Query Relaxation for XML

Dongwon Lee

U C L A
Outline

Introduction

XML Review

Main Work

- XML Relaxation Framework
- Relaxed Query Selectivity Issue
- Data Conversion Issue

Future Work

Summary
**HTML vs. XML**

HTML was designed to *display* data

XML was designed to *describe* data

- There is no fixed collection of markup tags. One must define his own tags, tailored for the kind of information
- XML separates syntax from semantics to provide a common framework for structuring information
- XML uses a schema language (e.g., DTD, XML-Schema) to formally describe the data.

HTML is one type of a bread; XML is a bread maker that can be used to bake various kinds
Details of XML

An XML document is a text with *markup tags*

Markup tags denote elements and attributes

- `<foo bar='val'> babo </foo>`
  - `<foo>`: an element start tag named *foo*
  - `bar`: an attribute with name *bar* and value *val*, enclosed by ' or ''
  - `babo`: the contents of the element
  - `</foo>`: a matching element end tag
  - `<foo/> = <foo></foo>`

An XML document must be *well-formed*:

- start and end tags must match
XML Example

<email>
  <from>Dongwon</from>  <to>Kate</to>
  <heading>Interview Schedule @ IST/PSU</heading>
  <body>Hi, Here is the detailed schedule...</body>
</email>

Root element is the topmost element: <email>

All elements must have closing tags:

- <p>foo<p>bar ⇒ <p>foo</p><p>bar</p>

XML tags are case sensitive & properly nested

- <Message>This is incorrect</message>
- <b><i>This is incorrect</b></i>
XML Example (cont)

Same information can be captured in both formats

HTML

```html
<html><head>
<title>Welcome to PSU</title>
</head><body>
<center>
<ul>
<li><a href='research.html'>Research</a></li>
<li><a href='academic.html'>Academics</a></li>
<li><a href='visitor.html'>Visitor's Guide</a></li>
...</ul>
</center></body></html>
```

XML

```xml
<xml>
 <item iID='school'>
  <sub-title addr='research.html'>Research</sub-title>
  <sub-title addr='academic.html'>Academics</sub-title>
  <sub-title addr='visitor.html'>Visitor's Guide</sub-title>
  ...
 </item>
...</xml>
```
Why is XML Important?

Simple, easy-to-use, intuitive tree model

Hot ($$$)

The standard for representation of Web information

Everyone is using it

The real force of XML is *generic languages and tools*!

By building on XML, you get a massive (standard) infrastructure for free
Welcome to XMLSOFTWARE

Welcome to XMLSOFTWARE. This site has been running for over two years, first at jtauber.com, but now at its own domain.

XMLSOFTWARE, along with the sister sites XMLINFO and SCHEMA.NET, aims to provide well organised information and resources on the Extensible Markup Language (XML), one of the most significant developments on the World Wide Web and in electronic publishing and electronic commerce.

How to Use This Site

The information and links on this site are organised into categories that you can always get to using the navigation along the left side of the page.

What's New / Update

Interspector
Code-based web development tool with support for XML.
version: 2.17
updated: 2002-03-21

X-Fetch Business Message Router
Business message routing system.
version: ?
updated: 2002-03-15

CorteXML
Tool for change-tolerant,
Outline

Introduction

XML Review

Main Work
  - *XML Relaxation Framework*
  - Relaxed Query Selectivity Issue
  - Data Conversion Issue

Future Work

Summary
What is Relaxation?
What is Relaxation?

I look for a flight from L.A. to U.P. for Saturday, 10am
I look for a flight from L.A. to U.P. for Saturday, 10am

My agent finds instead:

- A flight from L.A. to U.P. for Saturday, 3pm
What is Relaxation?

I look for a flight from L.A. to U.P. for Saturday, 10am

My agent finds instead:

- A flight from L.A. to U.P. for Saturday, 3pm
- A flight from L.A. to Pitt. for Saturday, 10am
What is Relaxation?

I look for a flight from L.A. to U.P. for Saturday, 10am

My agent finds instead:

- A flight from L.A. to U.P. for Saturday, \textit{3pm}
- A flight from L.A. to \textit{Pitt.} for Saturday, 10am
- A flight from L.A. to U.P. \textit{via Washington, Dulles} for Saturday, 3pm
What is Relaxation?

I look for a flight from L.A. to U.P. for Saturday, 10am

My agent finds instead:

- A flight from L.A. to U.P. for Saturday, 3pm
- A flight from L.A. to Pitt. for Saturday, 10am
- A flight from L.A. to U.P. via Washington, Dulles for Saturday, 3pm
- An Amtrak from L.A. to U.P. for Saturday, 3pm
- ...
What is Relaxation? (cont)

**Query Modification:** a query $S$ is “somehow” modified to $R$ such that $\langle Q \rangle \cup \langle R \rangle \neq \emptyset$, where $\langle Q \rangle$ is the answer of a query $Q$

- **Query Rewrite:** $\langle Q \rangle \equiv \langle R \rangle$
- **Query Restriction:** $\langle Q \rangle \supseteq \langle R \rangle$
- **Query Relaxation:** $\langle Q \rangle \subseteq \langle R \rangle$
- **Query Shift:** $\langle Q \rangle \not\supseteq \langle R \rangle \land \langle Q \rangle \not\subseteq \langle R \rangle$

Importance: query modification is a common activity in human discourse
Query Relaxation for XML

Needs for query relaxation for XML increase:

- XML has substantially bigger and more complex schema than relational model
- Users can’t ask complicated queries easily
- XML allows varied or missing structures and values
- Very common to integrate data over heterogeneous sources
Query Relaxation for XML

Needs for query relaxation for XML increase:

- XML has substantially bigger and more complex schema than relational model
- Users can’t ask complicated queries easily
- XML allows varied or missing *structures* and *values*
- Very common to integrate data over heterogeneous sources

⇒

*Relax* user’s query when not satisfactory
Query Relaxation for XML

Needs for query relaxation for XML increase:

- XML has substantially bigger and more complex schema than relational model
- Users can’t ask complicated queries easily
- XML allows varied or missing *structures* and *values*
- Very common to integrate data over heterogeneous sources

⇒

*Relax* user’s query when not satisfactory

*Relax* both *value* and *structure*
Issues for XML Query Relaxation

Clear understanding/specification of semantics
Issues for XML Query Relaxation

Clear understanding/specification of semantics $\implies$ QAC Framework
Issues for XML Query Relaxation

Clear understanding/specification of semantics $\Rightarrow$ QAC Framework

Effects to the existing techniques
Issues for XML Query Relaxation

Clear understanding/specification of semantics \(\implies\) \textit{QAC Framework}

Effects to the existing techniques \(\implies\) \textit{Query Selectivity Issue}
Issues for XML Query Relaxation

Clear understanding/specification of semantics $\Rightarrow$ **QAC Framework**

Effects to the existing techniques $\Rightarrow$ **Query Selectivity**

Issue

Re-use of relational model techniques
Issues for XML Query Relaxation

Clear understanding/specification of semantics $\Rightarrow$ QAC Framework

Effects to the existing techniques $\Rightarrow$ Query Selectivity Issue

Re-use of relational model techniques $\Rightarrow$ XML Data Conversion Issue
XML as a Tree

XML document can be represented as a Tree

Treat each element/attribute/string as a node with \((id, v)\):

- \(id\): unique integer (eg, pre-ordering)
- \(v\): label of the node
XML as a Tree (cont)

XML Query is also conceptually a tree

Capture the query in conjunctive relational notations

- node($var, val), pc_edge($var_v, $var_w), ad_edge($var_v, $var_w)

Answer to the query is the *boundings* of the variables $var$ of the query
Example XML Data

```
literature
  - book
    - author
    - year
    - pub
    - lastname: Hardy
      - year: 2000
      - pub: A&W
    - lastname: Ramanujan
      - year: 2000
    - pub: MIT
    - year: 2000
    - author: Erdos
  - journal
    - vol: 10
    - editor: Nash
      - year: 1999
    - revision
    - book
      - author
        - firstname: John
        - lastname: Nash
      - year: 1999
      - revision
```

Example XML Query

Q: “Find books written by Nash in 2000”

Edges: parent-child and ancestor-descendent relationships

Only one \textit{exact, total} match for $Q$
Example XML Query

Q: “Find books written by Nash in 2000”

Edges: parent-child and ancestor-descendent relationships

Only one *exact, total* match for Q
Example XML Query (cont)
Value Relaxation

$R_1$: “Find *literatures* written by Nash in 2000”

\[
\text{literature} \equiv \text{node}(\$1, \{\text{book, mag., journal}\}) \land \\
\text{node}(\$2, \text{year}) \land \text{node}(\$3, 2000) \land \\
\text{node}(\$4, \text{author}) \land \text{node}(\$5, \text{Nash}) \land \\
\text{pc}_\text{edge}(\$1, \$2) \land \text{pc}_\text{edge}(\$2, \$3) \land \\
\text{pc}_\text{edge}(\$1, \$4) \land \text{pc}_\text{edge}(\$4, \$5)
\]

Additional Knowledge

- book
- mag.
- journal

- monthly
- quarterly
- ...
Value Relaxation (cont)
$R_2$: “Find books \textit{somehow} written by Nash in 2000”

```xml
// Schema (DTD)
<!ELEMENT book (year|revision*|author+|pub|...)>  
<!ELEMENT year (#PCDATA)>  
<!ELEMENT revision (year|author+|pub|...)>  
<!ELEMENT author (firstname|lastname|#PCDATA)>  
<!ELEMENT firstname (#PCDATA)>  
```

```latex
\text{node}(\$1, \text{book}) \land \text{node}(\$2, \text{year}) \land \\
\text{node}(\$3, 2000) \land \\
\text{node}(\$4, \text{author}) \land \text{node}(\$5, \text{Nash}) \land \\
\text{ad\_edge}(\$1,\$2) \land \text{pc\_edge}(\$2,\$3) \land \\
\text{ad\_edge}(\$1,\$4) \land \text{ad\_edge}(\$4,\$5)
```
Edge Relaxation (cont)
$R_4$: “Find books \textit{written or edited} by Nash in 2000”

\[
\begin{align*}
\text{book} & \quad \text{year} \quad \text{author} \quad \text{Nash} \\
\text{node($1$,book)} & \land \text{node($2$,year)} \\
\text{node($3$,2000)} & \land \text{node($4$,author)} \\
\text{node($5$,Nash)} & \land \text{pc_edge($1$,2)} \\
\text{pc_edge($2$,3)} & \land \text{pc_edge($1$,4)} \\
\text{ad_edge($1$,5)}
\end{align*}
\]
Subtree Promotion (cont)

- Book
  - Author: John Nash
  - Year: 2000
- Journal
  - Volume: 10
  - Author: Nash
  - Year: 2000
- Revision
  - Year: 1999
  - Firstname: John
  - Lastname: Nash
- Editor: Nash
  - Publisher: A&W
- Publisher: MIT
Node Relaxation

$R_3$: “Find books related with Nash in 2000”

\[
\begin{align*}
\text{book} & \quad \equiv \\
\text{2000} & \quad \text{Nash} \\
\text{node}($1,\text{book}$) & \wedge \text{node}($2,\text{year}$) & \wedge \text{node}($3,2000$) & \wedge \text{node}($4,\text{author}$) & \wedge \text{node}($5,\text{Nash}$) & \wedge \\
\text{ad\_edge}($1,3$) & \wedge \text{ad\_edge}($1,5$) & \\
\end{align*}
\]
Node Relaxation (cont)
**QAC: Query Relaxation Framework**

In **QAC**, a query \( S \) is represented by triple \( S \equiv (Q, A, C) \), where:

- **\( Q \)** is a (labeled) conjunctive relational query, \( \bigwedge_i L_i : P_i \), where
  - \( L_i \) is an optional unique label
  - \( P_i \) is a conjunctive term

- **\( A \)** is a boolean function, called *Acceptance Test*, that takes as input the answer generated by executing the query \( Q \) over a database \( D \), and returns \{True, False\}

- **\( C \)** is a statement, called *Control Statement*, that guides the relaxation process as the user specifies
QAC Example

Q: “Find at least 10 books written by Nash in 2000”

QAC: $S \equiv (Q, A, \epsilon)$, where

\[ Q($1, $2, $3, $4, $5) \equiv node($1, book) \land node($2, year) \land \\
node($3, 2000) \land node($4, author) \land \\
node($5, Nash) \land \\
pc\_edge($1, $2) \land pc\_edge($2, $3) \land \\
pc\_edge($1, $4) \land pc\_edge($4, $5) \]

\[ A(\langle Q \rangle) \equiv count(\langle Q \rangle) \geq 10 \]
**QAC Example II**

- **Q**: “Find *at least 10* books written by Nash in 2000. *If needed, relax edges prior to nodes, but do not relax root. If any, exclude answers where $2=4$”

- **QAC**: $S \equiv (Q_2, A_2, C_2)$, where

\[
Q_2(\langle Q_1 \rangle) \equiv N_{book} : node(\langle Q_1 \rangle) \land N_{year} : node(\langle Q_1 \rangle) \land N_{2000} : node(\langle Q_1 \rangle) \land N_{author} : node(\langle Q_1 \rangle) \land N_{nash} : node(\langle Q_1 \rangle) \land E_{12} : pc\_edge(\langle Q_1 \rangle) \land E_{23} : pc\_edge(\langle Q_1 \rangle) \land E_{14} : pc\_edge(\langle Q_1 \rangle) \land E_{45} : pc\_edge(\langle Q_1 \rangle)
\]

\[
A_2(\langle Q_2 \rangle) \equiv (\langle Q_2 \rangle \not\in \langle Q_2 \rangle) \land \text{count}(\langle Q_2 \rangle) \geq 10
\]

\[
C_2 \equiv \neg N_1 \land [(E_1 \land E_4 \land N_1 \land N_2 \land N_3 \land N_4) \land (E_3 \land E_2)]
\]
**Summary**

*Query Relaxation* is a useful and powerful paradigm for XML query processing.
Summary

*Query Relaxation* is a useful and powerful paradigm for XML query processing

*QAC Framework* is capable of precisely describing the relaxation process for XML model
**Summary**

*Query Relaxation* is a useful and powerful paradigm for XML query processing

*QAC Framework* is capable of precisely describing the relaxation process for XML model

Remaining issues:
Summary

*Query Relaxation* is a useful and powerful paradigm for XML query processing

*QAC Framework* is capable of precisely describing the relaxation process for XML model

Remaining issues:

- How to support such query relaxation for XML model?
Summary

*Query Relaxation* is a useful and powerful paradigm for XML query processing

*QAC Framework* is capable of precisely describing the relaxation process for XML model

Remaining issues:

- How to support such query relaxation for XML model?
- Extend relational techniques
Summary

*Query Relaxation* is a useful and powerful paradigm for XML query processing.

*QAC Framework* is capable of precisely describing the relaxation process for XML model.

Remaining issues:

- How to support such query relaxation for XML model?
- Convert XML data to relational data
Outline

Introduction

XML Review

Main Work

- XML Relaxation Framework
- *Relaxed Query Selectivity Issue*
- Data Conversion Issue

Future Work

Summary
Relaxed Query Selectivity Problem

\[ N = \text{sel}(A) + \text{sel}(B) + \text{sel}(C') - \text{sel(overlap)} \]
Relaxed Query Selectivity Problem (cont)

Problem Def: Given an original XML query $Q$ and a set of $Q$’s relaxed XML queries $R_Q$, estimate the total number $N$ of the combined matches of $R_Q$: $N = sel(\sum_{r\in R_Q} r)$

Fundamental problem for

- Threshold problem: find all approximate answers beyond $\theta$
- Top-$k$ problem: find best $k$ approximate answers

Joint work with Divesh Srivastava @ AT&T Labs
Issues

\[ N = \text{sel}(A) + \text{sel}(B) + \text{sel}(C') - \text{sel}(A \cap B) - \text{sel}(B \cap C') - \text{sel}(A \cap C') + \text{sel}(A \cap B \cap C') \]
Issues

\[ N = \text{sel}(A) + \text{sel}(B) + \text{sel}(C) - \text{sel}(A \cap B) - \text{sel}(B \cap C') - \text{sel}(A \cap C') + \text{sel}(A \cap B \cap C') \]

Existing techniques cannot handle selectivity with \(\cap\)
Issues

\[ N = \text{sel}(A) + \text{sel}(B) + \text{sel}(C) - \text{sel}(A \cap B) - \text{sel}(B \cap C) - \text{sel}(A \cap C) + \text{sel}(A \cap B \cap C) \]

Existing techniques cannot handle selectivity with \( \cap \).

Do not want to drastically change existing techniques.
Our Approach

Step 1: using Suffix Tree and conventional selectivity estimation techniques, compute $sel(r)$ for each relaxed query $r \in R_Q$

Step 2: estimate overlap $sel(\text{overlap}) = sel(r_1 \cap ... \cap r_n)$

- Intersection symbol ($\cap$) cannot be handled by existing techniques
- Find a new formula $O \equiv r_1 \cap ... \cap r_n$ that does not contain intersections $\cap$
Solution 1: Using Automata Theory

Given \( A \cap B \), convert \( A \) and \( B \) into regular expressions \( R_a \) and \( R_b \)

Convert \( R_a \) and \( R_b \) into NFAs (non-deterministic finite automata) \( N_a \) and \( N_b \)

Merge two NFAs into the intersection NFA \( N_c \)

Convert \( N_c \) back to a regular expression \( R_c \)

Infer the final overlap formula \( O \) from \( R_c \)
Solution 1: Using Automata Theory

Given $A \cap B$, convert $A$ and $B$ into regular expressions $R_a$ and $R_b$

Convert $R_a$ and $R_b$ into NFAs (non-deterministic finite automata) $N_a$ and $N_b$

Merge two NFAs into the intersection NFA $N_c$

Convert $N_c$ back to a regular expression $R_c$

Infer the final overlap formula $O$ from $R_c$

Problem: Time Complexity is $O((nm)^34^{nm})$
Solution 2: Using Query Property

A
book
\[
\begin{array}{c}
\text{lastname} \\
\text{Nash}
\end{array}
\]
\[
\begin{array}{c}
\text{year}
\end{array}
\]

∩

B
book
\[
\begin{array}{c}
\text{author} \\
\text{Nash}
\end{array}
\]
\[
\begin{array}{c}
\text{year} \\
2000
\end{array}
\]

=
Solution 2: Using Query Property

A

\[
\text{book} \\
\text{lastname} \quad \text{year} \\
\text{Nash}
\]

\[\bigcap\]

B

\[
\text{book} \\
\text{author} \quad \text{year} \\
\text{Nash} \\
\text{2000}
\]

C

\[
\text{book} \\
\text{author} \quad \text{year} \\
\text{lastname} \\
\text{Nash} \\
\text{2000}
\]
Solution 2: Using Query Property

A

\(\text{book} \quad \text{lastname} \quad \text{year} \quad \text{Nash}\)

\(\cap\)

B

\(\text{book} \quad \text{author} \quad \text{year} \quad \text{Nash} \quad 2000\)

\(\Leftrightarrow\)

C

\(\text{book} \quad \text{author} \quad \text{year} \quad 2000\)

\(\text{lastname} \quad \text{Nash}\)

\(C\) is the overlap formula w/o intersection symbol

\(sel(A \cap B) \Leftrightarrow sel(C)\)
Experiments

Tested on two real datasets (DBLP, SPROT)

Tested various shapes of queries

- PATH: path queries
- DS: deep and skinny queries
- BS: bushy and shallow queries
- BAL: balanced queries

Tested different degrees of relaxations
Sample Experimental Results
Sample Experimental Results (cont)
Distribution of Absolute Error

<table>
<thead>
<tr>
<th>Error</th>
<th>% of queries</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt; 1.471725</td>
<td>75</td>
</tr>
<tr>
<td>&lt; 2.14375</td>
<td>20</td>
</tr>
<tr>
<td>&lt; 2.815775</td>
<td>5</td>
</tr>
<tr>
<td>&gt;= 2.815775</td>
<td>0</td>
</tr>
</tbody>
</table>

- **Our Method**
- **Upper Bound**
- **Lower Bound**
Distribution of Relative Error

- Estimate / Real count:
  - < 0.5
  - < 1
  - < 2
  - < 5
  - >= 5

- Our Method
- Upper Bound
- Lower Bound

% of queries vs. Estimate / Real count
Summary

We were able to solve the *Relaxed Query Selectivity Problem* while satisfying:

- Small sized auxiliary structure
- Minimal extension to relational techniques
- High accuracy of estimation
Summary

We were able to solve the *Relaxed Query Selectivity Problem* while satisfying:

- Small sized auxiliary structure
- Minimal extension to relational techniques
- High accuracy of estimation

This is just one area of relational database techniques extended
Summary

We were able to solve the *Relaxed Query Selectivity Problem* while satisfying:

- Small sized auxiliary structure
- Minimal extension to relational techniques
- High accuracy of estimation

This is just one area of relational database techniques extended

Many more research directions!
Outline

Introduction

XML Review

Main Work
  • XML Relaxation Framework
  • Relaxed Query Selectivity Issue
  • Data Conversion Issue

Future Work

Summary
Summary of Data Conversion Work

Data conversion between XML and Relational models

- Proposed two conversion algorithms
  - CPI: XML → Relational
  - NeT & CoT: Relational → XML

- Built XPRESS system
  - Consists of 50K Java/Perl/C++ codes
  - Successfully applied to 60MB DBLP data conversion
  - Downloaded and being used by 200+ world-wide users
Future Work

XML Query Relaxation framework affects many aspects of conventional relational database techniques

- Relaxed Query Selectivity
- Relaxed Query Evaluation & Optimization
- Relaxed Query Language
- ...
Conclusion

Proposed *XML Query Relaxation* as an intuitive framework towards intelligent query processing for XML model
Conclusion

Proposed *XML Query Relaxation* as an intuitive framework towards intelligent query processing for XML model

Details are available at:

http://www.cs.ucla.edu/~dongwon/paper/
Conclusion

Proposed *XML Query Relaxation* as an intuitive framework towards intelligent query processing for XML model

Details are available at:

http://www.cs.ucla.edu/~dongwon/paper/

Thank You!