links outside media objects. A link is also defined as a six-tuple:

Link = <LID, FromOID, ToOID, Type, Weight, Description>

where *LID* is the identifier of a link; *FromOID* and *ToOID* are the identifiers of the two objects connected by this link; *Type* is the type of the link; *Weight* is the weight of the link; and *Description* is the textual data used to describe the link.

Links can be built up by different applications through different ways. In general, there are mainly three ways to construct links:

1) Manually added by the users – For example, when a user manually makes a comment to a photo, a link is added to connect the photo and the comment document.
2) Automatically built by some tools and applications – An example is that video analysis tool can build some links between scenes and shots and between shots and frames. Another example is that a webpage analysis tool can build the links between the webpage and its components, or hyperlinks between two web pages.
3) Learnt through usages – For example, learning or mining through the user interactive and feedback information of an image retrieval system can build links between the similar images.

2.1.3. Multimedia object graph

In defining the logical model for MediaLand, one basic premise is that the model needs to be flexible enough to cater for much of the subjectivity and open-ended semantics of media objects. Indeed, users can accumulate a lot of media objects with diverse types through daily usages, such as Word documents, emails, personal photos, personal videos, mp3 files, and web pages, etc. These media objects are not independent. They correlate one another through many different ways. Thus in MediaLand, all media objects are connected by various types of links to construct a *multimedia object graph*. The formal definition is given as follows.

Definition 1: *A **multimedia object graph** (MOG for short) is a directed graph G=(CS, LS), where CS is a finite set of nodes and LS is a finite set of object links. Each element in CS corresponds to a media object $o_i \in O$, where O is the collection of media objects in the database. Each link in LS has the form of <lid, o_i, o_j, t, r, d>, denoting a semantic link (identified by lid) from O_i to O_j with t being the type, r being a real number indicating the weight of the link, and d being the description of the link.*

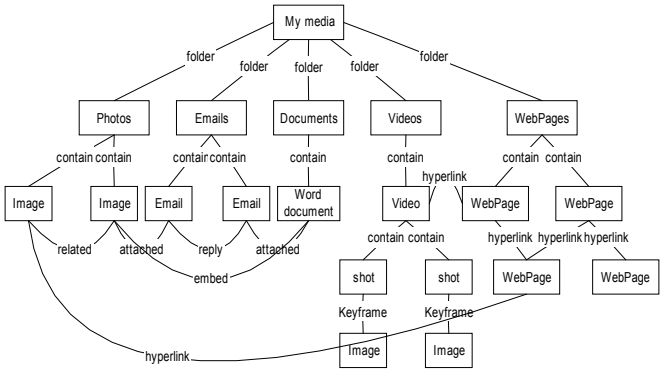

Figure 1. A sample personal collection of media data

Figure 1 illustrates a sample MOG which represents a personal multimedia collection on a personal computer. In Figure 1, each link is labled with a type name (eg, "hyperlink", "contain", "folder" etc.), and connects two media objects with some weight (not shown explicitly in the figure) to indicate the *strongness* of their association belonging to some type.

2.2. Conceptual Model

While the logical MOG model allows all media objects to be described flexibly with assigned heterogenous attributes, keywords, and features to them, from the management point of view it would be very inefficient to access and retrieve these objects. For this reason MediaLand also adopts the "schema" concept, that is, to assign uniform descriptive structures to the objects with the same type at the conceptual level. The collection of media objects with the same type is called a Media Class, which can be described by a six-tuple:

Media_Class = <OID_Domain, Type, DB_Schema, IR_Schema, CBR_Schema, Locator_Domain>

where *Type* is the common type of the objects in the class, *DB_Schema* is the schema to describe the attribute structures of the class, *IR_Schema* is the schema to describe IR features of the class, *CBR_Schema* is the schema to describe CBR features. Subsequently, a *Media_Object* is an instance of a *Media_Class* if and only if:

(1) *Media_Object.Type = Meida_Class.Type,*
(2) *DB_Attribute* is an instance of *DB_Schema,*
(3) *IR_Feature* is an instance of *IR_Schema,* and
(4) *CBR_Feature* is an instance of *CBR_Schema.*

Therefore, Media Class is a conceptual mechanism to cluster similar media objects and thus makes it more effective to manage, retrieve and browse media objects. We further force all objects belonging to the same class to have the same schema.

Since the descriptions of all links have the same data structure, there is only one class for all links in MediaLand. The schema of the Link Class is defined as:

Link_Class = <LID_Domain, FromOID_Domain, ToOID_Domain, Type_Domain, Weight_Domain, Description_Domain>

We now formally define a conceptual schema of MediaLand as follows.

Definition 2: *A **conceptual schema S** has a tripartite form **S** = <CS, LS, Ψ> where CS = {C_1, C_2, ..., C_n} is a set of Media Classes, LS is the Link Class having the form of <LID, C_i, C_j, T, W, D>, and* $\psi \subset CS \times LS$. *Each link l in LS denotes a semantic link from* $o_i \in C_i$ *to* $o_j \in C_j$ *with* $t \in T$ *being the link type,* $w \in W$ *being the weight of the link, and* $d \in D$ *being the description of the link.*

Figure 3 illustrates the relationships among the media objects, media classes, link class and various conceptual schemas.

In MediaLand, a media object can belong to more than one class. In other words, a media object can be modelled by multiple different schemas. For example, an email object can be described either by a complicated *email schema* with pertinent attributes such as subject, date, sender, etc. or by a simple *text schema* with only a keyword vector feature. The schema of a media class can thus be made up of one or more sub-schemas. An important point for MediaLand is that every IR_Schema, DB_Schema or CBR_Schema can be shared by multiple media classes. For example, the same "keyword vector" IR_Schema can be used both in an email class and a web page class. Therefore, the same type of descriptive data across multiple media classes are grouped into a single schema, which makes it convenient to provide cross-media retrieval support (as to be shown in section 3.3).

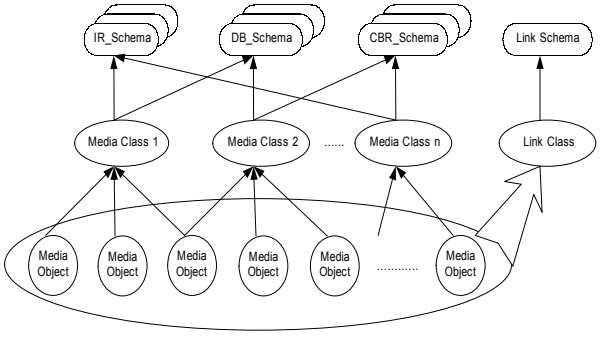

Figure 3. Relationships among objects, classes and schemas

3. A Multi-paradigm Querying Approach

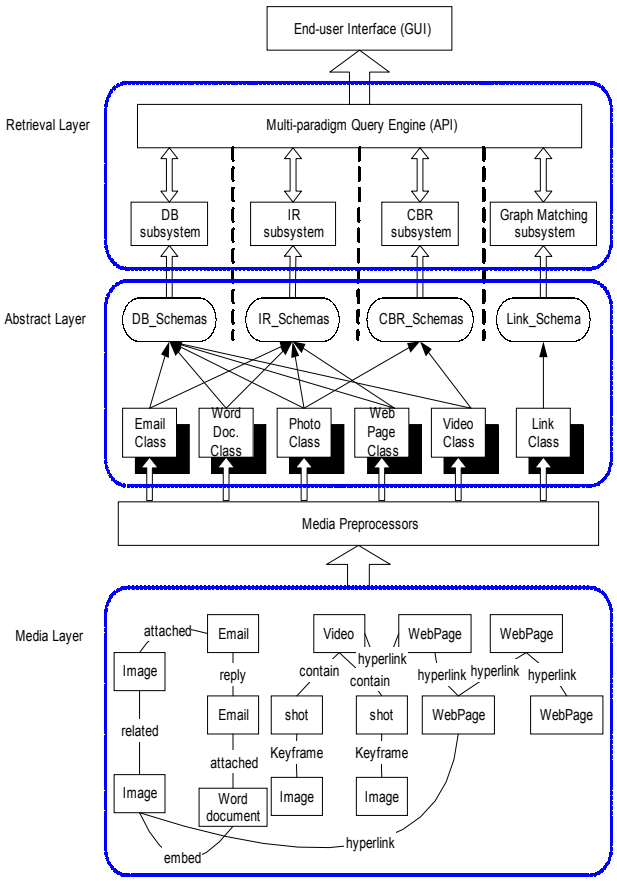

Figure 4. Multi-paradigm querying framework of MediaLand

In this section, we present a multi-paradigm querying approach supported by MedialLand. This approach is actually developed upon MedialLand's 4-tier architecture. We describe each of the layers, the necessary media preprocessor, and the querying constructs (operations) based on which complicated multimedia queries can be formulated and processed.

3.1. 4-tier Architecture

Figure 4 depicts the 4-tier architecture of MediaLand. As shown in Figure 4, these are Media Layer, Abstract Layer, Retrieval Layer, and the GUI (user interface) layer, respectively. These roughly correspond to the physical file system storage, logical and conceptual schemas, and the (application) view level with reference to the classic ANSI SPARC architecture. Below we introduce the details of each layer from bottom up.

3.2. Media Preprocessor

Media preprocessors are the connectors between Media Layer and Abstract Layer and are used to extract descriptive data from media data. In general, a specific

media preprocessor is needed for every media type. Figure 5 is a sample email preprocessor. But some preprocessors can be used by multiple media types. For instance, a keyword extractor can be used to extract keyword vectors from pure texts, word documents, emails, web pages, etc. Another example is that the feature extractors for images also can be applied to extract features from the key frames of videos. In some cases, if a media data contains other media data, the media preprocessor will call other preprocessors to deal with the embedded data. Referring to the example of Figure 5, if there is a photo attached to the email, the image preprocessor will be called to process the attached photo.

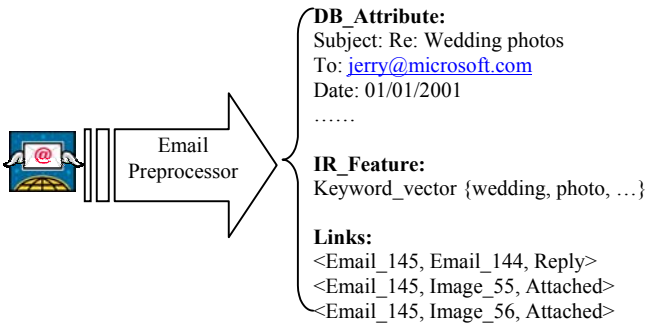

Figure 5. A sample Email preprocessor

3.3. Abstract Layer

Conceptually, every media class is made up of one or more schemas. All the attributes or features with the same type are clustered together. Such a kind of feature aggregation provides direct support of *media independence*. Each subsystem provides its specific retrieval method based on corresponding abstract data, not on the media data directly. Hence from the end-user's perspective the query strategy is media independent, in that he/she can pose queries with or without specifying from which media type(s) the search should be conducted. As to be shown in section 4, media independent search (or "cross media search") is extremely simple and easy to specify in MediaLand due to this property.

For convenience, the Abstract Layer also provides a **Class** operation to access objects of a given class, which is defined as:

Object_set **Class** (Type)

The result of the **Class** operation is an object set, which is formally defined as:

Definition 3: *An **object set** $S = \{OID_1, OID_2, \ldots, OID_n\}$ is a set of object IDs. There are no orders among the elements in the set.*

3.4. Retrieval Layer

3.4.1. Subsystems

The Retrieval Layer contains various retrieval subsystems, namely, DB subsystem, IR subsystem, CBR subsystem and Graph matching subsystem (cf. Figure 4). MediaLand do not impose any constraints on the underlying functions of the subsystems. In general, it only requires the subsystems to provide a few standard operations, which are commonly supported by nearly all existing retrieval systems.

3.4.1.1. DB subsystem

DB_Filter is the only operation required to be supported by the DB subsystem. This operation is used to filter out the objects satisfying some denoted conditions and is defined as:

Object_set **DB_Filter** (conditional_expression)

The parameter conditional_expression is similar to that of relational database. The result of **DB_Filter** operation is a set of qualified media objects.

Compared to traditional database query, the **DB_Filter** operation first filters out all candidate schemas whose attributes matching those in the conditional expression, then the conditional expression is used to select the qualified objects by searching across all candidate schemas.

3.4.1.2. IR & CBR subsystem

GradeSearch is the basic approximate search operation of IR and CBR systems and is defined as:

Object_list **GradeSearch** (Feature)

The result of the **GradeSearch** operation is an object list. An object list is a collection of objects and ranked by their relevance values to the denoted feature. Formally, an object list is defined as:

Definition 4: *An **object list** $L = \{(OID_1, w_1), (OID_2, w_2), \ldots, (OID_n, w_n)\}$ is a list of ordered OID-weight pairs. For arbitrary two pairs (OID_i, w_i) and (OID_j, w_j), $i \ j$ iff $w_i \ \ w_j$.*

The **GradeSearch** operation is used to select and rank the objects in the database by applying some information retrieval or content based retrieval approaches. Since there are a lot of features and retrieval models, this operation may be implemented through alternative ways. Moreover, it is also a good choice to provide several implementation functions for the **GradeSearch** simultaneously.

Typically, **GradeSearch** will retrieve all relevant objects irrespective of the similarity value. But usually users only want to retrieve the top N objects or the objects whose similarity value excess the denoted thresholds. Thus two kinds of operations are supplemented.

- Object_list **TopSearch**(Feature, Num)
  This operation is similar to the **GradeSearch** operation except that it only retrieves the top N

relevant objects (if there exists). We only need to add a parameter into the **GradeSearch** operation to transfer it into the **TopSearch** Operation.

- Object_list **ThresholdSearch**(Feature, Threshold)
  This operation is constructed by adding a threshold constraint to the **GradeSearch** operation and only those objects whose relevance values are larger than the threshold value are retrieved.

Sometimes, the retrieval result of approximate search operation, especially for **TopSearch** and **ThresholdSearch**, can be treated as an object set by omitting the weight values in the object list. A *Convert* function is defined to achieve this:

Object_set **Convert** (Object_list)

3.4.1.3. Graph matching subsystem

Since media objects are organized into an object graph through the links among them, graph pattern-matching techniques are employed to retrieve homomorphic subgraphs of a given pattern graph from the global object graph. First we define a new data type called homomorphic relation.

Definition 5: *A **homomorphic relation** from a graph G=(CS, LS) to a pattern graph G'=(CS', LS') is a relation R (f_1:OID, f_2:OID, ...f_n:OID). All of the fields in the relation have the same type OID. The number of fields is determined by the number of nodes in G'. For any $t \subseteq R$, if $x \to y$ is in G' then $t.x \to t.y$ is in G.*

It's clear that every tuple in the homomorphic relation is a qualified homomorphic subgraph of the given pattern graph. Now we are ready to define a basic operation for the graph matching subsystem, as the following:

*Homomorphic*_relation **Graph_Match** (pattern_graph)

Previous work point out that a main drawback of graph pattern-matching lies in its inherent computational complexity and the subgraph isomorphism problem is known to be NP-complete [26] [27]. But in MediaLand, this is not a big problem attributing to two reasons: Firstly, contrary to the classic graph matching problem, the links in our model are typed, which allow us to cut down the original object graph according to the link types in the pattern graph. Secondly, the graph matching operation usually is mixed with other operations and seldom used solely in MediaLand (as we will demonstrate in section 4). Thus other operations (usually much more efficient than the graph matching operation) can be executed first to filter out unwanted objects. These two ways can narrow down the search space dramatically and make graph matching a feasible search method in real application scenarios.

3.4.2. Multi-paradigm query engine

As mentioned earlier, one of the key characteristics emphasized by MediaLand is the support of multi-paradigm querying method. The multi-paradigm query engine targets to support complicated queries by combining the capabilities of multiple retrieval subsystems. In this section, we introduce the basic operations of merging the results from multiple subsystems. Unlike relational data model, which has only one data type – relation, in MediaLand three basic data types – object set, object list and homomorphic relation – are supported by different subsystems. Thus, a group of new operations which operate on heterogenous data types has been devised in MediaLand.

3.4.2.1. Intersection operations

Traditionally, intersection operation is used to get the common elements from two homogenous sets. In MediaLand, the functions of intersection operation are much broader than previous ones. A few new intersection operations are proposed to facilitate merging results of different data types.

Definition 6: *Intersection between two object sets S_1 and S_2 is defined as:*

$$S_1 \cap S_2 = \{ s \mid s \in S_1 \wedge s \in S_2 \}$$

The semantics of intersection between object sets is similar to traditional ones, that is, to simply select the common objects in the two object sets.

Definition 7: *Intersection between an object list L and an object set S is defined as:*

$$L \cap S = \{ l \mid l \in L \wedge l(OID) \in S \}$$

where l(OID) is a function to get the OID from the element l.

This intersection operation takes an object list as the first operant and an object set as the second operant. Its semantics is to filter out from the object list those objects not contained in the object set. The result of this operation is another object list keeping the order of objects in the original one.

Definition 8: *Intersection between a homomorphic relation R and an object set S is defined as:*

$$R \underset{f_i}{\cap} S = \{t \mid t \in R \wedge t(f_i) \in S\}$$

where f_i is the ith field in the relation and $t(f_i)$ is the value of the ith field of the tuple t.

The semantics of this kind of intersection operations is to filter out from the homomorphic relation those tuples whose value of the *ith* field is not contained in the object set. Another interpretation of this operation is that it equals to the natural join between the homomorphic relation and the object set.

Definition 9: *Intersection between two homomorphic relations R_1 and R_2 is defined as:*

$$R_1 \cap R_2 = \{ t \mid t \in R_1 \wedge t \in R_2 \}$$

R_1 and R_2 should have the same schemas.

The semantics of this intersection operation is the same as that of the relational model.

We do not define the intersection operation between two object lists because it is hard or, more precisely, meaningless to determine the weights and ranks of the objects in the result list. Instead, we define another operation called *Rerank* to merge the results of two object lists in Section 3.4.2.4. Moreover, it's obvious that it's meaningless to define intersection operation between an object list and a homomorphic relation since the semantics of such an operation is unclear.

3.4.2.2. Union operations

In MediaLand, union operations are used to combine the elements of two object sets or tuples of two homomorphic realtions, which has the same semantics as that of relational model.

Definition 10: *Union between two object sets S_1 and S_2 is defined as:*

$$S_1 \cup S_2 = \{ s \mid s \in S_1 \vee s \in S_2 \}$$

Definition 11: *Union between two homomorphic relations R_1 and R_2 is defined as:*

$$R_1 \cup R_2 = \{ t \mid t \in R_1 \vee t \in R_2 \}$$

Again, it is meaningless to union the elements of two object lists.

3.4.2.3. Rerank operation

Each retrieval subsystem retrieves and ranks objects independently. A mechanism of combining and re-ranking the objects from multiple object lists is an effective way to improve the retrieval performance, which also is a well-known research topic in the IR community [14] [16]. We define below a *Rerank* operation to merge two object lists and then reorder the objects according to a uniform criterion.

Definition 12: *Rerank of two object lists L_1 and L_2 is defined as:*

$$L_1 \underset{f}{\oplus} L_2 = \{ l \mid l \in L_1 \vee l \in L_2,$$

$$l(w) = f(\{k \mid k \in L_1 \vee k \in L_2, k(OID) = l(OID)\})\}$$

where f is a function to recalculate the weight of each element by combining both of the element's weights in L_1 and L_2.

There have been a lot of proposed weight recalculation functions in previous studies. Here we have only given the formal definition of Rerank operation. The weight recalculation function is itself an additional issue/topic and is beyond the scope of this paper.

4. Sample Application Query Processing

The operations described above can be combined to support very complicated multimedia queries. We give some query examples in this section to illustrate the usage and expressive power of these querying constructs.

Example 1 – Text retrieval

Consider the following query: *Find and rank all word documents according to the similarity between their keyword vectors and the sample vector {multimedia, database}.*

This query can be expressed by an intersection between a **GradeSearch** operation and a **Class** operation:

GradeSearch ({multimedia, database}) ∩ **Class** ("Word_document")

Example 2 – Content based retrieval

Let us consider another query: *Find and rank image objects according to the similarity between their color histogram features and the sample feature.*

Again, this query can be expressed by an intersection between a **GradeSearch** operation and a **Class** operation:

GradeSearch (sample_ColorHistogram_Feature) ∩ **Class** ("Image")

The above two examples show that existing traditional retrieval approaches can be easily covered by our multimedia query language. On the other hand, since all similar attributes and features from diverse media types are grouped together, the implementation of cross media retrieval in MediaLand is straightforward, as the following example shows.

Example 3 – Cross media retrieval

Suppose the user poses the following query: *Find and rank all objects that are relevant to "multimedia database".*

This query can be easily formulated through the following single operation:

GradSearch ({multimedia, database})

Note that the evaluation of this query will result in various media objects to be retrieved, so long that these objects have in their attributes/features the string of "multimedia database". (It is somewhat surprising to see how simple it is in MediaLand to fulfill cross media retrieval by just using a single operation in such a case.)

Example 4 – Combination of multiple retrieval methods

We now consider a rather "complex" query (as far as the specification in English is concerned): *Find all image objects with the date not earlier than "01/01/2001", and then rank the objects according to (1) the similarity between their shape feature and the sample feature, and (2) the similarity between their keyword vectors and the sample vector {sunset, ocean}.*

In MediaLand, the query can be specified through a combination of the following operations:

(**GradeSearch** ({sunset, ocean}) ⊕ **GradeSearch** (sample_Shape_Feature))
∩ **DB_Filter** ("date >= 01/01/2001")) ∩ **Class** ("Image")

First, two operations are used to retrieve and rank objects according to keyword and shape feature respectively. Second, the results of these two **GradeSearch** operations are merged and re-ranked. Then a **DB_Filter** and a **Class** operation are used to filter the object list further.

Example 5 – Associative retrieval

As a final example, let us consider the following more "tricky" query: *Find all images which are attached to some emails while the similarity between the images and the sample image is larger than 50%.*

In MediaLand, this query is specified as a combination of the following operations:

Graph_Match (x →(attached) y) I_{y} **Class** ("Email")

I_{x} **Convert** (**ThresholdSearch** (sample_image, 0.5) ∩ **Class** ("Image"))

The result of **Graph_Match** (X →(attached) Y) is a relation with schema (x:OID, y:OID). Then this relation is intersected with the email object set and qualified image object set.

5. Related Work

Multimedia data in the form of image, graphics, text, video, and audio possess properties that are not adequately supported by traditional database systems, such as large data size, time-dependent nature, content-based retrieval, and the demands on quality of service, etc. To address such limitations, during the last decade multimedia databases have been proposed and received an extensive study on its related techniques. A main objective is to provide reliable and efficient storage, maintenance, and access of different types of multimedia. Substantial modifications and extensions have been made to both relational databases (e.g. STARBURST system [11]) and object-oriented databases (e.g. [19] [20]) to include the "multimedia features". Many emerging multimedia systems, such as QBIC[2] and Informedia[12], manifest certain properties being characteristic of a multimedia database. Nevertheless, existing multimedia databases have not achieved the ultimate goal of providing an integrated environment for managing different media objects uniformly. As a matter of fact, many related problems have not been successfully addressed, particularly from the perspective of unified modeling of media objects and cross media retrieval. To the best of our knowledge, MediaLand is the first multimedia database which actually reaches to the extent of providing a uniform modeling framework to data objects of *all* the media types.

Information retrieval [22], content based retrieval [24], database querying [23], graph and tree matching [17] [25] [26] are four distinct research fields traditionally. Recently, there are some efforts to extend one approaches to handle other date types, such as integrating keyword based search into database systems [4][6][7][8], combining keywords with features in content-based retrieval systems[15], adding XML support into relational database systems [17], adopting XML to describe and access visual-audio data [13] etc. Although previous work has demonstrated some appealing results, still there lacking of a solid theory and system to provide a uniform framework for modeling, managing and retrieving various kinds of multimedia data. In contrast, MediaLand provides a multi-paradigm querying methodology by integrating the relevant techniques from database approach, information retrieval, content-based retrieval, and hypermedia (graph) modeling in a seamless manner.

6. Conclusion and future work

In this paper we have presented a multi-paradigm querying approach of MediaLand, an experimental multimedia database (MMDB) system being developed at Microsoft Research Asia. The main objective of MediaLand is to truly meet the requirements of MMDB management, by providing an integrated framework for modeling, managing and retrieving various kinds of media data in a uniform way. To suupport MMDB users who can be of different levels of experiences and expertise, a multi-paradigm query engine is devised upon MediaLand's 4-tier architecture to process cross media retrieval via a seamless integration of various existing search approaches. As a by-product, MediaLand can support the notion of "media independence" which is analogous to the concept of "data independence" from the classic ANSI SPARC standard.

MediaLand is an ongoing project, and there are a number of important issues remaining to be addressed in our subsequent research:

1) Query optimization

Since approximate search operations (especially those for content base retrieval and graph matching) are usually very time-consuming, optimization is very critical to the performance of query processing. In MediaLand, the essence of query optimization is to give an optimized execution order of the operations. A proper order can effectively reduce the cost of some expensive operations. Therefore, the query optimization in MediaLand is very different from those in relational database systems (more appropriately, it can be termed as "meta-optimization").

2) User interface facilities

Besides a "descriptive" query language, versatile user interface facilities are needed to be developed, which can support visual presentation of the query results

dynamically, incremental/iterative user feedbacks interactively, and the whole range of administrative/editing operations completely.

3) Performance Evaluation

Last but not the least, we also plan to work on issues related to performance evaluation. As queries in MediaLand can be both precise and imprecise, appropriate cost model functions need to be developed in order to evaluate the effectiveness of the various query optimization and indexing techniques. Also, a "ground-truth" database needs to be chosen and bench-mark queries need to be defined upon which the performance evaluation can be conducted more objectively.

Acknowledgement: The authors would like to express their sincere thanks to Prof. Hongjun Lu for a fruitful discussion on some issues/aspects related to this project, which also helped the presentation of this paper.

Reference

[1] Abiteboul, S., Querying Semi-structured Data, Proceedings of the International Conference on Database Theory, 1997.

[2] Apers, P. M. G., Blanken, H. M., Houtsma, M. A. W., Multimedia Databases in Perspective, Springer-Verlag,1997.

[3] Chaudhuri, S. and Gravano, L., Optimizing Queries over Multimedia Repositories, Proc. of ACM SIGMOD'96, Montreal, Canada, 1996, pp. 91-102.

[4] Dessloch, S. and Mattos, N., Integrating SQL Databases with Content-specific Search Engines. Proceedings of the 23rd VLDB conference. Athens, Greece, 1997.

[5] Fagin, R., Fuzzy Queries in Multimedia Database Systems. Proceedings of the 17th ACM symposium on Principles of database systems (PODS '98), June 1 - 4, 1998, Seattle, WA.

[6] Florescu, D., Kossmann. D. and Manolescu, I., Integrating Keyword Search into XML Query Processing. Proceedings of the 9th WWW conference. Amsterdam, NL, May 2000

[7] Fuhr N., Models for Integrated Information Retrieval and Database Systems. IEEE Data Engineering Bulletin 19(1), pp. 3-13.

[8] Goldman, R., Shivakumar, N., Venkatasubramanian, S. and Garcia-Molinas, H., Proximity search in databases. In Proceedings of the 24th VLDB Conference, 1998

[9] Gyssens M., Paredaens J., Van den Bussche J. and Van Gucht D., A Graph-oriented Object Database Model. IEEE Trans. on Knowledge and Data Eng., 6(4), 572-586, 1994.

[10] G tzer, U., Balke, W. and Kie ing, W., Optimizing Multi-Feature Queries for Image Databases. Proceedings of the 26th VLDB conference, pp. 419-428, Cairo, Egypt, 2000.

[11] Haas, L. M., "Supporting Multi-Media Object Management in a Relational Database Management System", Technical report. IBM Almaden Research Center, 1989.

[12] Hauptmann, A., Smith, M., "Text, speech, and vision for video segmentation: The Informedia project", In *AAAI Fall 1995 Symposium on Computational Models for Integrating Language and Vision,* 1995.

[13] Jos M. Mart ez (Editor), Overview of the MPEG-7 Standard. March 2001, http://www.cselt.it/mpeg/standards/mpeg-7/mpeg-7.htm

[14] Lee, J. H., Analyses of Multiple Evidence Combination. Proceedings of the 20th international ACM SIGIR conference. July 27 - 31, 1997, Philadelphia, PA.

[15] Lu, Y., Hu, C., Zhu, X., Zhang, H. and Yang, Q., A Unified Semantics and Feature Based Image Retrieval Technique Using Relevance Feedback. ACM MULTIMEDIA 2000--The 8th ACM International Multimedia Conference, Los Angeles, California , October 30 - November 3, 2000.

[16] Manmatha, R., Rath, T. and Feng, F., Modeling Score Distributions for Combining the Output of Search Engines. Proceedings of the 24th international ACM SIGIR conference. Sept. 9-13, 2001, New Orleans, LA.

[17] Mart ez, C. and Valiente, G. An Algorithm for Graph Pattern-Matching. In Proc. Fourth South American Workshop on String Processing, volume 8 of International Informatics Series, Carleton University Press (1997), pp. 180-197.

[18] Microsoft Corp., Microsoft SQL Server 2000.

[19] Oomoto, E., Tanaka, K., "OVID: Design and Implementation of a Video-Object Database System", *IEEE Transactions on Knowledge and Data Engineering*, vol.5, pp 629-641, 1993.

[20] Orenstein, J. A., "A Comparison of Spatial Query Processing Techniques for Native and Parameter Spaces", In *Proc. of ACM SIGMOD Conf.* pp 343-352, 1990.

[21] Rakow, T., Neuhold, E. J. and Lohr, M., Multimedia Database Systems - The Notions and the Issues. In G. Lausen, editor, Tagungsband GI-Fachtagung Datenbanksysteme in Buro, Technik und Wissenschaft (BTW), Dresden Marz 1995, pp. 1--29. Springer Verlag, Informatik Aktuell, 1995.

[22] Salton, G. and McGill, M.J., Introduction to Modern Information Retrieval, McGraw-Hill Book Company, 1983.

[23] Ullman, J. D., Principles of Database Systems. Computer Science Press, 2 edition, 1982.

[24] Zhang, H. J. et al, "Video parsing and browsing using compressed data" Multimedia Tools and Applications 1(1), 89-111, 1995

[25] Zhang K., Shasha D. and Wang, J., Approximate Tree Matching in the Presence of Variable Length don't Cares. *Journal of Algorithms*, 16(1):33-66, January 1994.

[26] Zhang, K., Wang, J.T. and Shasha, D. On the Editing Distance Between Undirected Acyclic Graphs. International Journal of Foundations of Computer Science 7(1): 43-58 (1996).