Data Integration Systems

Haas et al. 98
Garcia-Molina et al. 97
Levy et al. 96
Chandrasekaran et al. 2003

Zachary G. Ives
University of Pennsylvania

January 13, 2003
CIS 650 – Data Sharing and the Web

Administrivia

- Great job on the first round of reviews!
- Don't need to call me "Prof. Ives" – "Zack" is fine
- Office hours: Thursday 2:00-3:00 or send mail

- Students who didn't sign up to present will help out with presenting harder papers
- How many people need copies of the reader?
  - All papers are in PDF printable on the Web: http://www.cis.upenn.edu/~zives/cis650/
- Slides are available on the Web

Some Important Data Integration Design Points (from Monday)

- Garlic [Haas+97] – IBM Almaden (now in DB2)
  - Focus: intranet, SQL, few well-profiled source types
  - No mediated schema
- TSIMMIS [Garcia-Molina+97] – Stanford
  - Focus: semistructured data (OEM), OQL-based language (Lorel)
  - Mediated schema defined in terms of sources
- Information Manifold [Levy+96] – AT&T Research
  - Focus: local-as-view mappings, relational model
  - Sources defined in terms of mediated schema

Garlic: small-scale, controlled integration of heterogeneous data

- DB2 for heterogeneous source relations
  - Accept SQL query combining data across sources
  - Optimizer has built-in rules and cost estimators for each wrapped data source
    - Rules allow the optimizer to try all alternative ways of pushing operations to data source
    - Cost estimator predicts cost of executing at source, cost of shipping data
  - Limited query engine for combining data afterwards
- What's interesting about Garlic:
  - Commercially deployed – DB2 7.x+, DataJoiner
  - Design point is well-understood enough to do well

TSIMMIS and Information Manifold

- Focus: Web-based queryable sources
  - CGI forms, online databases, maybe a few RDBMSs
  - Each needs to be mapped into the system – not as easy as web search – but the benefits are significant vs. query engines
- A few parenthetical notes:
  - Focus of 1st generation systems is on languages and rewrite algorithms, not pure performance
  - Part of a slew of works on wrappers, source profiling, etc.
  - The creation of mappings can be partly automated – systems such as LSD, Cupid, Clio, … do this
  - Today most people look at integrating large enterprises (that's where the $$$ is!) – Nimble, BEA Liquid Data, Enosys, IBM DataJoiner/Garlic/Xperanto

TSIMMIS

- “The Stanford-IBM Manager of Multiple Information Sources” … or, a Yiddish stew
- An instance of a “global-as-view” mediation system
- One of the first systems to support semi-structured data

Semi-structured Data: OEM

- Observation: given a particular schema, its attributes may be unavailable from certain sources – inherent irregularity
- Proposal: Object Exchange Model, OEM

  OID: <label, type, value>

- … Does this look familiar in any way?
- … What problems does OEM solve, and not solve, in a heterogeneous system?

OEM Example

Show this XML fragment in OEM:

```
<book>
  <author>Bernstein</author>
  <author>Newcomer</author>
  <title>Principles of TP</title>
</book>

<book>
  <author>Chamberlin</author>
  <title>DB2 UDB</title>
</book>
```

Queries in TSIMMIS

- Specified in OQL-style language called Lorel
  - Different semantics: non-matching path NOT an error!
- Based on path expressions over OEM structures:

```
select library.book.title
where library.book.author = “Aho”
or library.book.subject = “compilers”
```

- Query converted to MSL template language

```
Q :- Q: <book {<title T> <author “Chamberlin”>}>
    AND EQ(T,”DB2 UDB”)
```

Query Answering in TSIMMIS – 1/2

```
Q :- Q: <book {<title T> <author “Chamberlin”>}>
    AND EQ(T,”DB2 UDB”)
```

- Wrappers have templates and binding patterns ($X) in MSL:

```
B :- B: <book {<author $X>}>
// $$ = “select * from book where author=“ $X //
```

- We find those that “match” (i.e., are at least as specific), as with B above
- Now we need to plug values in for binding patterns…

Query Answering in TSIMMIS – 2/2

- Now we provide the input to the view:

```
B :- B: book {select * from book
      where author = "Chamberlin"}
```

which would return:

```
    {o1: <book {o2: <author "Chamberlin">,
                o3: <year "1992">,
                o4: <title "DB2 UDB">}>,
     {o5: <book {o3: <author "Chamberlin">,
                o5: <title "DB2/CS">>,
     {o6: <book {o7: <author, "Chamberlin">,
                o8: <year, "1997">}>}
```

but we need to apply some other conditions to answer our query, so we do a **composition** with B's results:

```
    Q' :- Q': <book {<title T>}> AND EQ(T, "DB2 UDB")
```

○ (B: book {select * from book where author="Chamberlin"})

Strengths of TSIMMIS

- Early adopter of semistructured data
  - More powerful than relational global-as-view mediators, which can't support missing attributes
  - Doesn't fully solve heterogeneity problem, though!
- Simple algorithms for view unfolding
- Easily can be composed in a hierarchy of mediators
- … And one of the earlier data integration papers by a major DB group…

Limitations of TSIMMIS

- Some data sources may contain data with certain ranges or properties
  - "Books by Aho", "Students at UPenn", …
  - How do we express these? (Important for optimality!)
- Mediated schema is basically the union of the various MSL templates – as they change, so may it
- How do we come up with an optimal plan for executing a query?
- How do we execute the plan to get integrated data?

The Information Manifold

- Defines the mediated schema independently of the sources!
  - “Local-as-view” instead of “global-as-view”
  - Guarantees soundness and completeness of answers
  - Allows us to specify information about data sources
  - Focuses on relations (with OO extensions), datalog
- “Bucket algorithm” for query reformulation
  - Reduces typical amount of overhead in reformulation versus some other methods – we’ll hear more about these later in the semester

Observations of Levy et al.

- When you integrate something, you have some conceptual model of the integrated domain
  - Define that as a basic frame of reference
- May have overlapping/incomplete sources
  - Define each source as the subset of a query over the mediated schema
  - We can use selection or join predicates to specify that a source contains a range of values:

    ComputerBooks(…) $\subseteq$ Books(Title, …, Subj),
    Subj = “Computers”

The Local-as-View Model

- If we look at the Information Manifold model:
  - "Local" sources are views over the mediated schema
  - Sources have the data – mediated schema is virtual
  - Sources may not have **all** the data from the domain – "open-world assumption"
- The system must use the sources (views) to answer queries over the mediated schema
- This is "answering queries using views"

Answering Queries Using Views (for Conjunctive Queries)

- Assumption: queries are in datalog, are conjunctive queries, and we have set semantics
  - This means they have SELECT, PROJECT, JOIN with conjunction (AND) only

    q(a, t, p) :- author(a, i, _), book(i, t, p), t = "DB2 UDB"
- Some intuitions about this class of queries:
  - Adding a conjunct to a query removes answers from the result but never adds any
  - ➢ Any conjunctive query with at least the same constraints & conjuncts will give valid answers

The Bucket Algorithm

- Given a query Q with relations and predicates
  - Create a bucket for each subgoal in Q
  - Iterate over each view (source mapping)
    - If source includes bucket's subgoal:
      - Create mapping between q's vars and the view's var at the same position
      - If satisfiable with substitutions, add to bucket
  - Do cross-product of buckets, see if result is contained (exptime, but queries are probably relatively small)

Let’s Try a Bucket Example

- Query

  q(a, t, p) :- author(a, i, _), book(i, t, p), t = “DB2 UDB”

- Sources

  s1(a,t) :- author(a, i, _), book(I, t, p), t = “123”

  s2(a,t) :- author(a, i, _), book(I, t, p), t = “DB2 UDB”

  s3(a,t,p) :- author(a, i, _), book(I, t, p), t = “123”

  s4(a,i) :- author(a, i, _), a = “Smith”   s5(a,i) :- author(a, i, _)

  s5(i,p) :- book(I, t, p)

Source Capabilities in the Information Manifold

- Basically, these are ways of expressing binding patterns (plus a little more)
  - What parameters may be passed in – S_{in}
  - How many must be passed in – min <= # <= max
  - What variables are returned as output – S_{out}
  - What variables the source can select on – S_{sel}
  - Not supported: different schemas for diff. patterns

Given the binding patterns Book^{bff}(auth,title,pub) and Book^{fbf}(auth,title,pub), where we can also select on auth and title (using "< c"), what would the capability look like?

Strengths of Info Manifold

- More robust way of defining mediated schemas and sources
  - Mediated schema is clearly defined, less likely to change
  - Sources can be more accurately described
- Relatively efficient algorithms for query reformulation, creating executable plans

Weaknesses of Info Manifold

- Doesn’t support semistructured data
  - Answering queries using views is harder here!
- Still requires standardization on a single schema
  - Can be hard to get consensus
- Performance not really an emphasis

- Some other aspects were captured in related papers
  - Overlap between sources; coverage of data at sources
  - Semi-automated creation of mappings
  - Semi-automated construction of wrappers

Similarities & Differences between TSIMMIS and the Information Manifold

- Relatively concurrent – 1995-97 or so
- Both support input bindings and intend to integrate the Web
- Both support schema mediation, but using "opposite" formalisms
- Both use queries as the mappings between source and mediated schema

Later Systems Focused on Query Processing

Tukwila/Piazza [Ives+99,Halevy+02] – Washington

- Descendants of the Information Manifold
- Similar capabilities, but with adaptive processing of XML as it is read across streams

Niagara [DeWitt+99] – Wisconsin

- XML querying of web sources
- Giving answers a screenful at a time

TelegraphCQ [Chandrasekaran+03] – Berkeley

- Adaptive, select-project-join queries over infinite streams

TelegraphCQ Overview

- "Continuous queries" over data from sensors, stock market, etc.
  - Many such queries, registered by many users
  - Queries are over a "window" or interval:
    - What is average price of stock in the 5 minutes after it hits peak?
- Focus of TelegraphCQ is adaptivity:
  - Data characteristics change, so maybe query execution strategy needs to
  - Different queries are posted all the time – try to consolidate work as we go
  - No schema mediation, though!

Wrap-up for this Section

- At the heart of data integration is a translation problem:
  - Translation between data formats
  - Translation between query languages
  - Translation between schemas
- These problems aren't solved!
  - Even the best mapping language isn't expressive enough
  - Many of the problems are undecidable!
  - But they're usable enough for many apps, and heuristics (and best-effort) can be used
- Next week: Monday is MLK Jr Day; Wednesday we'll look at efforts to do "distributed data integration"