Enhancing Traditional Databases to Support Broader Data Management Applications

Yi Chen
Computer Science & Engineering
Arizona State University
What Is a Database System?

- Of course, there are traditional relational database management systems (RDBMS)
- Was introduced in 1970 by Dr. E. F. Codd (of IBM)
- Commercial relational databases began to appear in the 1980s
- The focus of most work in the past 30 years
A Relational Database (RDBMS)

Table (Relation)

Row (Tuple, Record)

Climber

Column (Attribute)

Name | Skill | age
--- | --- | ---
James | Beginner | 21
Bob | Experienced | 33

Climbs

Name | Route       | Date       | Duration
--- | ----------- | ---------- | -------
Bob  | Last Tango | 10/10/05   | 5
Bob  | Last Tango | 1/10/06    | 4.5

A predefined data structure (schema) is required.
Querying RDBMS: SQL

Selection: $\sigma_{\text{Name} = \text{"James"}}$

Projection: $\Pi_{\text{Route} = \text{"Last Tango"}}$

Join: $\text{Climber} \bowtie_{\text{Climber.name} = \text{climbs.name}} \text{Climbs}$
The Advantages of RDBMS

- Good data organization
- High efficiency for large datasets via indexing and query optimization
- Concurrency control and reliability
But, 80% of the World’s Data is Not in RDBMS!

Examples:

- WWW, Emails
- Personal data, documents of various format
- Sensor data
- A lot of scientific data (experimental data, large images, documentation, etc)

Why not?

- There are several assumptions in relational databases that do not fit for handling this data.

My research addresses how to enhance RDBMS to manage them.
Challenges for RDBMS (I)

- **RDBMS Assumption**: data conforms to a predefined fixed schema, which is separated from the data itself.

- **Reality**: 
  - Data may be collected from different sources on the web, therefore has different schemas.
  - Schema can change over time for a single source.

- **Requirements**: We need to handle data of different schemas and have the schemas tightly associated with the data.
XML as a Data Representation Format

- XML has become a standard data format for various applications, because of:
  - Flexibility in schemas -- semi-structured data
  - Self-describing feature
  - Representing tree data model naturally
XML: the Standard for Web Data Representation

Web Service Requester

XML Data

Internet

XML Data

NCBI Web Service Publisher

PubMed

GenBank

BLAST

...
XML: Representing Phylogenetic Trees

From the Tree of the Life Website, University of Arizona
Challenges for RDBMS (II)

- **RDBMS Assumption:** Data is clean and consistent.

- **Reality:** real world data is dirty
  - Data collected from different sources may have missing and conflicting information
  - Data that is obtained from data mining is often not error-prone
  - Experimental data often contains random errors

- **Requirements:** we need to measure data quality and handle imprecise and/or incomplete data
Roadmap of This Talk

- Managing XML by leveraging mature RDBMS [Chen et al 04]
  - Introduction to XML
  - A generic and efficient XML-to-RDBMS mapping
    - Data mapping from trees to tables
    - Query translation from tree navigation queries to SQL queries that are efficient

- Handling imprecise and incomplete data in DBMS [Chen et al 06]
Sample XML Data

<books>
  <book>
    <title>The lord of the rings...</title>
    <section>
      <title>Locating middle-earth</title>
      ...  
    </section>
    ...  
  </book>
  ...  
</books>
Sample XML Queries

- XML query languages are based on hierarchical structure navigation (e.g. XPath)

- Sample queries:
  - What are all the section titles: //section/title

Descendant axis
Child axis
Sample XML Queries

- XML query languages are based on hierarchical structure navigation (e.g. XPath)

- Sample queries:
  - What are all the section titles: //section/title
  - What are the titles of sections that contain a figure: //section[//figure]/title
How to Query XML Data efficiently?

- RDBMS have achieved high performance in query evaluation.
- Can we leverage RDBMS by encoding XML to tables?
Analogy: Fourier Transforms

Complex

\[ g \ast h = \int_{-\infty}^{+\infty} g(u)h(u)du \]

Efficient

\[ G(f)H(f) \]
Mapping XML Data to RDBMS

Challenge:
How to build the bridge between hierarchies and tables?

XML data → XPath → XML fragments → Storage Mapping → SQL → Relational databases

Query Translation
### Data Mapping

**Diagram:**
- **Parent ID** [Florescu & Kossmann 99]
- **Graph:**
  - (1) books
  - (2) book
  - (3) title
    - “The lord of the rings …”
  - (4) section
  - (5) title
    - “Locating middle-earth”
  - figure
  - description
    - “King Theoden's golden hall”

**Table:**

<table>
<thead>
<tr>
<th>ID</th>
<th>Tag</th>
<th>Value</th>
<th>Structural Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>books</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>book</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>title</td>
<td>The...</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>section</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>title</td>
<td>Locating…</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>
Data Mapping

Design special labels to encode node relationships

<table>
<thead>
<tr>
<th>ID</th>
<th>Tag</th>
<th>Value</th>
<th>Structural Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>books</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>book</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>title</td>
<td>The...</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>section</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>title</td>
<td>Locating...</td>
<td></td>
</tr>
</tbody>
</table>

(1) books
(2) book
(3) title
(4) section
(5) title

“Locating middle-earth”

“King Theoden's golden hall”

“A hall fit for a king”

“The lord of the rings …”
How to choose XPath subqueries, such that:
- they can be easily translated to SQL subqueries
- the SQL subqueries can be efficiently evaluated

How to combine SQL subqueries to a complete one?
Query Translator

Q: //book[//figure]/section/title
Query Translator: (I) Decomposition to Suffix Paths

Q: //book[//figure]/section/title
Encoding Suffix Paths Using P-labeling

(342000,343000) \[ \sigma_{342000 \leq \text{Plabel} \leq 343000} T \]

Evaluating suffix paths SQL selections on P-labels
Query Translator: (II) Selection on P-labels

Q: //book[[//figure]/section]/title
D-labeling Scheme

- D-labeling is used to connect suffix paths.

- D-labels (start, end, depth) can be used to detect ancestor-descendant relationships between nodes in a tree.
A Bi-labeling Based Query Translation

(σ_{Plabel} for //book//figure(T'))
\[ T'.Start \leq T.Start \land T'.End \geq T.End \]
(σ_{Plabel} for //figure T)

Stitching suffix paths
SQL joins on D-labels
Query Translator: (III) Join on D-labels

Q: //book[//figure]/section/title
Comparison with Previous Approach

Previous Approach
[Li & Moon 01, Zhang et al 01, Tatarinov et al 02, Grust 02, DeHaan et al 03, etc]

Ours:
fewer disk accesses,
fewer joins
Experiment Setup

Compare our system (BLAS) with the previous approach using D-labeling scheme only

- **Data sets**

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Size (MB)</th>
<th>Nodes (K)</th>
<th>Tags</th>
<th>Depth</th>
<th>DTD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Protein</td>
<td>70</td>
<td>2277</td>
<td>66</td>
<td>7</td>
<td>Tree</td>
</tr>
<tr>
<td>Shakespeare</td>
<td>26</td>
<td>640</td>
<td>19</td>
<td>7</td>
<td>Acyclic graph</td>
</tr>
<tr>
<td>Auction</td>
<td>69</td>
<td>1238</td>
<td>77</td>
<td>12</td>
<td>Cyclic graph</td>
</tr>
</tbody>
</table>

- **Query sets**
  - Suffix path queries
  - Path queries
  - XPath queries
  - Benchmark queries

- **Query Engines:** DB2, TwigStack Join [Bruno et al 02]
Query Execution Time

Query Name:
A: Auction
P: Protein
S: Shakespeare

1: suffix path query
2: path query
3: XPath query

Time

D-labeling
BLAS

Queries

QA1
QA2
QA3
QP1
QP2
QP3
QS1
QS2
QS3

Yi Chen --- January 23, 2006
Scalability

Benchmark data, Benchmark query Q3

File Size (MB)

Time (seconds)

DLabeling
BLAS

Yi Chen --- January 23, 2006
Summary of XML Data Management

- We proposed a generic XML-to-RDB mapping, based on a bi-labeling scheme.

- It is more efficient compared with previous approach, since it generates SQL queries that require:
  - fewer disk accesses
  - fewer joins
  - fewer intermediate results

- Experiments show the effectiveness
Roadmap of This Talk

- Managing XML by leveraging mature RDBMS [Chen et al 04]
  - Introduction to XML
  - A generic and efficient XML-to-RDBMS mapping
    - Data mapping from trees to tables
    - Query translation from tree navigation queries to SQL queries that are efficient

- Handling imprecise and incomplete data in DBMS [Chen et al 06]
Probabilistic Databases: Managing Imprecise Data

- How to measure the imprecision of the data?
- The simplest model: associate each tuple a probability

<table>
<thead>
<tr>
<th>Climber</th>
<th>Name</th>
<th>Skill</th>
<th>age</th>
<th>Pr</th>
</tr>
</thead>
<tbody>
<tr>
<td>James</td>
<td>Beginner</td>
<td>21</td>
<td>p1</td>
<td></td>
</tr>
<tr>
<td>Bob</td>
<td>Experienced</td>
<td>33</td>
<td>p2</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Climbs</th>
<th>Name</th>
<th>Route</th>
<th>Date</th>
<th>Duration</th>
<th>Pr</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bob</td>
<td>Last Tango</td>
<td>10/10/05</td>
<td>5</td>
<td>q1</td>
<td></td>
</tr>
<tr>
<td>Bob</td>
<td>Last Tango</td>
<td>1/10/06</td>
<td>4.5</td>
<td>q2</td>
<td></td>
</tr>
</tbody>
</table>

Assume that all the tuples are independent events
How Does This Affect Query Evaluation? [Fuhr&Roellke 97]
Challenges: Correctness Depends on Execution Order [Suciu et al 05]!

Find distinct climb routes that have been climbed by an experienced climber.

\[ LT \ 1-(1-p_2q_1)(1-p_2q_2) \]

Bob LT \ p_2(1-(1-q_1)(1-q_2))

Wrong !

Correct

LT \ 1-(1-q_1)(1-q_2)

Bob LT p_2q_1
Bob LT p_2q_2

Name | Skill | Pr
--- | --- | ---
Bob | Exp | p_2
Bob | LT | q_1
Bob | LT | q_2

Name | Route | Pr
--- | --- | ---
Bob | LT | q_1
Bob | LT | q_2

ASU
How to Handle Incomplete Data?

Find cars that are convertible and have price less than 10k.

<table>
<thead>
<tr>
<th>Seller</th>
<th>Make</th>
<th>Model</th>
<th>Body Style</th>
<th>Year</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mark</td>
<td>BMW</td>
<td>325Cic</td>
<td>convertible</td>
<td>2006</td>
<td>36000</td>
</tr>
<tr>
<td>Alice</td>
<td>Ford</td>
<td>Taurus</td>
<td></td>
<td>2001</td>
<td>8000</td>
</tr>
</tbody>
</table>

This can not be convertible!

Our techniques infer possible values with probability by mining data statistics.
Querying Incomplete Databases

- Given missing value prediction
  Querying incomplete databases
  Querying probabilistic databases

- What if we are not able to store the predicted values? --- e.g. data integration applications

- On-the-fly query rewriting

How should we handling imprecise and incomplete XML Data?
Conclusions

- Traditional RDBMS does not satisfy the requirements in ever growing scientific and web applications

- We have discussed two enhancement to RDBMS
  - Efficient XML data management
  - Handling imprecise and incomplete data

- Other enhancement to RDBMS that I am working on
  - Data stream processing
  - Scientific workflow modeling and query processing
Thank you!

Questions?