Explorative Analysis and Visualization of Large Information Spaces

Christian Grün

1 Background

• very large XML documents exist: Wikipedia, library catalogs, DBLP, protein data, ...
• databases are needed to store and efficiently query XML data
• visualizations facilitate interactive browsing and searching

2 Research Questions

Observations

• effective techniques have been developed for relational storage of XML data, such as the XPath Accelerator [3]
• native XML databases may further optimize support of XML/XPath/XQuery specific operations

Questions

• can we port the relational approach to native databases?
• can we add indexes to speedup content-based queries?
• can we utilize the storage patterns to build interactive visualizations on mega- and giga-bytes of XML data?

3 BaseX

• native XML database with GUI interface
• supports XML documents > 50 GB
• features hierarchical visualizations
• allows for realtime querying and browsing
• open source since 03/2007 (www.basex.org)

Storage [1]

XML Document

<address id="address0">
  <name>Giovanni Stanuti</name>
  <street>Via Pellaro 27</street>
  <code>80142</code>
  <town>Napoli</town>
</address>

Table Representation

PRE DIST DSC ATS KIND CONTENT
0 1 9 1 ELEM address
2 1 0 ELEM street
3 2 1 0 ELEM name
4 1 0 TEXT Giovanni Stanuti
5 4 1 0 ELEM street
6 1 0 TEXT Via Pellaro 27
7 6 1 0 ELEM code
8 1 0 TEXT Napoli
9 8 1 0 ELEM town
10 1 0 TEXT Napoli

XML Tree

Disk Storage

Table:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>address</td>
<td>address0</td>
</tr>
<tr>
<td>name</td>
<td>Giovanni Stanuti</td>
</tr>
<tr>
<td>street</td>
<td>Via Pellaro 27</td>
</tr>
<tr>
<td>code</td>
<td>80142</td>
</tr>
<tr>
<td>town</td>
<td>Napoli</td>
</tr>
</tbody>
</table>

Text:

- Giovanni Stanuti
- Via Pellaro 27
- Napoli

• modified XPath Accelerator encoding, including
  - distances to parent nodes
  - number of descendants and attributes
• inlining of attributes
• compressed disk storage of table rows:
  - omission of default values
  - inline numeric values
• block-based storage to support updates
• text, attribute and fulltext indexes

Querying

• support of W3C XPath 2.0 and XQuery 1.0 recommendations (99.7% XQTS conformance)
• support of W3C XQuery Full-Text 1.0 Working Draft
• partial XQuery Update support

Queries are optimized and rewritten to access index structures:

Query:

//country[religions = 'Muslim']

Result:

path(IndexAccess(TXT, 'Muslim'), parent::religions/parent::country)

Visualizations [2]

• fast map, tree and table view algorithms, especially tailored for utilizing the compact disk storage
• on-the-fly execution and filtering of keyword, XPath and XQuery requests
• XQuery editor with syntax & error highlighting
• real-time interaction via multiple coordinated views

Literatur

