An XML-to-Relational User-Driven Mapping Strategy Based on Similarity and Adaptivity

Irena Mlynkova

irena.mlynkova@mff.cuni.cz

Charles University

Faculty of Mathematics and Physics

Department of Software Engineering

Prague, Czech Republic

Introduction

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

⇒ **A natural alternative: To exploit tools and functions of (object-)relational database management systems ((O)RDBMS)**

(–) XML trees vs. flat relations ⇒ inefficiency

(+) Long theoretical and practical history, mature technology

⇒ **The techniques should be further enhanced**

Goals of This Presentation

Proposal of improvement of XML processing based on (O)RDBMS

- **Overview and classification of existing approaches**
- **Motivation for improvements**
- **Proposal of improvement of user-driven methods**
- **Related open issues**
- **Conclusion**

Content

1. **Overview of existing approaches**
2. Motivation for improvements
3. Proposed improvement
4. Related open issues
5. Conclusion

Managing XML Data

- File system
  - (–) Inability of querying without additional pre-processing of data
- Pure object-oriented approach
  - (–) No efficient and comprehensive tool
- Native methods
  - (+) No need to adapt structures to a new purpose
- (O)RDBMS ⇐
  - (+) Most practically used

Database-Based XML Processing Methods (1)

Key concern: Choice of the most efficient XML-to-relational mapping strategy

- **Fixed** – predefined set of mapping rules and heuristics
  - Generic vs. schema-driven
- **Adaptive** – adapt the target schema to intended usage
  - Cost-driven
- **User-involving** – storage decisions in hands of users
  - User-defined vs. user-driven

Database-Based XML Processing Methods (2)

- **Generic** vs. **schema-driven** – omitting / exploiting XML schema
  - Straightforward mapping
- **Cost-driven** – search a space of possible mappings and choose the one which conforms the target application most = the least "expensive"
  - Application: sample XML data, XML queries
- **User-defined** vs. **user-driven** – the amount of user involvement
  - User-driven = a type of adaptivity
    - Schema is adapted to the annotations

User-Driven Methods: Shortcomings and Improvements

- **Default mapping strategy is always fixed**
  - **Systems are able to store schema fragments in various ways ⇒ adaptive enhancing is natural**
- **Weak exploitation of user-given information**
  - **Annotations are just directly applied**
  - **Idea: Annotations = "hints" how a user wants to store particular XML patterns ⇒**
    - **We search for similar fragments**
    - **We use the knowledge in adaptive enhancing**

⇒ General idea: Emphasis on user-given information

Mlynkova, Pokorny: Adaptability of Methods for Processing XML Data using Relational Databases - the State of the Art and Open Problems. RCIS 2007. Ecole Marocaine des Sciences de l'Ingénieur, 2007.

Content

1. Overview of existing approaches
2. Motivation for improvements
3. Proposed improvement
4. Related open issues
5. Conclusion

Why User-Given Information? (Example 1)

- Situation: Documents with XHTML fragments
- Problem: Shredding into tables = inefficient fragment reconstructions
  - XHTML DTD contains complete graphs on up to 10 nodes
- What if the real complexity is much simpler?
  - Statistical analysis: Yes, it is <u>much</u> simpler!

$\Rightarrow$ Simpler storage strategy (CLOB)

Mlynkova, Toman, Pokorny: Statistical Analysis of Real XML Data Collections. COMAD 2006. Tata McGraw-Hill Publishing Company Limited, 2006.

Why User-Given Information? (Example 2)

- Situation: Updatability of data vs. fast query evaluation
- Problem: Amount of mutual relationships information
  - Fast querying $\Rightarrow$ additional indices, numbering schemes
  - Fast updates $\Rightarrow$ the simplest information of mutual relationships
  - Fast querying, fast updates $\Rightarrow$ compromise

Why User-Given Information? (Example 3)

- Situation: Data redundancy
- Question: Is it always necessary to strictly follow the rules of normal forms?
  - No, it is not.
  - Optimal XML-to-relational storage strategy = 4NF
    - No null values, no redundancy

⇒ In all the cases we need additional information given by a user
  - XML data, XML queries, annotations…

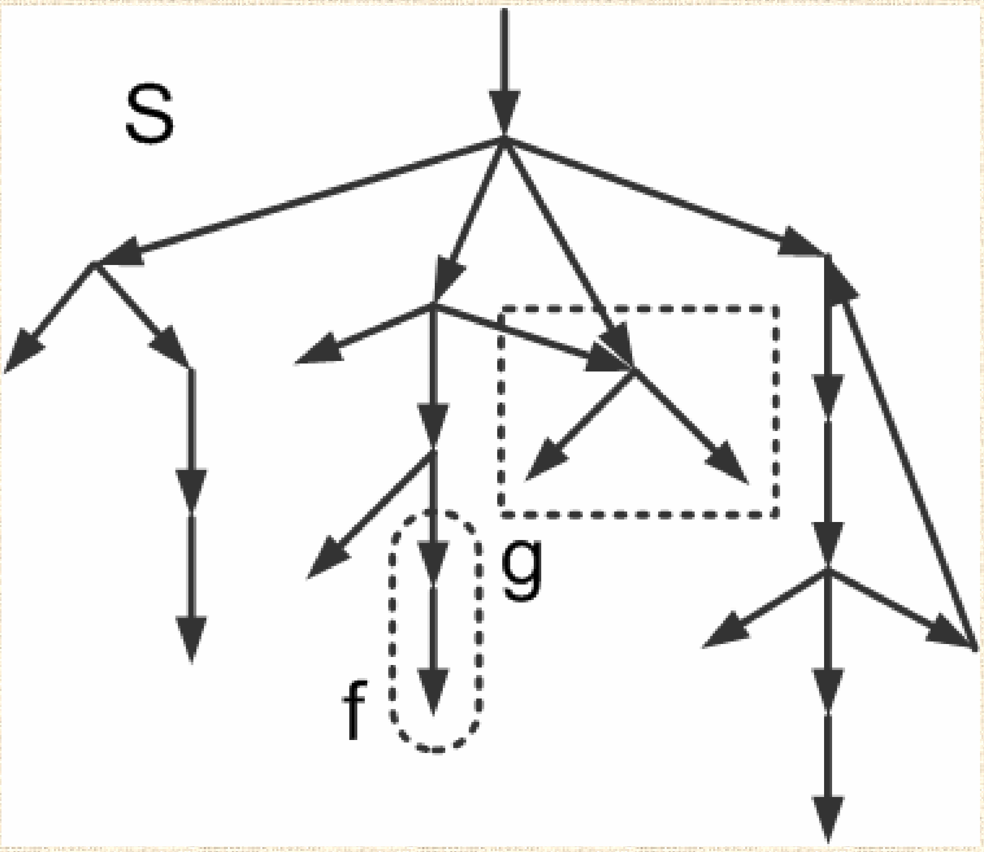
Content

1. Overview of existing approaches
2. Motivation for improvements
3. **Proposed improvement**
4. Related open issues
5. Conclusion

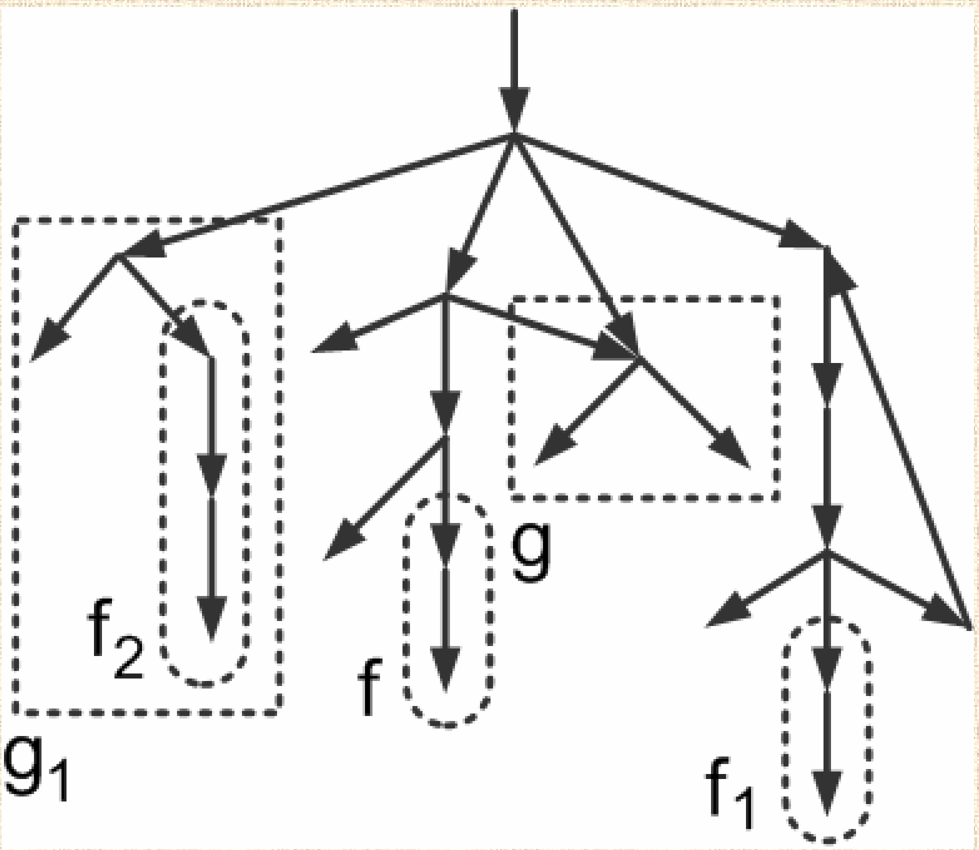
Basic Ideas

- 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 schema fragments
  - Adaptive strategy

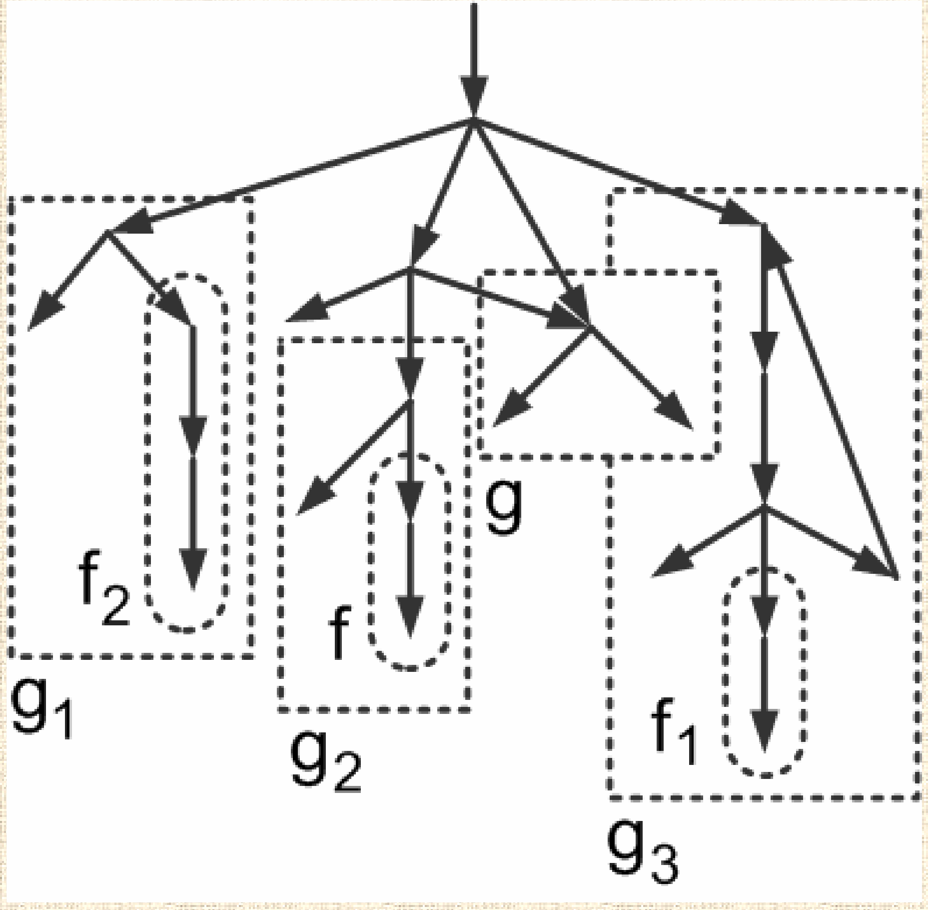
Step 1. Annotated Schema



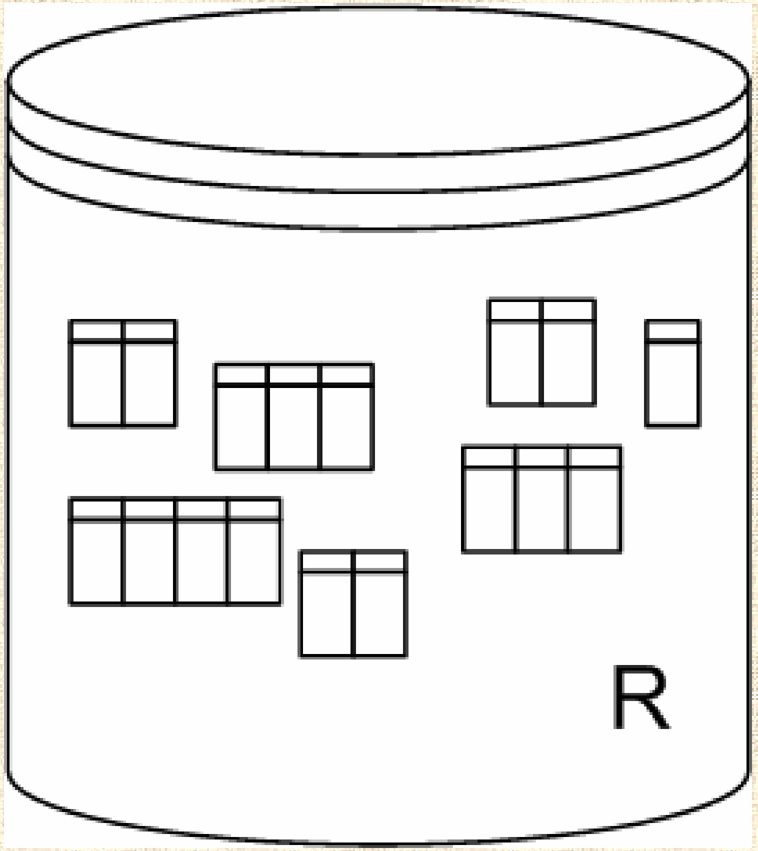
Step 2. Searching for Similar Fragments



Step 3. Adaptive Strategy



Step 4. Mapping to Relations



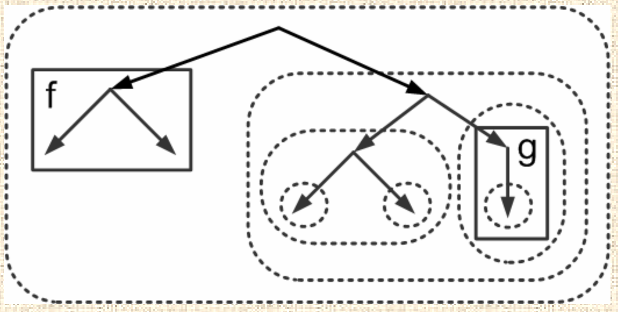
Open Issues

- What **types of annotations**, i.e. fixed mapping strategies, are supported? How are they combined?
- What **similarity measure** is used? Can we optimize the exhaustive search for similar fragments?
- Should we use a classical **adaptive strategy**?

Types of Annotations

- Particular mapping methods influence versatility of implementation
  - CLOB, shredding to tables, indices, numbering schemes…
- Key aspect: Intersection of annotated fragments
  - Redundant - both methods are applied on intersection
    - XHTML fragments ⇒ shredding + CLOB
  - Overriding - only one of the methods is applied
    - Classical user-driven strategies
  - Influencing - both methods are combined into one storage strategy
    - Shredding + indices/numbering schemes

Similarity Measure and Search Algorithm (1)



- Closely related
  - No knowledge of measure

    ⇒ Few ways how to avoid exhaustive search
    - Clustering (expensive preprocessing)
- Idea: Exploit knowledge of the similarity measure
- Modification of a classical approach:
  - A set of matchers = partial similarity measures
    - Similarity of a particular feature
      - Depth, number of nodes, complexity of content…
  - A composite similarity measure = combines the results
    - Weighted sum

Similarity Measure and Search Algorithm (2)

- **Features of matchers $\Rightarrow$ a bottom-up strategy**
  - **Knowledge for child nodes $\Rightarrow$ knowledge for parent node**
    - **e.g. depth, number of nodes**
- **Heuristics: Searching can terminate if reasonable amount of matchers exceed their only optimum within the current root path**
  - **Not possible for the composite measure**
  - **e.g. similarity of depths**
    - **With growing number of nodes the depth grows until it reaches the depth of the searched schema fragment**
- **Note: Formally described in the paper**

Adaptive Strategy (1)

- Classical approach: Target DB schema is adapted to sample XML data and queries
  - + annotations = too much information
- Idea:
  - Queries = How the data are typically manipulated
  - Data = How complex are XML documents

$\Rightarrow$ How to store the data
  - Annotations = How particular schema fragments should be stored

$\Rightarrow$ Annotations can be reused $\Rightarrow$ no need for additional information

Adaptive Strategy (2)

- **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**
- **Intersection of original and new annotations: Newly defined are overridden**

Content

1. Overview of existing approaches
2. Motivation for improvements
3. Proposed improvement
4. **Related open issues**
5. Conclusion

Features of Implementation

- Determine the usability and versatility of system
  - The set of annotations and their intersection
  - Partial matchers
  - Composite similarity measure
    - Key problem: Tuning of parameters
  - Similarity threshold(s)
  - Side effects:
    - User intervention when more possibilities occur
    - Support for forbidden intersection of annotations
    - Support for fixed fragments
    - …

Behavior on Real-World Data

- We know typical characteristics of real-world data - where is the problem?

- The behavior of more complex similarity measure cannot be predicted
  - Tuning process
- The behavior of the search algorithm on contracted graph cannot be predicted
  - No research on contracted graphs
    - Of course…

Content

1. Overview of existing approaches
2. Motivation for improvements
3. Proposed improvement
4. Related open issues
5. **Conclusion**

Currently Solved Issues

- Plenty of open issues = lots of work
- "Finished" research (under reviewing process)
  - Similarity measure focusing on structure of fragments
    - Tuning - based on statistical analysis of real-world data
  - Experimental evaluation of behavior of the adaptive strategy
    - Number of detected similar fragments, number of contractions, is the graph always contracted totally?
- Current research
  - Efficient querying over the resulting schema (query plans vs. intersecting annotations)
  - Combination with classical adaptive methods

Mlynkova: UserMap - an Enhancing of User-Driven XML-to-Relational Mapping Strategies. Technical report 2007/3. Charles University, Prague, Czech Republic, April 2007

Thank you