The XML Query Execution Engine (XEE)

Dietl Scheffler

Motivation

XML has become widely accepted for both data representation and exchange of information over the Internet. For this reason, the amount of data in XML format is rapidly growing, making appropriate systems for their storage and retrieval necessary.

The objective of the XML Query Execution Engine (XEE) is to integrate web-based database and information retrieval technology to improve the storage, retrieval, and access of large XML document collections, in particular with respect to updates.

The XEE system provides an experimental testbed for the Access Support Tree & TextArray data structures designed at DBIS for storage, retrieval, and update purposes of XML documents. The basic idea of the Access Support Tree & TextArray data structure is to separate the (logical) structure of a document from its "visible" text content.

Access Support Tree & TextArray (AST/TA)

Access Support Tree

An Access Support Tree is an ordered tree \( AST = (V, E) \). Properties of vertices \( v \in V \):

- Type \( \pi(v) \in \{ \text{element}, \text{PI, Comment, Character, CDATA}, \text{Enlitity} \} \)

- Label \( \lambda(v) \in \{ \text{element names}, \text{PI}, \text{Comments plus context} \} \)

- Attributes \( \alpha(v) \in \{ \text{character, CDATA, EntityLit} \} \imes \lambda(v) \)

- Text surrogate \( s(v) \in \{ \text{position, length} \} \)

- Position \( \text{beginning of the text segment referenced by } s \) in TA

- Length of the text segment referenced by \( s \) in TA

- Offset \( \text{offset} \) adjusts the position in text surrogate if it changes after updates.

Relationships between vertices \( v \in V \):

- Edge \( e \in E \) \( \times \) parent child relationship

- Parent vertex \( \pi(v) \): \( \text{parent}(v) \in \{ \text{element}, \text{PI, Comment, Character, CDATA}, \text{Enlitity} \} \)

- Child vertex \( i \): \( \text{child}(v, i) \in \{ \text{element names}, \text{PI}, \text{Comments plus context} \} \)

TextArray

A TextArray is the continuous character sequence \( \sigma \) of the text content of an XML document in document order.

TextArray supports two different views on text content:

- Text as a sequence of characters

- Text as a sequence of words

In total, this is a sequence of words that requires text analysis. For this purpose, the TextArray has two different separations: separation (1) non-separating, (2).

System Requirements

(1) Querying XML documents must integrate both the concept of a query language and the concept of information retrieval.

(2) The idea of separating the layout from the structure and content of a document is extended to the separation of structure and content, thus taking over the basic DBMS concept of partitioning data and meta data.

(3) The structure of documents to be stored is not necessarily constrained by any schema. That is, the system must be able to store such generic documents even if they do not provide any DTD or any other schema information.

(4) Efficient operations on documents must be supported, especially while updating both the structure and the content of documents.

Approach: Scoring structure and content separately

XML document mapped into AST/TA data structures

Structure (tags) and textual content of XML documents

Persistant Access Support Tree & TextArray (AST/TA)

Sample XML tree

Persistant AST corresponding to the sample XML tree

The page offset concept of a Persistant AST

The Offset Dictionary

AST page layout

Persistent TextArray

Persistent TextArrays are implemented as persistent B+ trees. The leaves of such a tree represent the text content of the corresponding document.

XEE is an experimental XML Management System

The XEE system:

- provides a testbed for the storage format of the AST/TA data structure,

- enables the investigation of the mapping between the AST/TA data structure and various query models,

XEE is a Model-Based Native XML Database

The XEE system:

- is especially designed to store XML documents,

- stores generic XML documents, which do not require any DTD or schema,

- provides an internal object model for documents and assists this model for experimental testing. XEE implements the Document Object Model, the XQuery 1.0 and XPath 2.0 Data Model, and the Proximal Node Model,

- is based on a proprietary storage format—the AST/TA data structure.