XML Data in (Object-) Relational Databases

RNDr. Irena Mlýnková
irena.mlynkova@mff.cuni.cz

Charles University
Faculty of Mathematics and Physics
Department of Software Engineering
Prague, Czech Republic
Content

1. Introduction
2. Analysis of Related Work
3. Hybrid User-Driven Adaptive Method
4. Similarity Function
5. Statistical Analysis of Real-World XML Data
6. Query Evaluation
7. Conclusion
8. Summary
Motivation

- **XML** = a standard for data representation and manipulation
  ⇒ Growing demand for efficient managing and processing of XML data

- **Current approaches**
  - **File system**
    - Inability of querying without additional data pre-processing
  - **Pure object-oriented approach**
    - No efficient and comprehensive tool
  - **Native methods**
    - No need to adapt structures to a new purpose ⇒ most efficient
  - **(O)RDBMS**
    - Mature and verified technology ⇒ most practically used
Database-Based XML Processing Methods

Key concern: Choice of the optimal XML-to-relational mapping

- How XML data are stored into relations
- Exploitation of various types of supplemental information
  - XML schema, sample XML documents, expected query workload, user interaction, etc.
- **Generic** vs. **schema-driven** – omitting / exploiting XML schema
- **Fixed** vs. **adaptive** – the amount of input data
  - Data model vs. sample XML documents and XML queries
- **User-defined** vs. **user-driven** – the amount of user involvement
  - User defines both schema and mapping vs. user specifies local changes of a default mapping
    - User-driven: schema is adapted to the annotations
- Which of the XML-to-relational mappings is the best? Can the existing approaches be enhanced? If so, how?
Outline of the Thesis

1. Analysis of related work
   • Classification and evaluation of existing approaches
   • Identification of open problems and possible solutions
2. Proposal of a hybrid user-driven adaptive method
   • Solution of several identified open issues
3. Proposal of similarity function
   • Schema-level structural similarity
   • Tuning of weights of the function
   • Exploitation of results of analysis of real-world data
4. Statistical analysis of real-world XML data
   • New findings, detailed characteristics of real-world data
5. Query evaluation over resulting system
   • Correction of the set of annotations, types of annotations
   • Problems related to query evaluation
Content

1. Introduction
2. Analysis of Related Work
3. Hybrid User-Driven Adaptive Method
4. Similarity Function
5. Statistical Analysis of Real-World XML Data
6. Query Evaluation
7. Conclusion
8. Summary
Adaptive Methods

- Not a straightforward mapping, adapt to a current application
- **Cost-driven**
  - Choose the most efficient storage strategy automatically
  1. Search a space of possible mappings of initial schema $S_{init}$
     - Set of XML-to-XML schema transformations $T = \{t_1, t_2, ..., t_n\}$
  2. Choose the optimal one for given sample
     - XML documents $D = \{d_1, d_2, ..., d_k\}$ valid against $S_{init}$
     - Query workload $Q = \{q_1, q_2, ..., q_l\}$ over $S_{init}$
- **User-driven**
  - Infinite space of mappings $\Rightarrow$ approaches differ in search heuristics
  - Optimization of user-defined methods
  - User can influence default fixed mapping $f_{def}$ of $S_{init}$ using a set of annotations $A$
    - Predefined set of fixed XML-to-relational mappings $\{f_{map}^i\}_{i=1,...,n}$
  - Approaches differ in $f_{def}$ and $\{f_{map}^i\}_{i=1,...,n}$
    - Highly restricted
Open Problems

- **Problems of missing input data**
  - $S_{\text{init}} \Rightarrow$ derivation of schema from sample XML documents $D$
  - $D \Rightarrow$ analyses of real XML data
  - $Q \Rightarrow$ dynamic adaptability

- **Efficient solution of subproblems**
  - Numerous simplifications (omitting of mixed contents, recursion, ...)
  - $f_{\text{def}}$ is always fixed $\Rightarrow$ combination with cost-driven idea

- **Deeper exploitation of user-given information**
  - Idea: Schema annotations = "hints" how to store particular XML patterns $\Rightarrow$ similar fragments should be stored similarly

- **Theoretical analysis of the problem**
  - No theoretic study of XML-to-XML transformations + NP-hardness

- **Dynamic adaptability**
  - Changes of queries or data $\Rightarrow$ crucial worsening of efficiency $\Rightarrow$ dynamic changes of the schema
Publications


Note: The Best Paper Award


Content

1. Introduction
2. Analysis of Related Work
3. Hybrid User-Driven Adaptive Method
4. Similarity Function
5. Statistical Analysis of Real-World XML Data
6. Query Evaluation
7. Conclusion
8. Summary
User-Driven Methods: Shortcomings and Improvements

- Default mapping strategy $f_{def}$ is always fixed
- Systems are able to store schema fragments in various ways
- Weak exploitation of user-given information
  - Annotations from $A$ are just directly applied
  - Idea: Annotations = "hints" how a user wants to store XML patterns

$\Rightarrow$ General idea: Emphasis on user-given information
- Searching for similar fragments in the not annotated schema parts
  - The user is not forced to annotate all schema fragments
  - The system can reveal new structural similarities
- Searching for optimal mapping strategy for the remaining fragments
  - Adaptive strategy
  - Another exploitation of similarity
Schema of the Mapping Process
Adaptive Strategy

- **Key operations:**
  - **Contraction** = replaces each annotated fragment with an auxiliary node
  - **Expansion** = all auxiliary nodes are expanded to original schema fragments

- **Algorithm:**
  1. The searching for similar fragments and operation contraction repeats until there are no identified candidates for annotating
  2. The resulting schema is expanded

- **Assumption:** Reliable similarity function

- **Open Issues:**
  - Can we find similar schema fragments?
  - Can we find any in contracted graphs?
  - How many contractions can be applied, if any?

⇒ Experiments
Results

The percentage of annotated nodes

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>dat</th>
<th>doc</th>
<th>ex</th>
<th>rep</th>
<th>res</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average number of iterations</td>
<td>2.7</td>
<td>3.9</td>
<td>2.9</td>
<td>4.1</td>
<td>4.3</td>
</tr>
<tr>
<td>Average % of not annotated nodes</td>
<td>2.1</td>
<td>53.4</td>
<td>13.5</td>
<td>25.6</td>
<td>31.1</td>
</tr>
<tr>
<td>% of fully contracted schemes</td>
<td>93.7</td>
<td>22.2</td>
<td>81.1</td>
<td>0.0</td>
<td>28.6</td>
</tr>
</tbody>
</table>
Publications


Note: Nomination to the Excellent Paper Award

Content

1. Introduction
2. Analysis of Related Work
3. Hybrid User-Driven Adaptive Method
4. Similarity Function
5. Statistical Analysis of Real-World XML Data
6. Query Evaluation
7. Conclusion
8. Summary
Similarity Function (1)

- No suitable existing approach ⇒ proposal of a new one
- Focus on:
  - Schema-level similarity
  - Structural similarity
    - Existing works: semantic similarity
  - Aspects influencing the XML-to-relational mapping
    - e.g. omitting of element context
  - Reasonable tuning of parameters
    - Existing works usually omit
- Idea: Precise description and comparison of structure of schema fragments ⇒ exploitation of statistical analysis of real-world XML data
  - Analyzed characteristics describe data structure in detail
  - Results can be exploited for realistic tuning
Similarity Function (2)

- **Matcher** = similarity of a particular aspect
  - e.g. number of elements/attributes, depth, fan-out, etc.
  - Similarity of parameters = value $\in [0,1]$
- **Composite similarity function** = aggregation of results of matchers
  - Weighted sum $\Rightarrow$ tuning of weights?
  - Existing works: average of results of matchers
- **Idea**: Tuning the weights so that the function can identify similar number of given patterns as the analysis
- **Tuning process** = **constraints optimization problem**
  - Can be solved using respective approaches
    - Genetic algorithms, simulated annealing, etc.
Tuning Process - Average vs. Tuned Weights

- **R** = manually determined matches, **P** = matches determined by algorithm
- **I** = true positives, **F** = false matches
- Precision = \( |I| / |P| \) = reliability of the function
- Recall = \( |I| / |R| \) = share of real matches that is found
- Overall = \((|I| - |F|) / |R|\) = post-match effort
Publications


Mlýnková, I.: Similarity of XML Schema Fragments Based on XML Data Statistics.

Note: Paper under review
Content

1. Introduction
2. Analysis of Related Work
3. Hybrid User-Driven Adaptive Method
4. Similarity Function
5. Statistical Analysis of Real-World XML Data
6. Query Evaluation
7. Conclusion
8. Summary
Analyzed Data

- Semi-automatically collected
- Removal of damaged, artificial, too simple, or useless XML data

Statistics

<table>
<thead>
<tr>
<th>Statistics</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of XML documents</td>
<td>16,534</td>
</tr>
<tr>
<td>Number of XML collections</td>
<td>133</td>
</tr>
<tr>
<td>Number of DTDs/XSDs</td>
<td>98</td>
</tr>
<tr>
<td>Total size of documents (MB)</td>
<td>20,756</td>
</tr>
<tr>
<td>Minimum size of a document (B)</td>
<td>61</td>
</tr>
<tr>
<td>Maximum size of a document (MB)</td>
<td>1,971</td>
</tr>
<tr>
<td>Average size of a document (MB)</td>
<td>1.3</td>
</tr>
<tr>
<td>Documents with DTD (%)</td>
<td>74.6</td>
</tr>
<tr>
<td>Documents with XSD (%)</td>
<td>38.2</td>
</tr>
<tr>
<td>Documents without DTD/XSD (%)</td>
<td>7.4</td>
</tr>
</tbody>
</table>

- Testing collections – Shakespeare's plays, XMark, Inex, …
- Standard XML schemes – XHTML, SVG, RDF, DocBook, …
- Database exports – FreeDB, IMDb, …
- Known document types – OpenOffice, …
- Randomly crawled data – novels in XML, RNAdb, …
Contributions

• More detailed classification of XML data
  • 6 categories = 2 classical + 4 new ⇒ finer division
    • Data-centric, document-centric
    • Report, research, exchange, semantic web
  ⇒ Tests performed within the categories
• Confirmation or refutation of results of existing papers
  • Focus on often omitted constructs
  • Findings: Semi-automatically collected data have schema more often, recursion and mixed contents are not uncommon, etc.
• New findings and conclusions
  • Brand-new constructs ⇒ more detailed characteristics
    • New types of element fan-out and recursion, DNA patterns, relational patterns, etc.
• Detailed characteristics of real-world data per category
  ⇒ Tuning of similarity function
Publications


Note: The Best Student Paper Award


Note: An invited talk
Content

1. Introduction
2. Analysis of Related Work
3. Hybrid User-Driven Adaptive Method
4. Similarity Function
5. Statistical Analysis of Real-World XML Data
6. Query Evaluation
7. Conclusion
8. Summary
Open Issues of Query Evaluation

- **Correction** of the candidate set of annotations proposed by the algorithm
  - Annotations can be meaningless $\Rightarrow$ automatic identification
    - Not all combinations can be applied or are required by the user
  - Multiple choices $\Rightarrow$ user interaction + default settings

- Annotated **fragments can intersect**
  - General problem of user-driven approaches
    - Existing works: the allowed mapping strategies are too simple
  - Interface between storage strategies
    - Processing of parts of a query using different storage strategies
  - How to cope with redundancy
    - A single fragment can be stored using multiple strategies $\Rightarrow$ which of them should be used?
Correction of Annotations

• Types of annotation intersections:
  • **Redundant** = both storage strategies are applied
    • e.g. XHTML fragments ⇒ CLOB + shredding into tables
  • **Overriding** = only one of the storage strategies is applied
    • Classical situation of default mapping + annotations
  • **Influencing** = storage strategies are combined
    • e.g. shredding into tables + additional numbering schema

• Sample set of annotations + experimental system
  • Demonstration of meaningless and multiple-choice combinations
    • e.g. simple numbering schema must be always combined with a kind of shredding
    • e.g. storing into CLOB can be redundant or overriding
### Attribute Table

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>INOUT</td>
<td>inline, outline</td>
<td>Specifies whether the annotated fragment should be inlined or outlined to/from parent table.</td>
</tr>
<tr>
<td>GENERIC</td>
<td>edge, attribute, universal</td>
<td>The annotated fragment is stored using the specified type of generic-tree mapping strategy, i.e. Edge, Attribute, or Universal mapping.</td>
</tr>
<tr>
<td>SCHEMA</td>
<td>basic, shared, hybrid</td>
<td>The annotated fragment is stored using the specified type of schema-driven mapping strategy, i.e. Basic, Shared, or Hybrid mapping.</td>
</tr>
<tr>
<td>TOCLOB</td>
<td>true</td>
<td>The annotated fragment is stored to a CLOB column.</td>
</tr>
<tr>
<td>INTERVAL</td>
<td>true</td>
<td>The annotated fragment is indexed using the Interval encoding.</td>
</tr>
</tbody>
</table>

### Overriding + Redundant

<table>
<thead>
<tr>
<th></th>
<th>INOUT</th>
<th>GENERIC</th>
<th>SCHEMA</th>
<th>TOCLOB</th>
<th>INTERVAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>INOUT</td>
<td>⊘</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>GENERIC</td>
<td>×</td>
<td>⊘</td>
<td>✓</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>SCHEMA</td>
<td>×</td>
<td>✓</td>
<td>⊘</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>TOCLOB</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
<td>⊘</td>
<td>×</td>
</tr>
<tr>
<td>INTERVAL</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
</tbody>
</table>

### Influencing

<table>
<thead>
<tr>
<th></th>
<th>INOUT</th>
<th>GENERIC</th>
<th>SCHEMA</th>
<th>TOCLOB</th>
<th>INTERVAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>INOUT</td>
<td>⊘</td>
<td>✓</td>
<td>✓</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>GENERIC</td>
<td>×</td>
<td>⊘</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>SCHEMA</td>
<td>×</td>
<td>×</td>
<td>⊘</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>TOCLOB</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>⊘</td>
<td>×</td>
</tr>
<tr>
<td>INTERVAL</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
<td>×</td>
<td>⊘</td>
</tr>
</tbody>
</table>
Interface and Redundancy

- Interface – depends on the supported set of mapping strategies
- General types of annotations:
  - **Early binding** = processed before the XML schema is mapped
    - Modify the structure of the relational schema – e.g. INOUT, TOCLOB
  - **Late Binding** = exploited as late as a query is evaluated
    - Enhances a storage strategy with additional information – e.g. INTERVAL
- **Redundancy** ⇒ multiple ways how to evaluate a query (a kind of query plan)
  - Evaluation graph
    - Edges = storage strategies
    - Vertices = interfaces among storage strategies
    - Length of an edge = cost of evaluating of part of a query with a possible strategy + cost of interface between the strategy and the previous one
  ⇒ shortest path search
Publications


Note: Paper under review
Content

1. Introduction
2. Analysis of Related Work
3. Hybrid User-Driven Adaptive Method
4. Similarity Function
5. Statistical Analysis of Real-World XML Data
6. Query Evaluation
7. Conclusion
8. Summary
Conclusion and Future Work

- Main contributions of the thesis
  - Detailed analysis of existing works and possible improvements
  - Proposal of a hybrid user-driven adaptive XML-to-relational mapping strategy
  - Proposal of a schema-level structural similarity function
  - Tuning process
  - Statistical analysis of real-world XML data
- Current research
  - Elaborate implementation of the proposed system
    - Currently: prototype implementation
    - Emphasis: "Side" aspects, improvement of query evaluator
  - Extending of annotations with expected queries
- Possible future work
  - Combination with true cost-driven approaches
  - Dynamic adaptation of the relational schema
Content

1. Introduction
2. Analysis of Related Work
3. Hybrid User-Driven Adaptive Method
4. Similarity Function
5. Statistical Analysis of Real-World XML Data
6. Query Evaluation
7. Conclusion
8. Summary
Summary

- 8 refereed papers:
  - 7 international conferences
    - IEEE Computer Society, Springer, McGraw-Hill, 2x International Association for
      Development of the Information Society, 2x local proceedings
    - 2 best (student) paper awards, 1 nomination to excellent award
  - 1 journal: International Journal of Computer Science and Applications
- 4 nonrefereed papers:
  - 2 invited talks (EurOpen '04, XML Prague '06)
  - 6 technical reports
  - 191 pages in total
- Textbook:
    - 38 pages
- Citations:
  - 5 international conferences (ACM, 2x IEEE Computer Society), 3 local
    journals and conferences, 5 theses (Masaryk University, University of
    West Bohemia, Czech Technical University, 2x Charles University)