Putting Semi-structured Data to Practice

Alon Levy

Seattle, Washingon

University of Washington

Semi-structured Data

- In many applications, data does not have a rigidly and predefined schema:
  - e.g., structured files, scientific data, XML.
- Managing such data requires rethinking the design of components of a DBMS:
  - data model, query language, optimizer, storage system.
- The emergence of XML data underscores the importance of semi-structured data.

Outline of the Talk

- Semi-formal definition and examples.
- Modeling semi-structured data
- Querying semi-structured data
- Challenges in practice:
  - Application: web-site management
  - The XML challenge
  - A DBMS challenge: query optimization
- Current research challenges

Main Characteristics

Schema is not what it used to be:

- not given in advance (often implicit in the data)
- descriptive, not prescriptive,
- partial,
- rapidly evolving,
- may be large (compared to the size of the data)

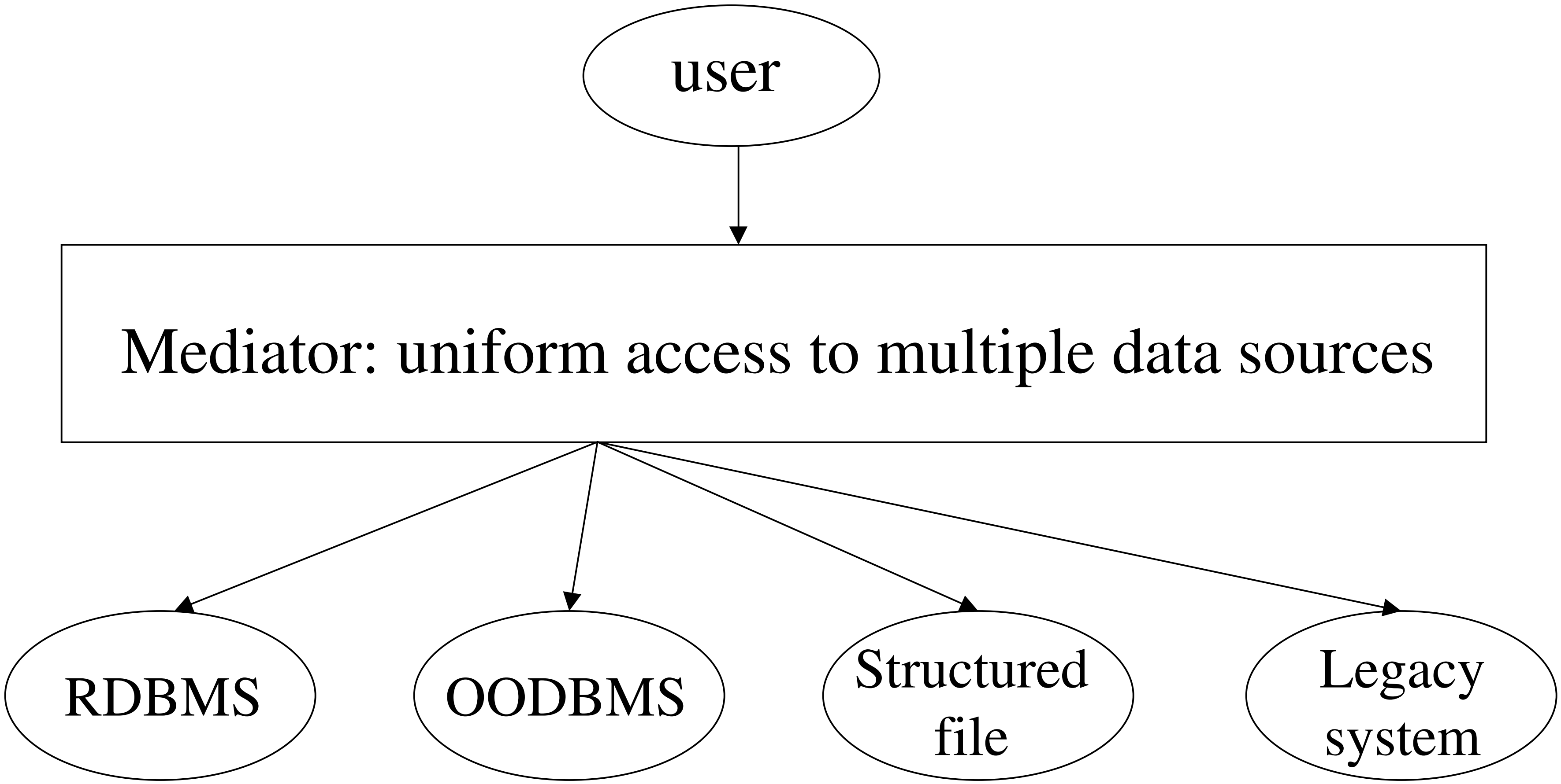
Types are not what they used to be:

- objects and attributes are not strongly typed
- objects in the same collection have different representations.

Example: XML

```xml
<bib>
  <book year="1995">
     <title>  Database Systems </title>
    <author> <lastname> Date </lastname> </author>
    <publisher> Addison-Wesley  </publisher>
  </book>
  <book year="1998">
    <title> Foundation for Object/Relational Databases </title>
    <author> <lastname> Date </lastname> </author>
    <author> <lastname> Darwen </lastname> </author>
    <ISBN> <number> 01-23-456 </number >  </ISBN>
  </book>
 </bib>
```

Example: Data Integration



Each source represents data differently:

different data models, different schemas

Physical versus Logical Structure

- In some cases, data can be modeled in relational or object-oriented models, but extracting the tuples is hard
  - extracting data from HTML:
    - [Ashish and Knoblock, 97], [Hammer et al., 97], [Kushmerick and Weld, 97].
- Semi-structured data: when the data cannot be modeled naturally or usefully using a standard data model.

Managing Semi-structured Data

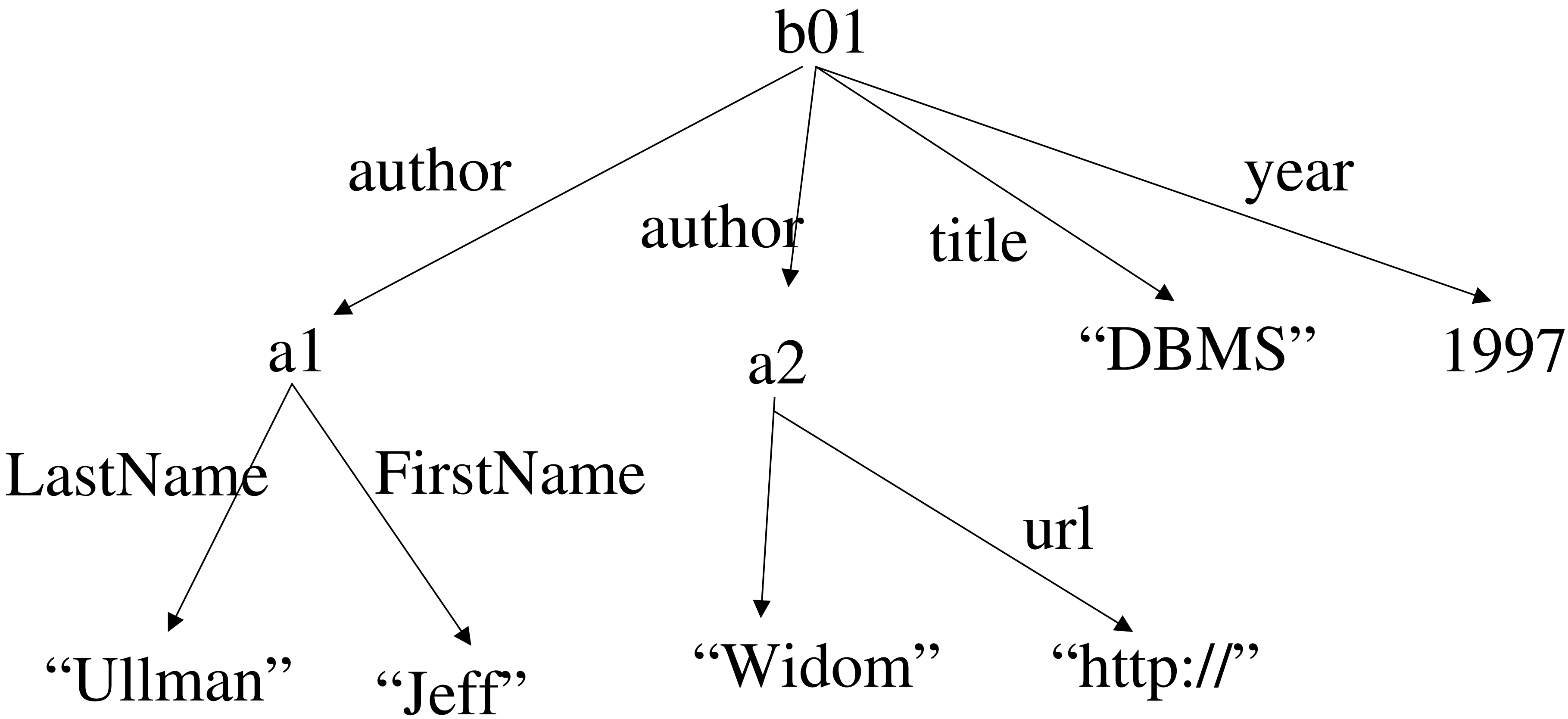
- How do we model it? (directed labeled graphs).
- How do we query it? (many proposals, all include regular path expressions).
- Optimize queries? (beginning to understand).
- Store the data? (looking for patterns)
- Integrity constraints, views, updates,…,

Outline of the Talk

- Semi-formal Definition and examples.
- Modeling semi-structured data
- Querying semi-structured data
- Challenges in practice:
  - Application: web-site management
  - The XML challenge
  - A DBMS challenge: query optimization
- Current research challenges

Modeling Semi-Structured Data

Labeled directed graphs: (from OEM [TSIMMIS]):



Nodes are objects; labels on the arcs are attribute names.

Querying Semi-structured Data

- Important features:
  - ability to navigate the data (regular path expressions),
  - querying the attribute names (arc variables),
  - create new structures,
  - type coercion.
- Languages: Lorel (Stanford), UnQL (U. Penn), StruQL (AT&T, INRIA, UW).

The StruQL Query Language

- A StruQL query is a function from a set of input graphs to an output graph.
- A StruQL expression contains two parts:
    - A query component, and
    - A restructuring component.

Formally:

INPUT *graph names*

WHERE *conjunction of regular path expression atoms*

CREATE *name the nodes in the output graph using Skolem functions*
LINK *specify the links in the resulting graph.*

OUTPUT *resulting-graph name.*

Example Query: StruQL

WHERE Articles(art), art -> l -> value,

l in { "Title", "Abstract", "Date", "Text",
"Image", "Topimage", "RelatedSite"},

art -> * -> art1, Article(art1)

CREATE ArticlePage(art), ArticlePage(art1)

LINK   ArticlePage(art) -> l -> att,
ArticlePage(art) -> “related article” -> ArticlePage(art1)

StruQL Details

- Regular path expressions are constructed by a grammar:

  R <- "a" | ε | R1.R2 | R1|R2 | R1* | L | _
- Atoms in the WHERE clause are of the form *X -> R -> Y* or *C(X)*

- The LINK clause includes atoms of the form:

  LINK f(X) --> "new link" --> g(X) or
  LINK f(X) --> L --> g(X)

- Queries can be nested, inheriting the WHERE clauses of their outer blocks.

Outline of the Talk

- Semi-formal Definition and examples.
- Modeling semi-structured data
- Querying semi-structured data
- Challenges in practice:
  - Application: web-site management
  - The XML challenge
  - A DBMS challenge: query optimization
- Current research challenges

Semi-Structured Data in Practice

- A significant application area:
  - Web-site management
- An unexpected test:
  - XML (Extended Markup Language)
- An important technical challenge:
  - Query optimization

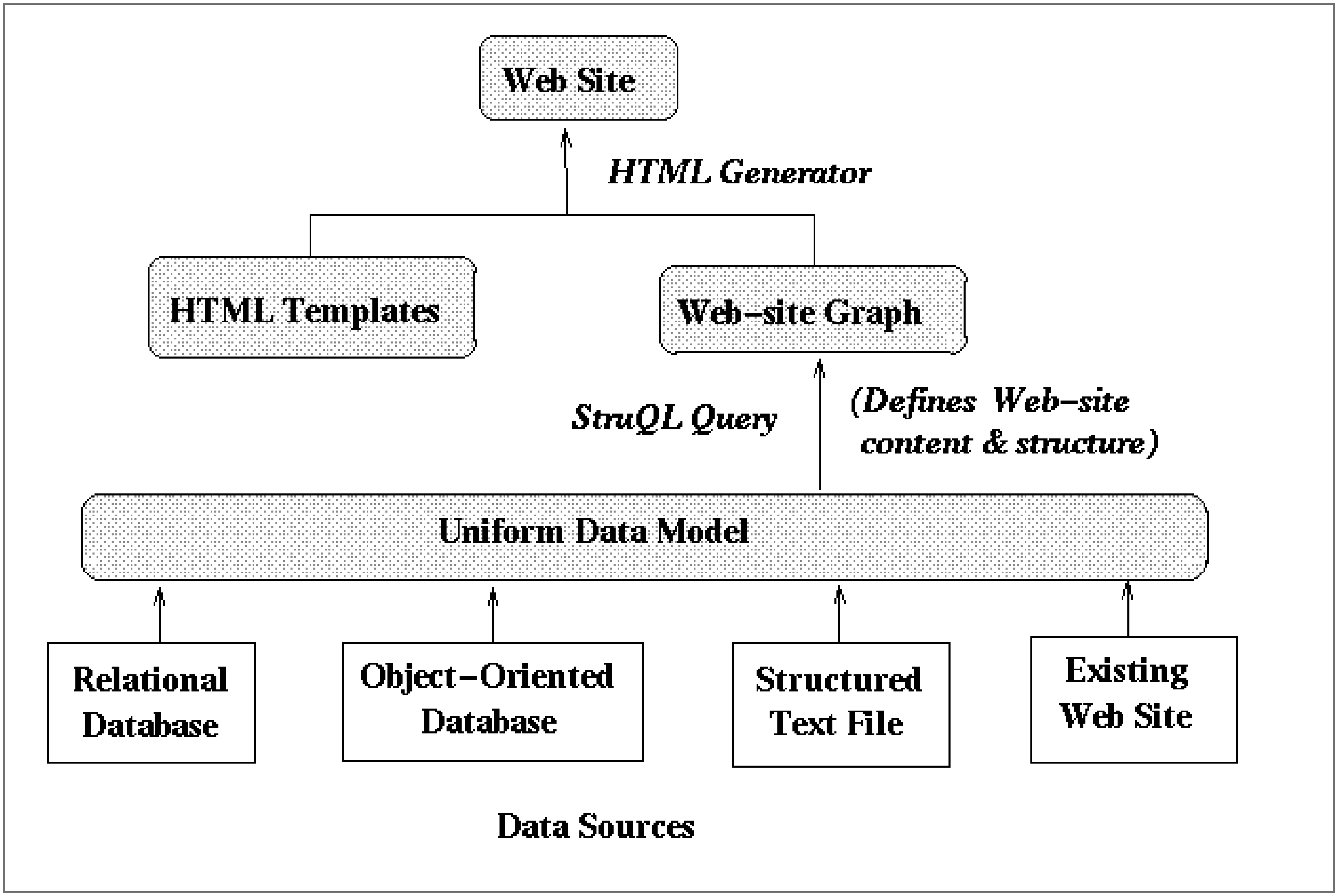
Web-Site Management

- **Problem:** designers are concerned with managing content, structure, and graphical presentation at the same time.
- **Consequently it is hard to:**
  - restructure web sites
  - enforce integrity constraints
  - easily create multiple sites from the same data
  - efficiently update a site.

Declarative Specification of Web-sites

- Key idea: specify the structure of the Web-site declaratively:
  - *A Web-site as a view over an integrated collection of data.*
- Several systems have been built following this paradigm:
  - Strudel (AT&T, INRIA, U. of Washington)
  - Araneus (U. of Roma), YAT (INRIA), Autoweb(Milan), **Tiramisu(UW)**

Strudel Architecture



Strudel

- Key ideas:
  - Introduce intermediate abstract representation of the web site:
    - Declaratively define the structure of the web site: *pages, links between them, and their content.*
  - Integrates content from multiple sources.
- Advantages:
  - Derives multiple sites from the same data.
  - Supports easy restructuring and modification.
  - Declarative representation is a platform for:
    - Specifying and enforcing integrity constraints,
    - Designing warehousing configuration to tradeoff site prematerialization and *click-time* computation.

Why Semi-structured Data?

- raw data is often semi-structured [e.g., DB&LP]
- convenient for data integration,
- web-sites are ultimately graphs,
- rapidly evolving schema of the web-site,
- schema of web-site does not enforce typing
- iterative nature of web-site construction.

Outline of the Talk

- Semi-formal Definition and examples.
- Modeling semi-structured data
- Querying semi-structured data
- Challenges in practice:
  - Application: web-site management
  - The XML challenge
  - A DBMS challenge: query optimization
- Current research challenges

The Test of XML

- XML (Extended Markup Language) is emerging as a standard for exchanging data on the Web.
- Enables separation of content (XML) and presentation (XSL).
- DTD's (Document Type Descriptors) provide partial schemas for XML documents.
- Applications will need to manage XML data.

Can the database community & semi-structured data be of any help?

Semi-structured Data vs. XML

- Attributes ---> tags
- objects ---> elements
- atomic values ---> CDATA (characters)

- Order? Assumed in XML.
- XML attributes (fixable)
- References in XML.

Real problem: XML comes with no data model!

References and Attributes

```
<bib>
  <book year="1995”, key=“o12”, references=“o24”>
     <title>  Database Systems </title>
    <author> <lastname> Date </lastname> </author>
    <publisher> Addison-Wesley  </publisher>
  </book>
  <book year="1998”, key=“o24”>
    <title> Foundation for Object/Relational Databases </title>
    <author> <lastname> Date </lastname> </author>
    <author> <lastname> Darwen </lastname> </author>
    <ISBN> <number> 01-23-456 </number >  </ISBN>
  </book>
 </bib>
```

Semantics of Queries with Order

```
select N
from Bib.book X,
     X.reference Y, Y.reference Z,
     Y.author.lastname N, Z.year U
where X.publisher = "Addison-Wesley"
ordered-by U
```

Semantics of the answer in unclear!

XML-QL

```
where <book>
        <publisher><name>Addison-Wesley</></>
        <title> $t</>
        <author> $a</>
      </> in "www.a.b.c/bib.xml"
construct <result>
            <author> $a</>
            <title> $t</>
         </>
```

Proposal submitted to the W3C (workshop to be held on December 3-4th).

Outline of the Talk

- Semi-formal Definition and examples.
- Modeling semi-structured data
- Querying semi-structured data
- Challenges in practice:
  - Application: web-site management
  - The XML challenge
  - A DBMS challenge: query optimization
- Current research challenges

Query Optimization: Challenges

- Statistics:
  - What do they even mean when the data is so irregular?
  - Data comes from external sources.
- Evaluation of regular path expressions:
  - need to optimize queries with limited forms of recursion.
- Mismatch between logical and physical schemas:
  - graphs are the logical model, but their storage varies considerably.

Logical vs. Physical Mismatch

- Graphs can be stored by:
  - materializing only forward pointers on edges,
  - maintaining some backward pointers
  - indexing on collections
- We can model the storage by binding patterns:
  - $\{title^{bf}\}$, $\{author^{bf}, author^{fb}\}$
- Other storage patterns can be modeled by GMAPs (Tsatalos et al., 96).

The Effect of Binding Patterns on the Search Space

- Need to search the space of annotated query plans:
  - every query execution plan is also annotated with the set of inputs it requires.
- If there are only few binding patterns available:
  - search space becomes smaller
- Multiple binding patterns per relation:
  - size of the space grows.

Florescu et al.: pruning methods for searching this space.

Conclusions

- Semi-structured data is everywhere.
- XML imposes a sense of urgency. An opportunity for the DB community to impact the WWW.
- We know how to model and query such data.
- Challenges: optimization, storage, adding partial structure.
- How can we help users structure information?