IDEA GROUP PUBLISHING

701 E. Chocolate Avenue, Suite 200, Hershey PA 17033, USA
Tel: 717/533-8845; Fax 717/533-8661; URL-http://www.idea-group.com

ITP4498

A Structure- and Content-based Multimedia Information Retrieval System for XML Documents

Du-Seok Jin, Jeong-Jae Lee, and Jaewoo I. Chang
Dept. of Computer Engineering, Chonbuk National University, Chonju, Chonbuk 560-756, South Korea, {dsjin, jjlee, jwchang}@dblab.chonbuk.ac.kr

ABSTRACT

Because the number of XML documents is dramatically increasing, we need to develop a multimedia information retrieval system which can support both the retrieval based on document structure and the retrieval based on image content. In order to support the structure-based retrieval, we design keyword, structure, element, and attribute index structures by indexing XML documents based on the basic element unit and implement them by using the o2store storage system. For supporting the content-based retrieval, we design and implement high-dimensional index structures for both color and shape features based on X-tree. Finally, we do the performance evaluation of our multimedia information retrieval system in terms of system efficiency, such as retrieval time, insertion time, and storage overhead.

INTRODUCTION

In 1996, the World Wide Web (Web in short) Consortium proposed XML (eXtensible Markup Language) as a standard markup language to make Web documents [1]. The XML has as good expressive power as SGML and is also easy to use like HTML. Since then, because the number of XML documents is dramatically increasing, it is difficult to reach a specific XML document required by users. Meanwhile, an XML document not only has a logical and hierarchical structure commonly, but also contains its multimedia data, such as image and video. Thus, it is necessary to develop a multimedia information retrieval system that can support both the retrieval based on document structure and the retrieval based on image content.

In general, since the conventional information retrieval systems for XML documents support only structure-based retrieval, it is impossible to efficiently deal with a user query which requires XML document retrieval based on both document structure and image content. In this paper, we design and implement a multimedia information retrieval system that can efficiently retrieve XML documents based on both document structure and image content. In order to support the structure-based retrieval, we design four efficient index structures, i.e., keyword, structure, element, and attribute, by indexing XML documents based on the basic element unit and implement them by using the o2store storage system. For supporting the content-based retrieval, we design and implement high-dimensional index structures for both color and shape features based on X-tree.

PAPER OVERVIEW

This paper consists six sections. Section three explains related work in the area of structure-based and content-based information retrieval. A structure- and content-based multimedia information retrieval system is designed in Section four. Section five presents the performance evaluation of our system in terms of system efficiency, followed by conclusions and some issues for future research in Section six.

RELATED WORK

Since there has been many researches on SGML information retrieval with SGML documents and the techniques for SGML information retrieval can be directly applied to XML documents, we, in this section, describe some related work on the representation of SGML document structures. First, RMIT in Australia proposed five query types for structure-based retrieval which should be supported in SGML information retrieval [2]. Most of the types consist of retrieval for upper-level elements (e.g., parent element) or lower-level elements (e.g., child elements) from a given element. For supporting the five types of queries, RMIT proposed a *subtree model* which indexes all the elements in a SGML document and stores all the terms appeared in the elements [3]. Although the model supports efficient retrieval for a specific query, it has disadvantages of long indexing time and high storage overhead because index information should be repeatedly stored. Secondly, RMIT proposed a *SCL structure* that extends the *GCL structure* [4]. After assigning numbers to both terms and markups in SGML documents, they use the *SCL structure* to store term interval, markups and inclusion relationships among elements. The *SCL structure* has an advantage that it can handle graph-structured documents, but it has an disadvantages that it cannot represent the depth of the elements effectively. Finally, SERI in South Korea proposed a *K-ary Complete Tree Structure* which represents a document as a K-ary complete tree. In this method, each element corresponds to a node in a K-ary tree [5]. Therefore, a relationship between two elements can be acquired by computation. This method has an advantage that it is fast to find an element including a given logical relation by calculation. But, as the depth of a K-ary tree is deeper, the number of nodes is increasing exponentially with a large number of unused nodes. In addition, the insertion and deletion of a node causes the change of the assigned numbers of all the other nodes in the tree.

The key issues of studies on content-based retrieval include image processing techniques used for feature extraction of images, and high dimensional indexing structures for fast retrieval, and content-based image retrieval based on color histogram, texture, and shape. First, *QBIC(Query By Image Content) project* [6, 7] of

IBM Almaden research center studied content-based image retrieval on a large on-line multimedia database. The study supports various query types based on the visual image features such as color, texture, and shape. Secondly, VisualSEEk of the Colombia University developed a tool for content-based retrieval and browsing. It deals with user queries which combine a spatial location of image object and color [8]. Finally, the *CORE (Content-base Retrieval Engine)* [9] of the National University of Singapore studied novel indexing techniques based on image features so that the engine provides content-based retrieval on multimedia objects. The engine provides the functionality of query feedback to support query refinement.

A STRUCTURE- AND CONTENT-BASED MULTIMEDIA INFORMATION RETRIEVAL SYSTEM

A structure- and content-based multimedia information retrieval system mainly consists of four components, i.e., structure-based indexing part, content-based indexing part, storage manager part, and unified retrieval part. Figure 1 shows the whole system architecture of the structure- and content-based multimedia information retrieval system. When a multimedia XML document is given, we first parse the XML document and perform image segmentation from the document by using an image preprocessing algorithm. The parsed document structure information is transported into its parsing tree in order to index its document structure consisting of element units. The parsed image content information is also transported into its content-based information in order to get the index information of its color and its shape. The structure-based and content-based index information is separately stored into its own index structures, respectively. Using the index information extracted from a set of documents, some documents are retrieved by the unified retrieval part in order to answer user queries, through World Wide Web (WWW).

Document Structure-based Indexing

Because an element is a basic unit for retrieving XML documents, it is required to support not only retrieval based on document unit, which is used in the traditional information retrieval system, but also retrieval based on element unit. For this, we design an information retrieval system which can efficiently support query by element, through an index on document structure after analyzing XML documents. Suppose XML documents including Korean porcelain information. To make a document structure tree for XML documents, we first parse the XML documents by using sp-1.3 XML parser [10]. Next, we construct the document structure tree from the parsed result. Then the constructed tree is delivered into a low-level storage manager. Finally the storage man-

Figure 1. System Architecture

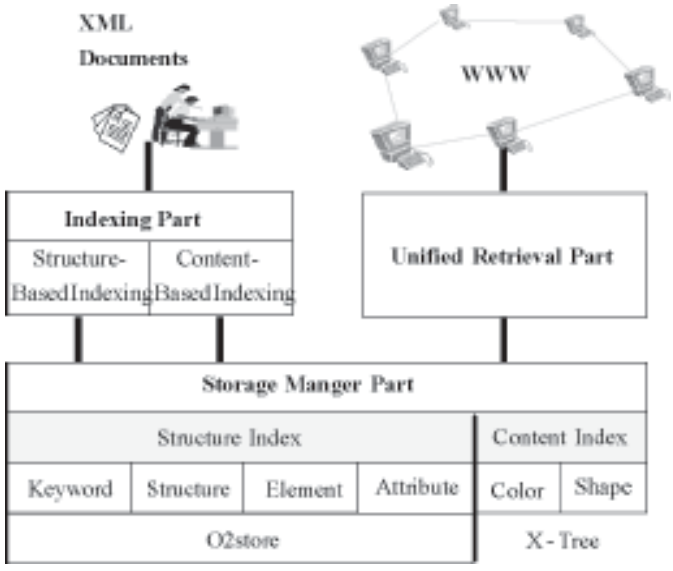


Figure 2. A procedure to construct document structure tree

XML Docum

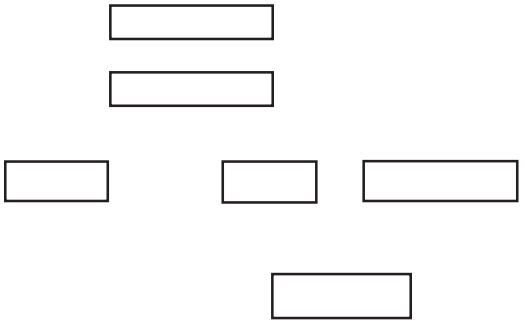
Figure 3. An example of XML Document

```
<relic>
    <porcelain TYPE= “Chong-Ja">
     <name>
        Chong-ja kettle
     </name>
     <decoration>
        btusfbwer
    </decoration>
                    :
     <detail>
          It is Chong-ja kettle in little gourd shape ...
                    :
      <detail>
     </description>
     <image SRC='hc_1">
     </image>
   </porcelain>
</relic>
```

Figure 4. Parsed data of XML document

?xml versior
(RELIC
ATYPE CD
(PORCELA
(NAME

Figure 5. Document structure tree



ager extracts document structure information and image content information from the tree information, and stores them into a database. Figure 2 shows a transformation procedure to make a document structure tree and Figure 3 and 4 show an example of XML documents and its parsed data, respectively. Figure 5 describes the document structure tree being constructed.

Image Content-based Indexing

For image content-based retrieval, we analyze image objects of XML documents and extract image feature vectors by separating object regions from the background of images. To extract the object regions, we use the fuzzy c-mean (FCM) algorithm which is a famous clustering algorithm to divide object regions from color images [11]. The algorithm has an advantage that the separation of image objects from its background can be performed well when an image has little noise, as the case of our porcelain images. In order to obtain an image feature vector for shape, we use the image object being produced by the image preprocessing algorithm and generate a 24-dimensional feature vector based on distances between the center point and a set of edge points. An algorithm for generating a shape feature is as follows.

a) After sorting each pixel of object in a column and a row respectively, calculate the center point of object using its maximum and minimum values.

Figure 6. Color and shape feature vector extraction

b) By increasing 15 degrees at the central point, starting from the X-axis, select 24 pixel points met at the edge.
c) Compute the distance between the central point and the 24 pixel points on the edge.
d) Normalize the 24 distances by dividing them by the maximum distance.
e) Generate a 24-dimensional feature vector.

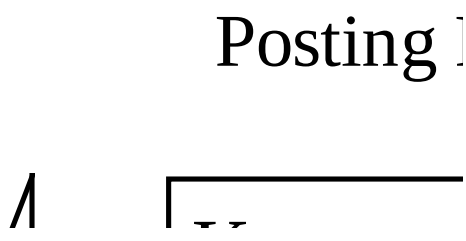
Since the proximity among colors in the RGB color space never means their similarity among colors, we use HSV (Hue, Saturation, Value) color space model which can provides an uniform distribution of colors. In this model, H means an aggregate of color, ranging from 0 to 360 degree. S means the chroma of color and V means the brightness of color. An algorithm to generate a 22-dimensional color feature vector is as follows. Figure 6 shows an example in which both 24-dimensional shape feature vector and 22-dimensional color feature vector are extracted from its real image.

a) Transform all color pixels of an image object in the RGB color space into those in the HSV color space.
b) Generate a color histogram by using color histogram generation algorithm.
c) Normalize the color histogram by dividing it by the number of all the pixel.
d) Generate a 22-dimensional feature vector.

Low-level Storage Manager

The low-level storage manager consists of two parts, i.e., index structures for document structure-based retrieval and index structures for image content-based retrieval. The index structures for structure-based retrieval are composed of keyword index, structure index, element index, and attribute index by indexing XML documents based on element unit, i.e., the basic unit of XML documents. The keyword index consists of three files, i.e., keyword index file being composed of keywords extracted from data token element (e.g., PCDATA, CDATA) of XML documents, posting file including the IDs of document and element where keywords appear, and location file containing the location of keyword occurrence in elements. Figure 7 describes the keyword index structure and shows an extendible form of an index structure used for the conventional information retrieval system by adding element information. DF (document frequency) represents the number of documents containing a given keyword. DID is the identifier of the document stored. Because a keyword can be appeared in a set of elements constituting a document, the index includes the different value of EF (element frequency) per each document where the EF means the number of elements having a given keyword in each document. Oid is the identifier of the element stored. Eid is an identifier for an element name. Here we use Eid (element name ID) rather than the actual element name because the element name being variable in size is appeared repeatedly. TF (term frequency) represents the number of keyword occurrences in an element. locId is the identifier of location information, i.e. paragraph (P),

Figure 7. Keyword index structure



sentence(S), and word(W), in the location file.

Since the structure index is used for searching an inclusion relationship among elements, it should represent the logical structure of a document and guarantee good performance on both retrieval time and storage overhead. For this, we propose an element-unit parse tree structure to represent the hierarchical structure of a document. In the structure, we can easily find an inclusion relationship among elements because an element contains the location of its parent, its left sibling, its right sibling, and its first left child. Figure 8 shows the structure index structure based on the element-unit parse tree where an element is identified by Oid and Eid. For fast searching of inclusion relationship among elements, we makes use of the identifier of parent element (ParentOid), the identifier of left sibling element (LsiblingOid), the identifier of right sibling element (RsiblingOid), and the identifier of the first left child element (FchildOid). In addition, NW(node weight) represents relevance weighting degrees between a parent node and one of its child nodes. It is computed as the similarity between a parent term vector and its child term vector. The element index is used for locating a start element. It also plays an important role in mapping it into an actual element name from the Eid of the element obtained from the content index or the attribute index. The attribute index is used for retrieval based on an attribute name and an attribute value of an element. When a user query requires keyword search with structure information, its storage manager first finds Oids concerned with the keyword in the keyword index, and then extracts proper Oids to find Eid, ParentOid, LsiblingOid, RsiblingOid, FchildOid in the structure index.

As the number of dimensions of feature vectors is increasing for image content-based retrieval, the retrieval performance of the traditional index structures is exponentially increasing. To cope with this problem, we construct color and shape index structures using the X-tree [12]. The X-tree index structure was proposed as an efficient high dimensional index structure to achieve good retrieval performance even though the dimension of feature vectors is high. The X-tree makes use of special super nodes being extended in size so as to minimize the size of overlapped region due to node splitting. Thus, our color index and shape index structures can store high-dimensional feature vectors and can support image content-based retrieval efficiently. Figure 9 shows both color and shape index structures constructed using X-tree.

Figure 8. Structure index structure

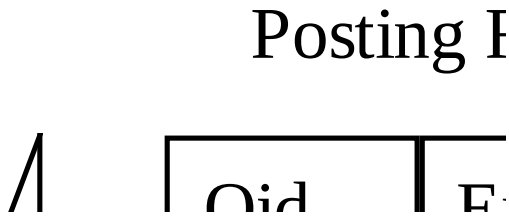
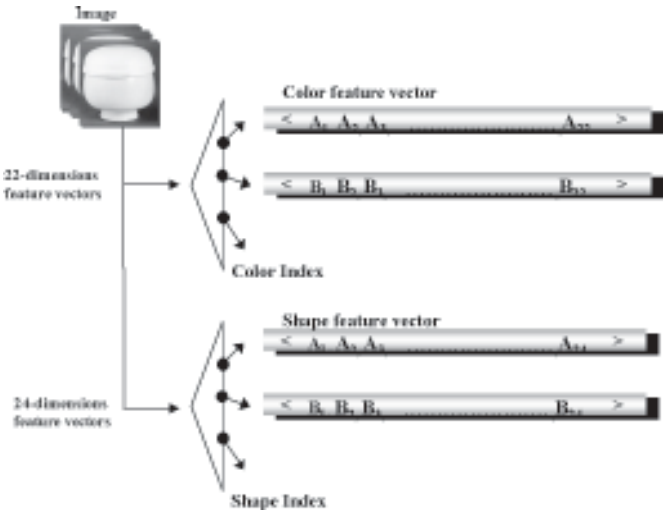


Figure 9. Color and shape index structures using X-tree



Unified Retrieval

Even though there have been various retrieval models in the traditional text-based information retrieval, there is little research on retrieval models for both structured- and content-based multimedia information retrieval. In this section, we can specially treat two types of user queries, i.e., document structure query and image content query. To answer the document structure query, we first search the structure index and offer proper results for it. In a document structure query, a similarity (S_w) between an element q and an element t is computed as the similarity between the term vector of node q and that of node t, as shown in the following equation [13].

$$S_w = COSINE(NODE_q, NODE_t) = \frac{\sum_{k=1}^{m}(TERM_{qk} \cdot TERM_{tk})}{\sqrt{\sum_{k=1}^{m}(TERM_{qk})^2 \cdot \sum_{k=1}^{m}(TERM_{tk})^2}}$$

When results for a user query are documents, the documents can be represented as D = { E_0, E_1, …., E_{n-1} } where E_i means an element i in a document D. Also, a similarity (D_w) between an element q and a document D is computed as follows.

$$D_w = \text{MAX}\ \{ COSINE(NODE_q, NODE_{E_i}),\ 0 \le i \le n-1\}$$

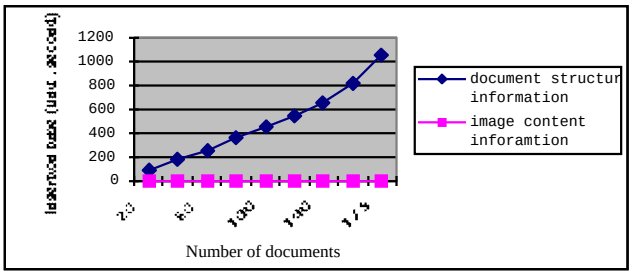
Secondly, to answer an image content query, we first extract color or shape feature vectors from a given query image. Then we compute Euclidean distances between a query color (or shape) feature vector and the stored image color (or shape) feature vectors by searching the color (or shape) index. Finally we compute a similarity between the query feature vector and the image feature vector as 1 – (Euclidean distance / maximum distance), and retrieve relevant documents with high similarity in the decreasing order of the similarity. In case we require content-based retrieval based on both color and shape feature vectors, we compute its color distance and its shape distance separately and integrate them into one unified similarity by the multiplication of both the color and the shape weight. A similarity, $C_W(q, t)$, between a query image q and a target image t in the database is calculated as the following equation.

$$C_w = \begin{cases} 1 - \dfrac{Distc(q,t)}{Nc}, & \text{if a query contains only a color feature.} \\ 1 - \dfrac{Dists(q,t)}{Ns}, & \text{if a query contains only a shape feature.} \\ (1 - \dfrac{Distc(q,t)}{Nc}) \times (1 - \dfrac{Dists(q,t)}{Ns}), & \text{if a query contains both color and shape feature.} \end{cases}$$

$$T_w = \begin{cases} C_w \times 0.5 + D_w \times 0.5, & \text{if results are document for user query} \\ C_w \times 0.5 + S_w \times 0.5, & \text{if results are element for user query} \end{cases}$$

Here Distc(q, t) and Dists(q, t) mean a color vectors distance and a shape vector distance between a query image q and a target image t, respectively. Nc and Ns mean the maximum color and the maximum shape distances for normalization, respectively. Finally, when the weight of the structured-based retrieval is equal to that of the content-based retrieval, i.e., 0.5, a similarity (T_w) for both structured- and content-based composite query is calculated as the following equation.

Figure 10. Insertion time



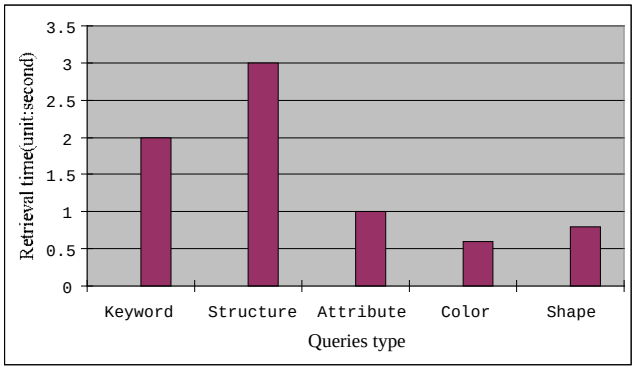
PERFORMANCE EVALUATION

We implement our structure- and content-based multimedia information retrieval system under SUN SPARCstation 20 by using GNU CCv2.7 compiler. For this, we make use of O2 OODBMS v4.6 as a storage system and Sp-1.3 as an XML parser. To evaluate our system efficiency, we measure retrieval time, insertion time, and storage overhead. Table 1 shows a test data set used for the performance evaluation.

The insertion time for a multimedia document is shown in Figure 10. The insertion time for the structured information means one for adding a document into keyword, attribute, structure, and element index. The insertion time for the content information means one for adding an image into color and shape index. The size of index files is growing as the number of documents is increasing, and the search time of index files is in proportion to the size of the index files. That is to say, it takes less than 1 second to insert an image content information into the color and shape indexes. But, it takes about 6 seconds to insert the structured information of an XML document into the four indexes. Figure 11 shows retrieval times for queries base on keyword, attribute, structure, image color, and image shape. The retrieval times for the attribute, color, shape queries are less than 1 second. The retrieval time for the keyword query is about 2 second and that for the structure query is about 3 seconds. Consequently, it is shown that the time for answering the structure query is the longest.

We measure storage overhead as a ratio of the total size of our index files to that of the XML documents. Table 2 shows the storage overhead of our information retrieval system which is computed as the division of the size of all index files by the size of the original document. It is shown that our system requires about 50% storage overhead.

Figure 11. Retrieval time



CONCLUSIONS

In this paper, we designed and implemented a multimedia information retrieval system that can efficiently retrieve XML documents based on both document structure and image content. In order to support efficient structure-based retrieval, we designed our keyword, structure, element, and attribute index structures by indexing XML documents based on the basic element unit and implemented them by using the o2store storage system. For efficient image content-based retrieval, we designed and implemented high-dimensional index structures for both color and shape features based on X-tree. In our structure- and content-based multimedia information retrieval system, the retrieval time for the structure query is about 3 seconds while that for the image content query is below 1 second. Our information retrieval system spent about 6 seconds for inserting a document and required about 50 % storage overhead. As a further research, we need to implement our information retrieval system by using such a public storage system as the Shore storage system developed by the University of Wisconsin, leading to the popularity of our system.

ACKNOWLEDGMENTS

This research has been supported by ‘Outstanding information & communication school project’ of Korean Ministry of Information & Communication.(grant no. 98-74)

REFERENCE

[1] eXtensible Markup Language(XML), http :// www.w3.org/TR/PR-xml-971208.
[2] R. Sack-Davis, T. Arnold-Moore and J. Zobel, “Database Systems for Structured Documents,” In Proc. International Symposium on Advanced Database Technologies and Their Integration, 1994.
[3] B. Lowe, J. Zobel and R. Sacks-Davis, “A Formal Model for Databases of Structured Text,” In Proc. Database Systems for advanced databases, pp. 449-456, 1995.
[4] T. Dao and R. Sacks-Davis, “Indexing Structured Text for Queries on Containment Relationships,” In Proc. the 7th Australian Database Conference, Melbourne, 1996.
[5] Sung-Geun Han, Jeong-Han Son, Jae-Woo Chang and Zong-Cheol Zhoo, “Design and Implementation of a Structured Information Retrieval System for SGML Documents,” In Proc. Database Systems for Advanced Applications, pp. 81-88, 1999.
[6] W. Niblack, et. al., “The QBIC project: Querying by Image Content Using Color, Texture, and Shape,“ In Proc. SPIE Storage and Retrieval for Image and Video Databases, pp.173-187, 1993.
[7] M. Flickner, et. al., “Query by Image and Video Content: The QBIC System,” IEEE computer, pp. 23-32, Sep. 1995.
[8] J. R. Smith, S. F. Chang, “VisualSEEK: a Fully Automated Content-Based Image Query System,” ACM Multimedia Systems, Vol. 4, Nov 1996.
[9] J. K. Wu, et. al., “CORE: a Content-based Retrieval Engine for Multimedia Information Systems,” ACM Multimedia Systems, Vol. 3, No. 1. pp 25-41, 1995.
[10] http://www.jclark.com/sp.
[11] J. C. Bezdek and M. M. Triedi, “Low Level Segmentation of Aerial Image with Fuzzy Clustering,” IEEE Trans. on SMC, Vol.16, pp. 589-598, 1986.
[12] S. Berchtold, D. Keim, and H. –P. Kriegel, “The X-tree: An Index Structure for High-Dimensional Data,” In Proc. the 22nd Conf. on Very Large Databases, 1996.
[13] Salton, G., and M. McGill, “An introduction to Modern Information Retrieval,” McGraw-Hill, 1983.