Similarity and XML Technologies

Irena Mlynkova, Jaroslav Pokorny

{irena.mlynkova,jaroslav.pokorny}@mff.cuni.cz

Charles University

Faculty of Mathematics and Physics

Department of Software Engineering

Prague, Czech Republic

Introduction (1)

- XML = a standard for data representation and manipulation
  - Well-defined, easy-to-learn, enough powerful

⇒ A boom of efficient implementations of W3C recommendations

- A possible optimization: Exploitation of similarity of XML data
  - Treating similar data in a similar way, storing "close" to each other, generalization of an approach to the whole set of similar data, etc.

Introduction (2)

- The amount of existing approaches is enormous
  - Which of the techniques to choose?
  - Is there a suitable approach? Or approach we can just modify?

⇒ Goal of the paper: Overview and classification of existing works

  - A good starting point for exploring existing approaches, their modification, or proposal of a new one

Road Map

1. Approaches to XML similarity
2. Similarity of XML documents
3. Similarity of XML documents and XML schemes
4. Similarity of XML schemes
5. Conclusion

Content

1. Approaches to XML Similarity
2. Similarity of XML documents
3. Similarity of XML documents and XML schemes
4. Similarity of XML schemes
5. Conclusion

Exploitation of Similarity in XML Technologies (1)

- Classical areas of pattern matching (= search for document fragments conforming the given pattern):
  - Query evaluation
    - Query modelled as a labelled tree, search for conforming fragments
  - Document validation
    - Schema is viewed as a template, whole document must conform
  - Document transformation
    - Search for correct fragments, all must cover the whole document
- This paper: a different scope

Exploitation of Similarity in XML Technologies (2)

- **Clustering**
  - Storing similar data in a similar way / close to each other $\Rightarrow$ fast retrieval, processing of relevant subset of data
- **Dissemination-based applications**
  - Timely distribute data from the underlying sources to a set of customers according to user-defined profiles
  - Approximate similarity evaluation
- **Data/schema-integration systems**
  - Provide a user with a uniform view of the data coming from different sources
  - Semantic similarity of the data

Exploitation of Similarity in XML Technologies (3)

- Data warehousing
  - Transform the data from source format to the warehouse format
- E-commerce
  - Message translation
- …

Classification of Approaches

- The **purpose** of similarity evaluation (see before)
- **Type** of the data
  - Data level (XML documents) vs. data type level (XML schemes) vs. between the two levels
- **Precision**
  - Similarity = value $\in [0,1]$
    - 0 = strong dissimilarity, 1 = strong similarity
  - Threshold $T_{sim} \in [0,1]$ = required precision
- **Depth** = the amount of exploited information
  - Structural level vs. tag name level vs. constraint level, or their combinations

Content

General Characteristics

- Huge amount of works
  - XML document = directed labelled ordered tree
- Approaches:
  - *sim*(D_1,D_2) = how difficult is to transform D_1 into D_2
    - Tree edit distance, tree alignment
  - Representation of D_1 and D_2 that enables efficient similarity evaluation
    - Path sets, document signal

1. Tree Edit Distance

Inspiration: Similarity of strings = number of adding and removing of a character

$\Rightarrow$ Similarity of trees = number of adding and removing of a node

- Problem in XML: Repeatable, optional, and alternative elements $\Rightarrow$ documents valid against a DTD can have different structure
  - Operation on single node cause high distance

$\Rightarrow$ More complex edit operations

  - Insert/delete node/subtree, re-label

- Problem: Multiple transformation sequences
  - Goal: Minimum edit distance

2. Tree Alignment

- **A variation of tree edit distance**
- **Alignment of trees T_1, T_2 = inserting λ-nodes into T_1, T_2 s. t. resulting trees T_1' and T_2' have the same structure (ignoring the node labels) and "overlaying" T_1' on T_2'**
  - **λ-node = an auxiliary node**
- **The same problem: Multiple alignments**
  - **Multiple positions for a λ-node**
  - **Goal: Minimum alignment distance**

3. Path Sets

- XML document can be represented using:
  - Set of distinct root paths
    - A path from the root node to a leaf node
  - Set of all distinct subpaths of root paths
  - Set of paths (root/subpaths) + frequencies
- Depends on application

$\Rightarrow$ Similarity evaluation = finding intersection of path sets and measuring its size

  - Problem: Omits order and values

4. Document Signal (1)

- **XML document = time series**
  - **Impulse = occurrence of a start/end tag**
- **Distinct tag names are ordered; start/end tag t_i is assigned its position +/- $\gamma(t_i)$**
- **Occurrence of t_i is assigned an impulse I_i**

$$I_i = \gamma(t_i) \cdot (N-1)^{D_{depth} - l_{t_i}} + \sum_{t_j \in anc(t_i)} \gamma(t_j) \cdot (N-1)^{D_{depth} - l_{t_j}}$$

- **N is the number of distinct tags, D_{depth} is the depth of document, l_{ti} is the level of tag occurrence t_i, $anc(t_i)$ is the set of ancestors of tag occurrence t_i**

4. Document Signal (2)

- **Impulse represents position in the document**
  - **The higher level, the higher impulse**
- **Similarity of documents D_1, D_2 = similarity of signals $S_1 = [I_1^1, I_2^1, ..., I_n^1]$ and $S_2 = [I_1^2, I_2^2, ..., I_m^2]$**
- **Algorithm:**
  - **Signals are periodically extended**
  - ***Discrete Fourier Transform* is applied**
  - **The result is linearly interpolated**

$\Rightarrow$ new signals $S_1' = [J_1^1, J_2^1, ..., J_M^1]$ and $S_2' = [J_1^2, J_2^2, ..., J_M^2]$

$$sim(D_1, D_2) = \sqrt{\sum_{k=1}^{M/2} (|J_k^1| - |J_k^2|)}$$

Content

General Characteristics

- Complex problem: tree vs. set of regular expressions

  $\Rightarrow$ low number of papers
- Approaches:
  - Measuring the number of elements which appear in document but not in schema and vice versa
    - Common, plus, and minus elements
  - Measuring the closest edit distance between document and "all" documents valid against schema

1. Common, Plus, and Minus Elements

- Types of elements:
  - common - appear in both document and DTD
  - plus - appear only in document
  - minus - appear only in DTD
- The lower number of plus and minus and higher number of common elements is, the higher similarity is
- Algorithm:
  - Matches elements at particular levels
  - Evaluates all possibilities $\Leftarrow$ optional, repeatable, and alternative elements
  - Chooses the one with the highest similarity

2. Edit Distance

- The edit distance of an element e and corresponding element declaration *f* $dist_e(f) = min\ \{dist(e, e') \mid e' \text{ matches } f\}$
  - $dist(e, e')$ = classical tree edit distance
  - Thompson's algorithm for automaton construction or Regular Hedge Grammar

Content

General Characteristics

- Huge amount of works
- General idea:
  - A set of matchers
  - Matcher = similarity of a particular feature of the given schema fragments
    - e.g. similarity of leaf nodes, similarity of root element names, similarity of context, etc.
  - Matchers are aggregated into the resulting similarity value
    - Weighted sum

1. Schema Integration

- Various subsystems provide a schema of their data
  - SGML, XML, relational, object-oriented, etc.
- Aim: to provide a uniform schema for querying
- The schemes are transformed into common graph representation
- Matchers focus on semantic similarity
  - Affixes, n-grams, edit string operations, phonetic similarity, path similarity, etc.
  - Aggregation of semantic similarity of child nodes, leaf nodes, siblings, etc.
- Sometimes combined with simple structural similarity (data types) or user interaction

2. Machine Learning

- Phases:
  - Training phase - user provides similarity mapping between sample schemes
  - Matching phase - the training sets are used to match new source schemes
- Problems:
  - No training data
  - User-specific similarity
  - If a particular type of schema is not in the training set, evaluation could be misleading

3. Schema Matching with Specific Conditions

- **Large schemes**
  - Schema is fragmented, similarities of fragments are evaluated and propagated into global similarity
- **Large number of schemes**
  - Exploitation of clustering
- **"Opaque" names / types**
  - Problem: Names and data types are not similar $\Rightarrow$ exploitation of other information
  - Probability distribution of a word (element name/data type) + entropy

4. Theoretic Studies and Comparisons (1)

- **Theoretic study**
  - **Schema matching = constraints optimization problem $\Rightarrow$ exploitation of COP solutions**
- **Taxonomy - criteria for**
  - **Matchers - elements vs. structure (sets of elements), language vs. constraints (semantics vs. keys), matching cardinality (1:1, 1:n, etc.), auxiliary information (thesauri), etc.**
  - **Aggregation - hybrid (combines matching approaches) vs. composite (combines results)**

4. Theoretic Studies and Comparisons (2)

- Efficiency evaluation - criteria influencing efficiency
  - Input - schema language, number of schemes, schema similarity, auxiliary information
  - Output - mapping between attributes or whole table, nodes or paths, etc., cardinality
  - Quality - the match tasks are first solved manually and then compared with the automatic ones
    - Precision = |B| / (|B| + |C|)
    - Recall = |B| / (|A| + |B|)
      - A = matches needed but not automatically identified
      - B = matches identified by manual and automatic processing
      - C = matches falsely proposed by the automatic processing
  - Effort - pre-match (training, configuration, etc.), post-match (correction)

Content

1. Approaches to XML Similarity
2. Similarity of XML documents
3. Similarity of XML documents and XML schemes
4. Similarity of XML schemes
5. **Conclusion**

Conclusions and Open Issues

- Similarity of documents is well analyzed
- Similarity of documents and schemes is complex, needs to be improved
  - Idea: Exploitation of automatic construction of a schema
- Similarity of XML schemes, though well analyzed, focuses mainly on semantics
  - Structural similarity is required
    - XML-to-relational mapping strategies
  - Ideas:
    - Matchers precisely describing the structure rather than semantics
    - Edit tree distance for schemes and operators

Thank you

Our Similarity Exploitation

- Exploitation of schema similarity in XML-to-relational mapping strategies
  - A set of matchers which precisely describe the structure of the schema
    - e.g. depth, width, number of elements/attributes, complexity of whole schema/particular levels, etc.
  - Tuning of weights of the weighted aggregation of results
    - Using results of statistical analysis of real-world data
    - Described and solved as an optimization problem

Mlynkova: Similarity of XML Schema Fragments Based on XML Data Statistics. Innovations ´07, Dubai, United Arab Emirates, November 2007. IEEE Computer Society.

Mlynkova: A Journey towards More Efficient Processing of XML Data in (O)RDBMS. CIT ´07, Aizu-Wakamatsu, Japan, October 2007. IEEE Computer Society.