Integrationsformalismen

Semantische Datenintegration Seminar@Uni-Bremen

Oğuzhan Topsakal
June 20th, 2005
Outline

• Integration Formalism
• Types of Formalism Approaches
  – Rule-based Approach
  – Context Transformation Approach
• Global as a View (GAV)
• Local as a View (LAV)
• An Example for Rule Based: TSIMMIS
• An Example for Context Transformation: Farquar et al.
Integration Formalism & Its Types

Describes the mechanisms about how the heterogenous information from different information sources is combined in order to make a global view

- **Rule Based Approaches**
  - TSIMMIS at Stanford University

- **Context Based Approaches**
  - Formalizing Context (McCarthy et al.)
  - CARNOT System (Collet et al.)
  - COIN System (Goh et al.)
  - Integrating Sources Using Context Logic (Farquhar et al.)
Rule-based Approaches

Approaches for generating a mediated schema

- Global as a View (GAV)
- Local as a View (LAV)
Global as a View (GAV)

The schema elements of the global schema are defined over the schema elements of the local schemas (Query-centric)

• An Example Using GAV approach
  – Garcia-Molina et al. 1997 (TSIMMIS project)
Creating the Global Schema for GAV Approach

Course (Code, Title, Credits, UnivName)

Course relation of the Global Schema is defined in terms of local schema
relation Seminars of DS1, relation Courses of DS2, and relation Sections of DS3 forms
Course(..) => DS1, DS2, DS3
Local as a View (LAV)

The schema elements of the local schemas are defined over the schema elements of the global schema (View-centric)

Related database problems:

– Query optimization
– Maintaining physical data independence
– Data warehouse design
Creating the Views for LAV Approach *

Course(Code, Title, Credits, UnivName)

Global Schema

Course

Relation which corresponds to Course relation of the global schema is defined in terms of Course relation of the global schema

DS 1

Seminars

DS 2

Courses

DS 3

*Colors and direction of arrows change when compared to GAV example

*
Examples for GAV and LAV Approaches by Using THALIA Data

- **THALIA** (Test Harness for the Assessment of Legacy information Integration Approaches) is a publicly available testbed and benchmark for testing and evaluating integration technologies.

- THALIA provides data sources representing University course catalogs from computer science departments around the world.

- For a better understanding of GAV and LAV approaches, we provide examples by using THALIA data.
  - URL of THALIA: http://www.cise.ufl.edu/project/thalia.html
Global Schema & Local Schemas*

Suppose we have the following global (mediated) schema:

- **Course**((CourseCode, Title, Desc, Prereq, Credits, UnivName)
- **Instructor**((InstCode, Name, CourseCode, Email)
- **Location**((CourseCode, Room, Building)
- **Time**((CourseNo, Day, Hour)

Local Schemas of Universities:

- **DS1: Arizona_University**((Code, Time, Day, Place, Instructor) --- Only Graduate Level
- **DS2: Bremen_University**((Code, Instructor, Title, Room) --- Only In MZH Building
- **DS3: Carnegie_Mellon_University**((Code, Title, Day, Time, Units)
- **DS4: University_of_Florida**((Code, Title, Prereq, Description, Credits, Instructor, Day, Period, Building, Room) --- Only Courses with Prereq

*Global and Local Schemas are simplified for a clear example*
Query Answering by GAV Approach

**Course** (CourseCode, Title, Desc, Prereq, Credits, UnivName) => DS1, DS2, DS3 , DS4

**Instructor** (InstCode, Name, CourseCode, Email) => DS1, DS2 , DS4

**Location** (CourseCode, Room, Building) => DS1 , DS4

**Time** (CourseNo, Day, Hour) => DS1, DS3 , DS4

**Query1:** List the Codes of Courses given on Monday

\[ Q(CourseNo, \text{“Monday”}, \text{Hour}) \, :- \, \text{Time}(CourseCode, \text{“Monday”}, \text{Hour}) \Rightarrow \]
\[ \quad \text{DS1}(Code, Time, \text{”Monday”}, Place, Instructor) , \text{DS3}(Code, Title, \text{”Monday”}, Time, Units) \]

**DS1:** Arizona_University (Code, Time, Day, Place, Instructor) --- Only Graduate Level

**DS2:** Bremen_University (Code, Instructor, Title, Room) --- In MZH Building

**DS3:** Carnegie_Mellon_University (Code, Title, Day, Time, Units) – Only Weekends

**DS4:** University_of_Florida (Code, Title, Prereq, Description, Credits, Instructor, Day, Period, Building, Room) --- Only Courses with Prereq
Query Answering by LAV Approach

Arizona_University(Code, Time, Day, Place, Instructor) => Course(..), Instructor(..), Location(..), Time(..), \(^{(\text{UnivName} = \text{‘Arizona’}) ^{\text{\wedge}} (\text{CourseCode} > \text{‘500’)}}\)
Bremen_University(Code, Instructor, Title, Room) => Course(..), Instructor(..), Location(..) \(^{(\text{Building} = \text{‘MZB’)}}\)
Carnegie_Mellon_University(Code, Title, Day, Time, Units) => Course(..), Time(..)
University_of_Florida(Code, Title, Prereq, Description, Credits, Instructor, Day, Period, Building, Room) => Course(..), Instructor(..), Location(..), Time(..) \(^{\text{\wedge Prereq} <> \text{null}}\)

Query1: List the Codes of Courses given on Monday

Time(CourseNo, “Monday”, Hour) => Arizona_University(), Carnegie_Mellon_University(), University_of_Florida(..)

Course(CourseCode, Title, Desc, Prereq, Credits)
Instructor(InstCode, Name, CourseCode, Email)
Location(CourseCode, Room, Building)
Time(CourseNo, Day, Hour)
Comparison of GAV and LAV

• In Global as a View (GAV)
  – Reformulating the query in terms of the sources is easier (just needs unfolding of the query)
  – Adding a new source is harder. Requires redefinition of the global schema.

• In Local as a View (LAV)
  – Reformulating the query is harder.
  – Adding new source is easier (just need to express the new source as a view of the global schema)
  – It is easier to specify rich constraints on the contents of a source.
TSIMMIS Approach - Outline

- Goal and Overview of TSIMMIS
- Object Exchange Model (OEM)
- Mediator Specification Language (MSL)
- Wrapper Generation by rules
- Mediator Generation by rules
TSIMMIS Approach

- TSIMMIS stands for "The Stanford-IBM Manager of Multiple Information Sources"
- The TSIMMIS Project aims
  - To develop tools
  - To provide a framework

To assist humans to facilitate the rapid integration of heterogeneous information sources
  - Not to perform fully automated information integration
Requirements of a Mediator Architecture

- A common data model
- A common query language that allows
  - new mediators to join
  - new sources to provide input
- Tools to make the creation of new mediator systems easier
Components of TSIMMIS

- OEM data model
- MSL or LOREL query language
- Mediator and Wrapper Generator Tools
Object Exchange Model (OEM)

Components:
[OID | label | type | value]

**ObjectID**: Need not to be persistent

**Label**: Defines the object

**Type**: Either set or an atomic type

**Value**: Either an atomic value or a set of objects.
Mediator Specification Language

An example of a rule written in MSL:

\(<booktitle X> :- \<library \{<book \{<title X> <author "Aho">}\}> } > @s1\)

- Triangular brackets associate labels with their values.
- Curly brackets groups members of a set. This set is the value of an object that has a type set.
- The object pattern in the body is matched against the object structure of the source s1.
- The variable X binds to the value of the title subobjects of book objects that have an author subobject with the value 'Aho'.
Wrapper Generation Example

The wrapper generator takes a set of templates of the form:

MSL template
  // action //</action>

Example:
<books X> :-
  <library {X: <book {<title X> <author $AU}>}> }>@s1
  // sprintf(lookup-query,'find author %s',$AU) //</statement>

- The wrapper examines a query and compares it to the patterns in its specification file.
- If the query matches a pattern with some string in place of the parameter $AU$, then the associated action would be executed, with that string in place of the parameter.
Context Logic for Integration

• Context Logic
  – Extension of First Order Logic
    \( c\in\text{ist}(c, p) \)

• Idea
  – Define each information source as a context
  – Integrate the sources by lifting to a wider context
Research on Information Integration with Context Logic

- **Formalizing Context (McCarthy et al.)**
  - Defines context logic, lifting axioms
  - Gives an example for integrating databases
- **CARNOT system (Collet et al.)**
  - Defines articulation axioms which translate statements which are true in a source to statements which are meaningful in the Cyc knowledgebase
- **COIN system (Goh et al.)**
  - Forms a formal, logical specification of Context Interchange System with three components: Domain model, Elevation Axioms, Context Axioms
- **Integrating Sources Using Context Logic (Farquhar et al.)**
  - Translate relational DB tables into First Order Logic
  - Use lifting axioms of Context Logic to make implicit assumptions explicit
Integrating Information Sources Using Context Logic (Farquar et al.)

• Their goal is to enable
  – Meaningful integration across multiple sources
  – Users to access to complete power of an individual source
  – Taking advantage of their familiarity with a source

• Their approach
  – Reduces the up-front cost of integration
  – Expresses and resolves semantic conflicts
  – Provides incremental integration
Types of Context According to Farquhar et al. Approach

• Information Source Context
  – Direct translation of DB schema into assertions in first order logic
  – Done automatically but no semantic conflict is resolved

• Semantic Context
  – Lifting axioms are added manually to make the implicit assumptions explicit

• Integration Context
  – Contains axioms that lift sentences from several semantic contexts
Example: Product Database - Representing in First Order Logic

**Product table:**

<table>
<thead>
<tr>
<th>name</th>
<th>size</th>
<th>cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Television_1</td>
<td>14</td>
<td>256</td>
</tr>
<tr>
<td>Simm_1</td>
<td>256</td>
<td>14</td>
</tr>
</tbody>
</table>

**ProductType table:**

<table>
<thead>
<tr>
<th>name</th>
<th>type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Television_1</td>
<td>television</td>
</tr>
<tr>
<td>Simm_1</td>
<td>memory chip</td>
</tr>
</tbody>
</table>

**Information Source Context**

\[(\forall x,y,z \text{ product}(x, y, z) \Rightarrow \text{ string}(x) \land \text{ integer}(y) \land \text{ integer}(z))\]

\[
\text{relation(product)} \land \text{arity(product, 3)}
\land \text{primary-key(product, 1)}
\]

\[(\forall x,y,z \text{ product_type}(x, y) \Rightarrow \text{ string}(x) \land \text{ string}(y))\]

\[
\text{relation(product_type)} \land \text{arity(product, 2)}
\land \text{primary-key(product_type, 1)}
\]
Example: Product Database – Problems with Representation

**Product table:**

- **name** char  key
- **size** int
- **cost** int

<table>
<thead>
<tr>
<th>name</th>
<th>size</th>
<th>cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Television_1</td>
<td>14</td>
<td>256</td>
</tr>
<tr>
<td>Simm_1</td>
<td>256</td>
<td>14</td>
</tr>
</tbody>
</table>

**ProductType table:**

- **name** char  key
- **type** char

<table>
<thead>
<tr>
<th>name</th>
<th>type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Television_1</td>
<td>television</td>
</tr>
<tr>
<td>Simm_1</td>
<td>memory chip</td>
</tr>
</tbody>
</table>

Problems with representing a DB schema in logic:

- Attributes may be used polymorphically (Ex: size attribute can hold size in different units)
- Values need not have a unique denotation (Ex: The number 256 appears in both size and cost columns)

Solution is to use

- Existential quantification & Renaming
- Context Logic (Adding Lifting Axioms)
**Example: Product Database – Adding Lifting Axioms**

*Information Source Context*

\[
(\forall x,y,z \text{ product}(x, y, z) \Rightarrow \text{ string}(x) \\
& \text{ & integer}(y) \text{ & integer}(z))
\]

\[
\text{relation}((\text{product} \text{ & arity}(\text{product}, 3) \\
& \text{ & primary-key}(\text{product}, 1))
\]

\[
(\forall x,y,z \text{ product_type}(x, y) \Rightarrow \text{ string}(x) \\
& \text{ & string}(y))
\]

\[
\text{relation}((\text{product_type} \text{ & arity}(\text{product}, 2) \\
& \text{ & primary-key}(\text{product_type}, 1))
\]

* + *Lifting Axioms = Semantic Context*

\[
\text{ist}(\text{SC1}, \text{ product_type}(x, y)) \\
\Leftrightarrow \text{ist}(\text{IS1}, \text{ product_type}(x, y))
\]

\[
\text{ist}(\text{SC1}, \\
\exists y',z' \\
\text{ (product}(x, y', z') \\
& \text{ & magnitude}(y', \text{ natural-size-units}(x))=y \\
& \text{ & magnitude}(z', \text{ us-dollar})=z)) \\
\Leftrightarrow \text{ist}(\text{IS1}, \text{ product}(x, y, z)))
\]

\[
\text{ist}(\text{SC1}, \\
\text{ natural-size-units}(x) = \text{ bit}*1024 \\
\Leftrightarrow \text{ product-type}(x, \text{ memory-chip}))
\]

\[
\text{ist}(\text{SC1}, \text{ natural-size-units}(x) = \text{ inch} \\
\Leftrightarrow \text{ product-type}(x, \text{ television}))
\]
Integration Context

- Defined after constructing the information source context and semantic context
- Contains axioms that lift sentences from several semantic contexts
- Several Approaches are possible
  - Global Schema Approach
  - Federated Database Approach
  - Peer to peer Approach
Benefits of Using Context Logic

- Integrate new information sources incrementally
- Share assumptions without making them explicit
- Exploit ontologies
- Provide a richer model of integration
References

Fragen?

Vielen Dank für Ihre Aufmerksamkeit!