Ontology-Based Data Access: From Theory to Practice

Diego Calvanese

KRDB Research Centre for Knowledge and Data
Free University of Bozen-Bolzano, Italy

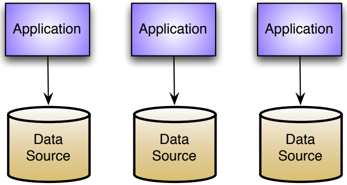
Freie Universität Bozen
Libera Università di Bolzano
Free University of Bozen · Bolzano

Currently on sabbatical leave at Technical University Vienna, Austria

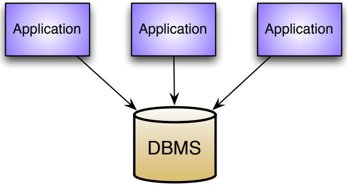
28e journées Bases de Données Avancées (BDA 2012)
24–26 October 2012, Clermont-Ferrand, France

Data management in information systems

Pre-DBMS architecture:

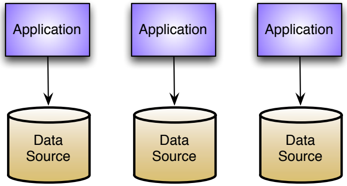


Ideal architecture based on a DBMS:



Data management today

In many cases, we are back at the pre-DBMS situation:



Data is:

- heterogeneous
- distributed
- redundant or even duplicated
- incoherent

Example 1: Statoil Exploration

Experts in geology and geophysics develop stratigraphic models of unexplored areas on the basis of data acquired from previous operations at nearby geographical locations.

Facts:

- 1,000 TB of relational data
- using diverse schemata
- spread over 2,000 tables, over multiple individual data bases

Data Access for Exploration:

- 900 experts in Statoil Exploration.
- up to 4 days for new data access queries, requiring assistance from IT-experts.
- 30–70% of time spent on data gathering.

Example 2: Siemens Energy Services

Runs service centers for power plants, each responsible for remote monitoring and diagnostics of many thousands of gas/steam turbines and associated components. When informed about potential problems, diagnosis engineers access a variety of raw and processed data.

Facts:

- several TB of time-stamped sensor data
- several GB of event data ("alarm triggered at time T")
- data grows at 30GB per day (sensor data rate 1Hz–1kHz)

Service Requests:

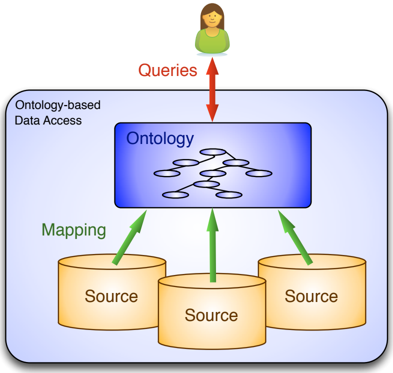
- over 50 service centers worldwide
- 1,000 service requests per center per year
- 80% of time per request used on data gathering

Proposed solution: Ontology-based Data Access

Manage data adopting principles and techniques studied in **Knowledge Representation** in Artificial Intelligence.

- Based on formalisms grounded in logic, with well understood semantics and computational properties.
- Provide a conceptual, high level representation of the domain of interest in terms of an **ontology**.
- Do **not migrate the data** but leave it in the sources.
- **Map** the ontology to the data sources.
- Specify all information requests to the data in terms of the ontology.
- Use the inference services of the OBDA system to translate the requests into queries to the data sources.

Ontology-based data access: Architecture



Based on three main components:

- Ontology: provides a unified, conceptual view of the managed information.
- Data source(s): are external and independent (possibly multiple and heterogeneous).
- Mappings: semantically link data at the sources with the ontology.

Ontology-based data access: Formalization

An **ontology-based access system** is a triple $\mathcal{O} = \langle \mathcal{T}, \mathcal{S}, \mathcal{M} \rangle$, where:

- $\mathcal{T}$ is the intensional level of an ontology.
  We consider ontologies formalized in description logics (DLs), hence the intensional level is a DL TBox.
- $\mathcal{S}$ is a (federated) relational database representing the sources;
- $\mathcal{M}$ is a set of mapping assertions, each one of the form

$$\Phi(\vec{x}) \rightsquigarrow \Psi(\vec{x})$$

  where
  - $\Phi(\vec{x})$ is a FOL query over $\mathcal{S}$, returning tuples of values for $\vec{x}$
  - $\Psi(\vec{x})$ is a FOL query over $\mathcal{T}$ whose free variables are from $\vec{x}$.

Ontology-based data access: Semantics

Let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ be an interpretation of the TBox $\mathcal{T}$.

Def.: **Semantics of an OBDA system**

$\mathcal{I}$ is a **model** of $\mathcal{O} = \langle \mathcal{T}, \mathcal{S}, \mathcal{M} \rangle$ if:

- $\mathcal{I}$ is a FOL model of $\mathcal{T}$, and
- $\mathcal{I}$ satisfies $\mathcal{M}$ w.r.t. $\mathcal{S}$, i.e., satisfies every assertion in $\mathcal{M}$ w.r.t. $\mathcal{S}$.

Def.: **Semantics of mappings**

We say that $\mathcal{I}$ **satisfies** $\Phi(\vec{x}) \leadsto \Psi(\vec{x})$ w.r.t. a database $\mathcal{S}$, if the FOL sentence

$$\forall \vec{x}.\, \Phi(\vec{x}) \rightarrow \Psi(\vec{x})$$

is true in $\mathcal{I} \cup \mathcal{S}$.

Note: the semantics of mappings is captured through material implication, i.e., **data sources** are considered **sound**, but **not necessarily complete**.

unibz.it

Challenges in OBDA

- How to instantiate the abstract framework?
- How to execute queries over the ontology by accessing data in the sources?
- How to deploy such systems using state-of-the-art technology?
- How to optimize the performance of the system?
- How to assess the quality of the constructed system?
- How to provide (automated) support for constructing the ontology?
- How to provide (automated) support for constructing the mappings?
- How to provide (automated) support for formulating queries?
- How to provide (automated) support for evolving the system (ontology, mapping, new data sources)?

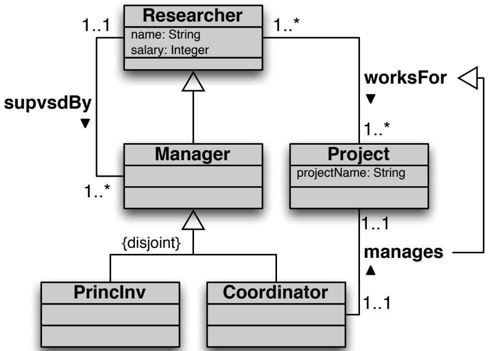
Instantiating the framework

1. Which is the “right” **ontology language**?
2. Which is the “right” **query language**?
3. Which is the “right” **mapping language**?

The choices that we make have to take into account the tradeoff between expressive power and efficiency of inference/query answering.

We are in a setting where we want to access large amounts of data, so **efficiency w.r.t. the data** plays an important role.

Ontologies vs. conceptual models



$$
\begin{array}{rcl}
\text{Manager} & \sqsubseteq & \text{Researcher} \\
\text{PrincInv} & \sqsubseteq & \text{Manager} \\
\text{Coordinator} & \sqsubseteq & \text{Manager} \\
\text{PrincInv} & \sqsubseteq & \neg\text{Coordinator} \\
\\
\text{Researcher} & \sqsubseteq & \exists \mathit{salary} \\
\exists \mathit{salary}^- & \sqsubseteq & \texttt{xsd:int} \\
 & (\textbf{funct}\ \mathit{salary}) & \\
\\
\exists\text{manages} & \sqsubseteq & \text{Coordinator} \\
\exists\text{manages}^- & \sqsubseteq & \text{Project} \\
\text{Coordinator} & \sqsubseteq & \exists\text{manages} \\
\text{Project} & \sqsubseteq & \exists\text{manages}^- \\
\text{manages} & \sqsubseteq & \text{worksFor} \\
 & (\textbf{funct}\ \text{manages}) & \\
 & (\textbf{funct}\ \text{manages}^-) & \\
 & \dots &
\end{array}
$$

We leverage on an extensive amount of work on the relationship between conceptual modeling formalisms and variants of DLs [Lenzerini and Nobili, 1990; Bergamaschi and Sartori, 1992; Borgida, 1995; C. *et al.*, 1999; Borgida and Brachman, 2003; Berardi *et al.*, 2005; Queralt *et al.*, 2012].

Outline

Use of mappings

In an OBDA system $\mathcal{O} = \langle \mathcal{T}, \mathcal{M}, \mathcal{S} \rangle$, the **mapping** $\mathcal{M}$ is a crucial component:

- $\mathcal{M}$ encodes how the data in the external source(s) $\mathcal{S}$ should be used to populate the elements of $\mathcal{T}$.
- We should talk about **OBDA** only when we are in the presence of a system that includes external sources and mappings.

Note:

- The data sources $\mathcal{S}$ and the mapping $\mathcal{M}$ define a virtual data layer $\mathcal{V} = \mathcal{M}(\mathcal{S})$ (i.e., a virtual ABox, in DL terminology), and queries are answered w.r.t. $\mathcal{T}$ and $\mathcal{V}$.
- We do not really materialize the data of $\mathcal{V}$ (that's why it is called virtual).
- Instead, the intensional information in $\mathcal{T}$ and $\mathcal{M}$ is used to translate queries over $\mathcal{T}$ into queries formulated over $\mathcal{S}$.

The impedance mismatch problem

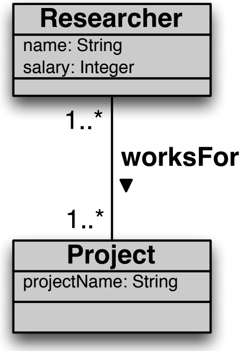
We need to address the **impedance mismatch** problem

- In relational databases, information is represented as tuples of values.
- In ontologies, information is represented using both objects and values . . .
  - . . . with objects playing the main role, . . .
  - . . . and values palying a subsidiary role as fillers of object attributes.

Proposed solution:

- We use **constructors to create objects** of the ontology from tuples of values in the DB.
- The constructors are modeled through Skolem functions in the query in the rhs of the mapping:
$$\Phi(\vec{x}) \ \rightsquigarrow \ \Psi(\vec{\mathbf{f}}, \vec{x})$$
- Techniques from partial evaluation of logic programs are adapted for unfolding queries over $\mathcal{T}$, by using $\mathcal{M}$, into queries over $\mathcal{S}$.

Impedance mismatch – Example



Actual data is stored in a DB:
– A researcher is identified by her SSN.
– A project is identified by its name.

$D_1[\mathit{SSN}\colon \text{String}, \mathit{PrName}\colon \text{String}]$
Researchers and projects they work for

$D_2[\mathit{Code}\colon \text{String}, \mathit{Salary}\colon \text{Int}]$
Researchers' code with salary

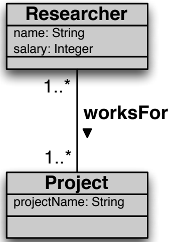
$D_3[\mathit{Code}\colon \text{String}, \mathit{SSN}\colon \text{String}]$
Researchers' Code with SSN

...

Intuitively:

- A researcher should be created from her SSN: **pers**(SSN)
- A project should be created from its name: **proj**(PrName)

Mapping assertions – Example



D_1[*SSN*: String, *PrName*: String]
Researchers and Projects they work for

D_2[*Code*: String, *Salary*: Int]
Researchers' code with salary

D_3[*Code*: String, *SSN*: String]
Researchers' code with SSN

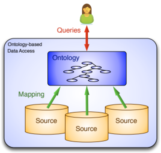
...

m_1:
```
SELECT SSN, PrName
FROM D1
```
$\rightsquigarrow$ Researcher(**pers**(*SSN*)),
Project(**proj**(*PrName*)),
projectName(**proj**(*PrName*), *PrName*),
worksFor(**pers**(*SSN*), **proj**(*PrName*))

m_2:
```
SELECT SSN, Salary
FROM D2, D3
WHERE D2.Code = D3.Code
```
$\rightsquigarrow$ Researcher(**pers**(*SSN*)),
salary(**pers**(*SSN*), *Salary*)

Outline

Incomplete information

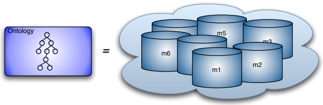


We are in a setting of **incomplete information**!!!

Incompleteness introduced:

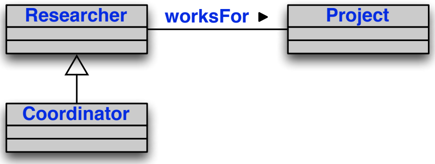
- by data source(s), in general assumed to be incomplete;
- by domain constraints encoded in the ontology.

Plus: Ontologies are logical theories, and hence perfectly suited to deal with incomplete information!



Minus: Query answering amounts to **logical inference**, and hence is significantly more challenging.

Incomplete information – Example



Assume for simplicity that each table in the underlying database is mapped directly to (at most) one ontology concept/relationship.

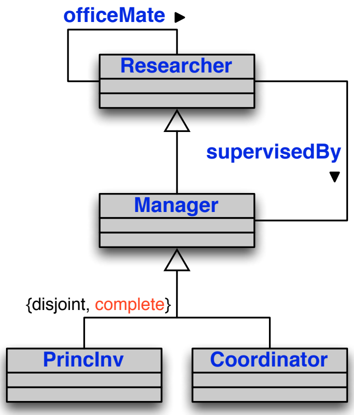
But the database tables may be **incompletely specified**, or even missing for some concepts/relationships.

DB: Coordinator $\supseteq$ { `serge`, `marie` }
Project $\supseteq$ { `webdam`, `diadem` }
worksFor $\supseteq$ { (`serge`,`webdam`), (`georg`,`diadem`) }

Query: $q(x) \leftarrow \mathsf{Researcher}(x)$

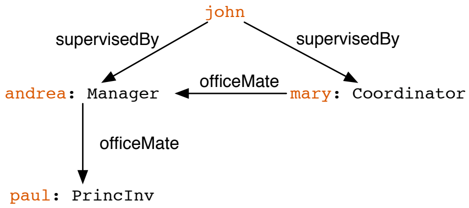
Answer: { `serge`, `marie`, `georg` }

QA over ontologies – Andrea's Example *



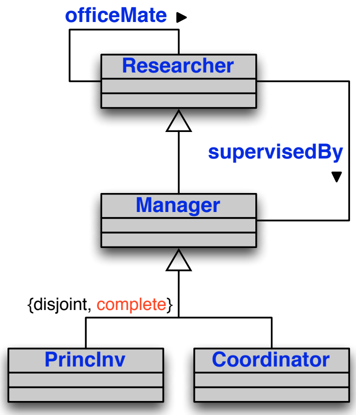
$\text{Manager} \equiv \text{PrincInv} \sqcup \text{Coordinator}$

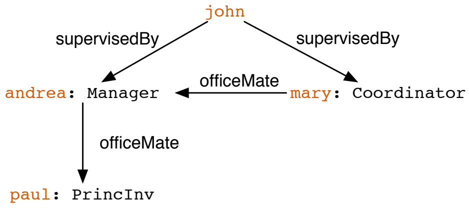
Researcher	⊇	{ andrea, paul, mary, john }
Manager	⊇	{ andrea, paul, mary }
PrincInv	⊇	{ paul }
Coordinator	⊇	{ mary }
supervisedBy	⊇	{ (john,andrea), (john,mary) }
officeMate	⊇	{ (mary,andrea), (andrea,paul) }



(*) By Andrea Schaerf [PhD Thesis 1994]

QA over ontologies – Andrea's Example (cont'd)





$$q(x) \leftarrow \exists y, z.\ \text{supervisedBy}(x, y),\ \text{Coordinator}(y),\ \text{officeMate}(y, z),\ \text{PrincInv}(z)$$

Answer: { john }

To obtain this answer, we need to **reason by cases**.

Query answering

Certain answers

Query answering amounts to finding the **certain answers** $cert(q, \mathcal{O})$ to a query $q(\vec{x})$, i.e., those answers that hold in all models of the OBDA system $\mathcal{O}$.

Two borderline cases for the language to use for querying ontologies:

1. Use the ontology language as query language.
   - Ontology languages are tailored for capturing intensional relationships.
   - They are quite **poor as query languages**.
2. Full SQL (or equivalently, first-order logic).
   - Problem: in the presence of incomplete information, query answering becomes **undecidable** (FOL validity).

A good tradeoff is to use **conjunctive queries** (CQs) or unions of CQs (UCQs), corresponding to SQL/relational algebra **(union) select-project-join queries**.

unibz.it

Complexity of conjunctive query answering in DLs

Studied extensively for various ontology languages:

	Combined complexity	Data complexity
Plain databases	NP-complete	in AC^0 (1)
Expressive DLs	$\geq$ 2ExpTime (2)	coNP-hard (3)

(1) This is what we need to scale with the data.

(2) Hardness by [Lutz, 2008; Eiter *et al.*, 2009].
Tight upper bounds obtained for a variety of expressive DLs [C. *et al.*, 1998; Levy and Rousset, 1998; C. *et al.*, 2007c; C. *et al.*, 2008c; Glimm *et al.*, 2008b; Glimm *et al.*, 2008a; Lutz, 2008; Eiter *et al.*, 2008].

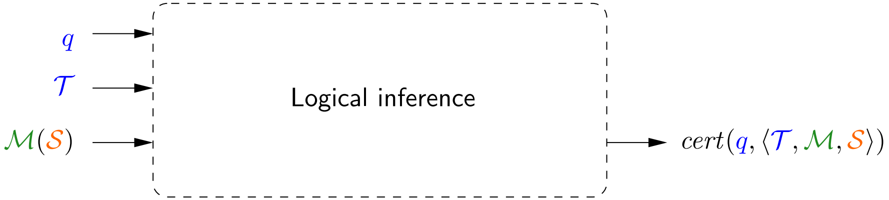
(3) Already for an ontology with a single axiom involving disjunction.
However, the complexity does not increase even for very expressive DLs [Ortiz *et al.*, 2006; Ortiz *et al.*, 2008; Glimm *et al.*, 2008a].

Challenges for query answering in the OBDA setting

Challenges

- Can we find interesting ontology languages for which query answering in OBDA can be done efficiently (i.e., in AC^0)?
- If yes, can we delegate query answering in OBDA to a relational engine?
- If yes, can we obtain acceptable performance in practical scenarios involving large ontologies and large amounts of data?

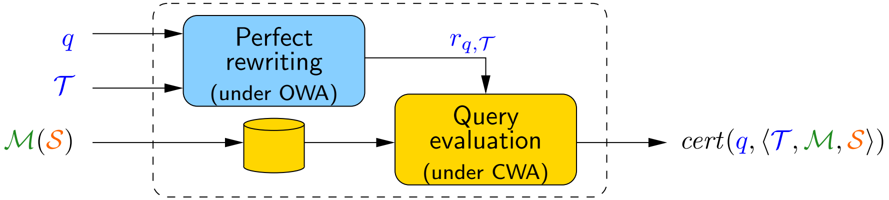
Logical inference for query answering


q
$\mathcal{T}$
$\mathcal{M}(\mathcal{S})$
Logical inference
$cert(q, \langle \mathcal{T}, \mathcal{M}, \mathcal{S} \rangle)$

To be able to deal with data efficiently, we need to separate the contribution of the data $\mathcal{S}$ (accessed via the mapping $\mathcal{M}$) from the contribution of q and $\mathcal{O}$.

$\rightsquigarrow$ Query answering by **query rewriting**.

Query answering by rewriting


q
$\mathcal{T}$
$\mathcal{M}(\mathcal{S})$
Perfect rewriting (under OWA)
$r_{q,\mathcal{T}}$
Query evaluation (under CWA)
$cert(q, \langle\mathcal{T}, \mathcal{M}, \mathcal{S}\rangle)$

Query answering can **always** be thought as done in two phases:

1. **Perfect rewriting**: produce from q and the ontology TBox $\mathcal{T}$ a new query $r_{q,\mathcal{T}}$ (called the perfect rewriting of q w.r.t. $\mathcal{T}$).
2. **Query evaluation**: evaluate $r_{q,\mathcal{T}}$ over $\mathcal{M}(\mathcal{S})$ seen as a complete database (and without considering $\mathcal{T}$).
   $\rightsquigarrow$ Produces $cert(q, \langle\mathcal{T}, \mathcal{M}, \mathcal{S}\rangle)$.

Note: The "always" holds if we pose no restriction on the language in which to express the rewriting $r_{q,\mathcal{T}}$.

$\mathcal{L}_Q$-rewritability

Let:

- $\mathcal{L}_Q$ be a class of queries (i.e., a query language), and
- $\mathcal{L}_T$ be an ontology TBox language.

Def.: $\mathcal{L}_Q$-**rewritability** of (conjunctive) query answering

(Conjunctive) query answering is $\mathcal{L}_Q$-**rewritable** in $\mathcal{L}_T$, if for every TBox $\mathcal{T}$ of $\mathcal{L}_T$ and for every (conjunctive) query q, the perfect rewriting $r_{q,\mathcal{T}}$ of q w.r.t. $\mathcal{T}$ can be expressed in $\mathcal{L}_Q$.

Note: When the only relevant measure is the size of the data $\mathcal{M}(\mathcal{S})$, then

$$\text{complexity of computing } cert(q, \langle\mathcal{T}, \mathcal{M}, \mathcal{S}\rangle) \\ = \\ \text{complexity of evaluating } r_{q,\mathcal{T}} \text{ over } \mathcal{M}(\mathcal{S})$$

Hence, $\mathcal{L}_Q$-**rewritability** is tightly related to the **data complexity of evaluating queries** expressed in the language $\mathcal{L}_Q$.

Language of the rewriting

The **expressiveness of the ontology language affects the rewriting language**, i.e., the language into which we are able to rewrite (U)CQs:

- When we can rewrite into FOL/SQL.
  $\rightsquigarrow$ Query evaluation can be done in SQL, i.e., via an RDBMS
  (*Note:* FOL is in AC^0).
- When we can rewrite into UCQs.
  $\rightsquigarrow$ Query evaluation can be "optimized" via an RDBMS.
- When we can rewrite into non-recursive Datalog.
  $\rightsquigarrow$ Query evaluation can be done via an RDBMS, but using views.
- When we need an NLogSpace-hard language to express the rewriting.
  $\rightsquigarrow$ Query evaluation requires (at least) linear recursion.
- When we need a PTime-hard language to express the rewriting.
  $\rightsquigarrow$ Query evaluation requires full recursion (e.g., Datalog).
- When we need a coNP-hard language to express the rewriting.
  $\rightsquigarrow$ Query evaluation requires (at least) the power of Disjunctive Datalog.

Outline

Description Logics

- **Description Logics (DLs)** stem from early days (70') KR formalisms, and assumed their current form in the late 80's & 90's.
- Are **logics** specifically designed to represent and reason on structured knowledge.
- Technically they can be considered as well-behaved (i.e., decidable) **fragments of first-order logic**.
- Semantics given in terms of first-order interpretations.
- Come in hundreds of variations, with different semantic and computational properties.
- Strongly influenced the W3C standard Web Ontology Language OWL.

The *DL-Lite* family

- A family of DLs optimized according to the tradeoff between expressive power and **complexity** of query answering, with emphasis on **data**.
  - The same complexity as relational databases.
  - In fact, **query answering is FOL-rewritable** and hence can be delegated to a relational DB engine.
  - The DLs of the *DL-Lite* family are essentially the maximally expressive DLs enjoying these nice computational properties.
- Nevertheless they have the "right" expressive power: capture the essential features of conceptual modeling formalisms.

DL-Lite provides robust foundations for Ontology-Based Data Access.

Note:

- The *DL-Lite* family is at the basis of the OWL 2 QL profile of the W3C standard Web Ontology Language OWL.
- More recently, the *DL-Lite* family has been extended towards n-ary relations and with additional features (see, e.g., [Calì *et al.*, 2009; Baget *et al.*, 2011; Gottlob and Schwentick, 2012]).

unibz.it

DL-Lite ontologies (essential features)

Concept and role language:

- Roles R: either atomic: P
  or an inverse role: P^-
- Concepts C: either atomic: A
  or the projection of a role on one component: $\exists P, \ \exists P^-$

TBox assertions:

Role inclusion:	$R_1 \sqsubseteq R_2$	Concept inclusion:	$C_1 \sqsubseteq C_2$
Role disjointness:	$R_1 \sqsubseteq \neg R_2$	Concept disjointness:	$C_1 \sqsubseteq \neg C_2$
Role functionality:	$(\textbf{funct } R)$		

ABox assertions: $A(c)$, $P(c_1, c_2)$, with c_1, c_2 constants

Note: DL-Lite distinguishes also between abstract objects and data values (ignored here).

Semantics of *DL-Lite*

DL-Lite (as all DLs) is equipped with a set-theoretic semantics.

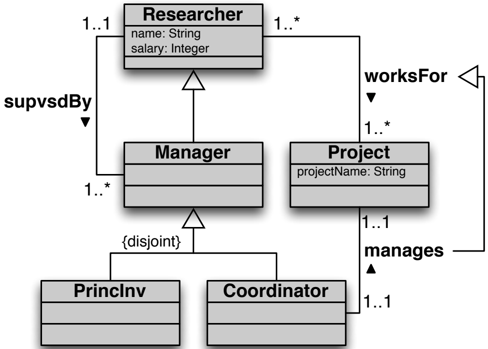
Construct	Syntax	Example	Semantics
atomic role	P	manages	$P^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
inverse role	P^{-}	manages^{-}	$\{(o, o') \mid (o', o) \in P^{\mathcal{I}}\}$
role negation	$\neg R$	$\neg$worksFor	$(\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}) \setminus R^{\mathcal{I}}$
atomic concept	A	Researcher	$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
existential restriction	$\exists R$	$\exists$worksFor^{-}	$\{o \mid \exists o'. (o, o') \in R^{\mathcal{I}}\}$
concept negation	$\neg C$	$\neg\exists$worksFor	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
concept incl.	$C_1 \sqsubseteq C_2$	Project $\sqsubseteq$ $\exists$worksFor^{-}	$C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$
role incl.	$R_1 \sqsubseteq R_2$	manages $\sqsubseteq$ worksFor	$R_1^{\mathcal{I}} \subseteq R_2^{\mathcal{I}}$
role funct.	$(\textbf{funct}\ R)$	(**funct** manages)	$\forall o, o, o''.(o, o') \in R^{\mathcal{I}} \wedge (o, o'') \in R^{\mathcal{I}} \rightarrow o' = o''$
mem. asser.	$A(c)$	Manager(ann)	$c^{\mathcal{I}} \in A^{\mathcal{I}}$
mem. asser.	$P(c_1, c_2)$	supvsdBy(bob, ann)	$(c_1^{\mathcal{I}}, c_2^{\mathcal{I}}) \in P^{\mathcal{I}}$

Note: We make the **unique-name assumption**, i.e., different constants denote different objects.

DL-Lite captures conceptual modeling formalisms

Modeling construct	*DL-Lite*	FOL formalization
ISA on classes	$A_1 \sqsubseteq A_2$	$\forall x(A_1(x) \rightarrow A_2(x))$
Disjointness of classes	$A_1 \sqsubseteq \neg A_2$	$\forall x(A_1(x) \rightarrow \neg A_2(x))$
Domain of relations	$\exists P \sqsubseteq A_1$	$\forall x(\exists y(P(x,y)) \rightarrow A_1(x))$
Range of relations	$\exists P^- \sqsubseteq A_2$	$\forall x(\exists y(P(y,x)) \rightarrow A_2(x))$
Mandatory participation *(min card = 1)*	$A_1 \sqsubseteq \exists P$ $A_2 \sqsubseteq \exists P^-$	$\forall x(A_1(x) \rightarrow \exists y(P(x,y)))$ $\forall x(A_2(x) \rightarrow \exists y(P(y,x)))$
Functionality *(max card = 1)*	$(\textbf{funct } P)$ $(\textbf{funct } P^-)$	$\forall x,y,y'(P(x,y) \wedge P(x,y') \rightarrow y = y')$ $\forall x,x',y(P(x,y) \wedge P(x',y) \rightarrow x = x')$
ISA on relations	$R_1 \sqsubseteq R_2$	$\forall x,y(R_1(x,y) \rightarrow R_2(x,y))$
Disjointness of relations	$R_1 \sqsubseteq \neg R_2$	$\forall x,y(R_1(x,y) \rightarrow \neg R_2(x,y))$
...	...	...

Capturing UML class diagrams/ER schemas in *DL-Lite*



$$
\begin{array}{rcl}
\text{Manager} & \sqsubseteq & \text{Researcher} \\
\text{PrincInv} & \sqsubseteq & \text{Manager} \\
\text{Coordinator} & \sqsubseteq & \text{Manager} \\
\text{PrincInv} & \sqsubseteq & \neg\text{Coordinator} \\
\\
\text{Researcher} & \sqsubseteq & \exists \mathit{salary} \\
\exists \mathit{salary}^- & \sqsubseteq & \texttt{xsd:int} \\
 & (\textbf{funct}\ \mathit{salary}) & \\
\\
\exists\text{worksFor} & \sqsubseteq & \text{Researcher} \\
\exists\text{worksFor}^- & \sqsubseteq & \text{Project} \\
\text{Researcher} & \sqsubseteq & \exists\text{worksFor} \\
\text{Project} & \sqsubseteq & \exists\text{worksFor}^- \\
\\
\exists\text{manages} & \sqsubseteq & \text{Coordinator} \\
\exists\text{manages}^- & \sqsubseteq & \text{Project} \\
\text{Coordinator} & \sqsubseteq & \exists\text{manages} \\
\text{Project} & \sqsubseteq & \exists\text{manages}^- \\
\text{manages} & \sqsubseteq & \text{worksFor} \\
 & (\textbf{funct}\ \text{manages}) & \\
 & (\textbf{funct}\ \text{manages}^-) & \\
 & \cdots &
\end{array}
$$

Note: *DL-Lite* cannot capture completeness of a hierarchy. This would require **disjunction** (i.e., **OR**).

Query answering in *DL-Lite*

Based on query rewriting: given a (U)CQ q and an ontology $\mathcal{O} = \langle \mathcal{T}, \mathcal{A} \rangle$:

1. **Compute the perfect rewriting of q w.r.t. $\mathcal{T}$**, which is a FOL query.
2. **Evaluate the perfect rewriting** over $\mathcal{A}$. (Ignore the mapping for now.)

To compute the perfect rewriting, starting from q, iteratively get a CQ q' to be processed and either:

- **expand an atom** of q' using an inclusion axiom, or
- **unify** atoms in q' to obtain a more specific CQ to be further expanded.

Each result of the above steps is added to the queries to be processed.

We can restrict expansion and unification so as to ensure termination without losing completeness [C. *et al.*, 2007a].

Note: negative inclusions and functionalities play a role in ontology satisfiability, but can be ignored during query answering (i.e., we have **separability**).

Query answering in *DL-Lite* – Example

TBox:

$\text{Coordinator} \sqsubseteq \text{Researcher}$	$\forall x(\text{Coordinator}(x) \rightarrow \text{Researcher}(x))$
$\text{Researcher} \sqsubseteq \exists\text{worksFor}$	$\forall x(\text{Researcher}(x) \rightarrow \exists y(\text{worksFor}(x,y)))$
$\exists\text{worksFor}^- \sqsubseteq \text{Project}$	$\forall x(\exists y(\text{worksFor}(y,x)) \rightarrow \text{Project}(x))$

Query: $\text{q}(x) \leftarrow \text{worksFor}(x,y), \text{Project}(y)$

Perfect rewriting:
$\text{q}(x) \leftarrow \text{worksFor}(x,y), \text{Project}(y)$
$\text{q}(x) \leftarrow \text{worksFor}(x,y), \text{worksFor}(\_,y)$
$\text{q}(x) \leftarrow \text{worksFor}(x,\_)$
$\text{q}(x) \leftarrow \text{Researcher}(x)$
$\text{q}(x) \leftarrow \text{Coordinator}(x)$

ABox:

$\text{worksFor}(\texttt{serge}, \texttt{webdam})$	$\text{Coordinator}(\texttt{serge})$
$\text{worksFor}(\texttt{georg}, \texttt{diadem})$	$\text{Coordinator}(\texttt{marie})$

Evaluating the perfect rewriting over the ABox (seen as a DB) produces as answer $\{\texttt{serge},\ \texttt{georg},\ \texttt{marie}\}$.

Complexity of query answering in *DL-Lite*

Ontology satisfiability and all classical DL reasoning tasks are:

- Efficiently tractable in the size of the TBox (i.e., PTIME).
- Very efficiently tractable in the size of the ABox (i.e., AC^0).

In fact, reasoning can be done by constructing suitable FOL/SQL queries and evaluating them over the ABox (FOL-rewritability).

Query answering for CQs and UCQs is:

- PTIME in the size of the TBox.
- AC^0 in the size of the ABox.
- Exponential in the size of the **query**, more precisely NP-complete.

  In theory this is not bad, since this is precisely the complexity of evaluating CQs in plain relational DBs.

Tracing the expressivity boundary

	Lhs concept	Rhs concept	funct.	Relation incl.	Data complexity of query answering
0	*DL-Lite*		$\surd^*$	$\surd^*$	in AC^0
1	$A \mid \exists P.A$	A	–	–	NLogSpace-hard
2	A	$A \mid \forall P.A$	–	–	NLogSpace-hard
3	A	$A \mid \exists P.A$	$\surd$	–	NLogSpace-hard
4	$A \mid \exists P.A \mid A_1 \sqcap A_2$	A	–	–	PTime-hard
5	$A \mid A_1 \sqcap A_2$	$A \mid \forall P.A$	–	–	PTime-hard
6	$A \mid A_1 \sqcap A_2$	$A \mid \exists P.A$	$\surd$	–	PTime-hard
7	$A \mid \exists P.A \mid \exists P^-.A$	$A \mid \exists P$	–	–	PTime-hard
8	$A \mid \exists P \mid \exists P^-$	$A \mid \exists P \mid \exists P^-$	$\surd$	$\surd$	PTime-hard
9	$A \mid \neg A$	A	–	–	coNP-**hard**
10	A	$A \mid A_1 \sqcup A_2$	–	–	coNP-**hard**
11	$A \mid \forall P.A$	A	–	–	coNP-**hard**

From [C. *et al.*, 2006; Artale *et al.*, 2009].

Notes:

- Data complexity beyond AC^0 means that query answering in **not FOL rewritable**, hence cannot be delegated to a relational DBMS.
- These results pose strict bounds on the expressive power of the ontology language that can be used in OBDA.

Outline

Experimentations and experiences

Several experimentations:

- Monte dei Paschi di Siena (led by Sapienza Univ. of Rome)
- Selex: world leading radar producer
- National Accessibility Portal of South Africa
- Horizontal Gene Transfer data and ontology
- Stanford's "Resource Index" comprising 200 ontologies from BioPortal
- Experiments on artificial data ongoing

Observations:

- Approach highly effective for bridging impedance mismatch.
- Rewriting technique effective against incompleteness in the data.

However, performance is a major issue that still prevents large-scale deployment of this technology.

Size of the rewriting

Example

TBox $\mathcal{T}$:

$$\begin{tikzcd}
 & & & A & & \\
 & B \arrow[urr] & & & C \arrow[ul] & \\
D \arrow[ur] & E \arrow[u] & F \arrow[ul] & G \arrow[ur] & H \arrow[u] & I \arrow[ul]
\end{tikzcd}$$

$q(x) \leftarrow A(x), P(x,y), A(y), P(y,z), A(z)$

UCQ rewriting of q w.r.t. $\mathcal{T}$ contains 729 CQs
i.e., a UNION of 729 SPJ SQL queries

The size of UCQ rewritings may become very large

- In the worst case, it may be $O((|\mathcal{T}| \cdot |q|)^{|q|})$, i.e., **exponential in $|q|$**.
- Unfortunately, this **blowup occurs also in practice**.

unibz.it

Taming the size of the rewriting

Note: It is not possible to avoid rewriting altogether, since this would require in general to materialize an infinite database [C. *et al.*, 2007a].

Several techniques have been proposed recently to limit the size of the rewriting:

- Alternative rewriting techniques [Pérez-Urbina *et al.*, 2010]: more efficient algorithm based on resolution, but produces still an exponential UCQ.
- Combined approach [Kontchakov *et al.*, 2010]: combines partial materialization with rewriting:
  - When $\mathcal{T}$ contains no role inclusions rewriting is polynomial.
  - But in general rewriting is exponential.
  - Materialization requires control over the data sources and might not be applicable in an OBDA setting.
- Rewriting into non-recursive Datalog:
  - Presto system [Rosati and Almatelli, 2010]: still worst-case exponential.
  - Polynomial rewriting for Datalog$^{\pm}$ [Gottlob and Schwentick, 2012]: rewriting uses polynomially many new existential variables and "guesses" a relevant portion of the canonical model for the TBox.

See [Kikot *et al.*, 2012a; Kikot *et al.*, 2012b] for discussion and further results.

A holistic approach to optimization

Recall our main objective

Given an OBDA system $\mathcal{O} = \langle \mathcal{T}, \mathcal{M}, \mathcal{S} \rangle$ and a set of queries, **compute the certain answers** of such queries w.r.t. $\mathcal{O}$ **as efficiently as possible**.

Observe:

- The size of the rewriting is only one coordinate in the problem space.
- Optimizing rewriting is necessary but not sufficient, since the more compact rewritings are much more difficult to evaluate.
- In fact, the **efficiency of the query evaluation by the DBMS** is the crucial factor.

Hence, a **holistic approach** is required, that considers all components of an OBDA system, i.e.:

- the TBox $\mathcal{T}$,
- the mappings $\mathcal{M}$,
- the data sources $\mathcal{S}$ with their dependencies, and
- the query load.

unibz.it

(Virtual) ABox dependencies

As in databases, we can exploit **dependencies on the data and on the ABox** to optimize query processing.

ABox dependencies express **conditions on the data in the (virtual) ABox**.

- Syntax: $C_1 \preceq C_2$ $R_1 \preceq R_2$
- Meaning: constrain the assertions present in the ABox
  - $\mathcal{A} \vdash A_1 \preceq A_2$ iff $A_1(d) \in \mathcal{A}$ implies $A_2(d) \in \mathcal{A}$
  - $\mathcal{A} \vdash A \preceq \exists P$ iff $A(d) \in \mathcal{A}$ implies $P(d, d') \in \mathcal{A}$ for some d'
  - $\mathcal{A} \vdash P_1 \preceq P_2$ iff $P_1(d, d') \in \mathcal{A}$ implies $P_2(d, d') \in \mathcal{A}$
  - $\cdots$

Note: ABox dependencies are fundamentally different from TBox assertions. They constrain the syntactic level (the ABox itself), and not the models.

Exploiting ABox/data dependencies in OBDA

In an OBDA system, ABox/data dependencies have an impact that spans all system components:

- Dependencies on the sources $\mathcal{S}$ induce via the mapping $\mathcal{M}$ dependencies on the virtual ABox $\mathcal{V} = \mathcal{M}(\mathcal{S})$.
- The mapping itself in general induces additional dependencies on $\mathcal{V}$ (that do not directly depend on $\mathcal{S}$).
- Dependencies on $\mathcal{V}$ interact with the TBox $\mathcal{T}$, and such interaction can be exploited for optimization.

OBDA optimization techniques

We exploit data dependencies in the following optimization techniques:

1. Eliminating redundant TBox assertions [Rodriguez-Muro and C., 2012].
2. Semantic Index to efficiently deal with concept hierarchies [Rodriguez-Muro and C., 2012].
3. Optimize query rewriting algorithm so as to eliminate rewritings that are redundant w.r.t. query containment under dependencies [Rodríguez-Muro and C., 2011; Rosati, 2012].

Eliminating redundant TBox assertions

TBox optimization is based on a characterization of assertions in a TBox $\mathcal{T}$ that are **redundant wrt a set Σ of ABox dependencies**.

Example (Direct redundancy)

Let $\mathcal{T}$ be:

$$\begin{tikzcd} \text{Human} \\ \text{Person} \arrow[u] \\ \exists\text{hasFather} \arrow[u, dotted] \end{tikzcd}$$

Let Σ be:

$$\begin{tikzcd} \text{Human} \\ \text{Person} \arrow[u, twoheadrightarrow] \\ \exists\text{hasFather} \arrow[u, dotted, twoheadrightarrow] \end{tikzcd}$$

Note: Σ may enforce e.g., that

$$\text{hasFather}(\text{luisa}, \text{franz}) \in \mathcal{A} \quad \text{implies} \quad \text{Human}(\text{luisa}) \in \mathcal{A}.$$

Then $\text{Person} \sqsubseteq \text{Human}$ is **redundant** in $\mathcal{T}$.

The overall characterization of redundant TBox assertions is more involved (see [Rodriguez-Muro and C., 2012]).

Computing an optimized TBox

Given a TBox $\mathcal{T}$ and a set Σ of ABox dependencies:

1. Compute the deductive closure $\mathcal{T}_{cl}$ of $\mathcal{T}$ (at most quadratic in size of $\mathcal{T}$).
2. Compute the deductive closure Σ_{cl} of Σ (at most quadratic in size of Σ).
3. Eliminate from $\mathcal{T}_{cl}$ all TBox assertions redundant wrt Σ_{cl}, obtaining $\mathcal{T}_{opt}$.

Notes:

- $\mathcal{T}_{opt}$ can be computed in polynomial time in the size of $\mathcal{T}$ and Σ.
- $\mathcal{T}_{opt}$ might be much smaller than $\mathcal{T}$.

Theorem

For every (virtual) ABox $\mathcal{A}$ satisfying Σ and for every UCQ q, we have that

$$cert(q, \langle\mathcal{T}, \mathcal{A}\rangle) = cert(q, \langle\mathcal{T}_{opt}, \mathcal{A}\rangle).$$

Hence, $\mathcal{T}_{opt}$ can be used instead of $\mathcal{T}$ independently of the adopted query rewriting method (provided the ABox satisfies Σ).

Deriving ABox dependencies

Derived from dependencies on the data in $\mathcal{S}$:

Example

Suppose we have two tables in $\mathcal{S}$:

ResT[SSN: String, ...]	stores data about researchers
ManT[SSN: String, ...]	stores data about managers

Consider the following mapping $\mathcal{M}$:

m_1: `SELECT SSN FROM ResT` $\rightsquigarrow$ Researcher(**pers**(SSN))
m_2: `SELECT SSN FROM ManT` $\rightsquigarrow$ Manager(**pers**(SSN))

If $\mathcal{S}$ satisfies the inclusion dependency ManT[SSN] $\subseteq$ ResT[SSN],
then $\mathcal{M}(\mathcal{S})$ satisfies the dependency Manager $\preceq$ Researcher.

Deriving ABox dependencies (cont'd)

Induced by the form of the data in $\mathcal{S}$ and the mapping $\mathcal{M}$:

Example

Suppose that in $\mathcal{S}$ we have one table: $\mathsf{ResT}[\mathit{SSN}\colon \mathsf{String}, \mathit{Level}\colon \mathsf{Boolean}, \ldots]$

- Stores data about researchers (including managers).
- For managers the value of *Level* is `true`, otherwise it is `false`.

Consider the following mapping $\mathcal{M}$:

m_1: `SELECT SSN FROM ResT` $\rightsquigarrow$ Researcher(**pers**(*SSN*))

m_2: `SELECT SSN FROM ResT WHERE Level='true'` $\rightsquigarrow$ Manager(**pers**(*SSN*))

We have that $\mathcal{M}(\mathcal{S})$ satisfies the dependency Manager $\preceq$ Researcher.
This holds since the lhs query of m_2 is contained in the lhs query of m_1 (in the traditional sense of query containment in DBs).

This situation corresponds to a natural way of constructing mappings, and is very common in OBDA systems.

Deriving ABox dependencies (cont'd)

We can use the mapping to enforce dependencies "corresponding" to the TBox assertions:

Example

Suppose that in $\mathcal{S}$ we have one table: $\mathsf{ResT}[\mathit{SSN}\colon \mathsf{String}, \mathit{Type}\colon \mathsf{Char}, \ldots]$

- Stores data about researchers of all types.
- The value of *Type* encodes the type or researcher: 'm' for managers, 'p' for principal investigators, 'c' for coordinators, and 'r' for other researchers.

We can define a mapping $\mathcal{M}$ that induces suitable dependencies:

m_1: `SELECT SSN FROM ResT` $\rightsquigarrow$ $\mathsf{Researcher}(\mathbf{pers}(\mathit{SSN}))$

m_2: `SELECT SSN FROM ResT WHERE Type='p'` $\rightsquigarrow$ $\mathsf{PrincInv}(\mathbf{pers}(\mathit{SSN}))$

m_3: `SELECT SSN FROM ResT WHERE Type='c'` $\rightsquigarrow$ $\mathsf{Coordinator}(\mathbf{pers}(\mathit{SSN}))$

m_4: `SELECT SSN FROM ResT WHERE Type='m' OR Type='p' OR Type='c'` $\rightsquigarrow$ $\mathsf{Manager}(\mathbf{pers}(\mathit{SSN}))$

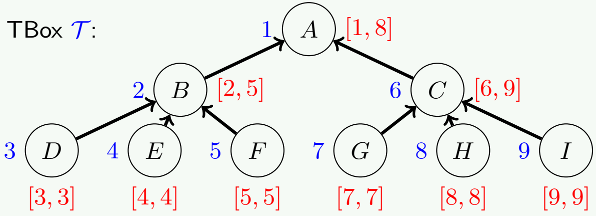
We have that $\mathcal{M}(\mathcal{S})$ satisfies e.g., the dependency $\mathsf{PrincInv} \preceq \mathsf{Manager}$.

Semantic Index

Storage technique that allows for efficient querying of (concept) hierachies.

Example

TBox $\mathcal{T}$:



ABox
$\mathcal{A} = \{B(c), E(c), H(d)\}$

Stored as:

Obj	Idx
c	2
c	4
d	8

1. We associate to each concept in $\mathcal{T}$ a numeric index, essentially by doing a breadth-first visit of the concept hierarchy.
2. We associate to each concept a (set of) interval(s), such that the interval(s) of a concept covers the indexes of all its subconcepts.
3. As ABox, we maintain a single table for all concepts, which associates to each individual c the numeric indexes of the concepts of which c is an instance.

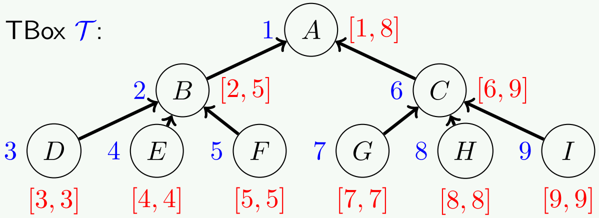
Note: Similar ideas have been applied previously in different contexts [Agrawal *et al.*, 1989; DeHaan *et al.*, 2003].

Querying with a Semantic Index

Queries over concept hierarchies can now be expressed as simple range queries.

Example

TBox $\mathcal{T}$:



ABox
$\mathcal{A} = \{B(v), E(w), H(v)\}$

Stored as:

Obj	Idx
v	2
w	4
v	8

For example, to obtain the instances of concept B, we can pose the query:

```
SELECT Obj FROM ConceptTable WHERE Idx >= 2 AND Idx <=5
```

- We can construct a Semantic Index in polynomial time in the size of $\mathcal{T}$.
- We can also use the range queries associated to concepts to define the mappings from the data to the ontology.
- In this way, data sources maintaining objects organized in large hierarchies can be queried very efficiently.

Experimentation using Stanford's "Resource Index"

- Semantic search over textual documents annotated with concepts from bio-ontologies.
- 200 ontologies from Bio-portal:
  - ontology language is very simple: only concept hierarchies
  - very large ontology: 200K concepts, millions of subclass relations
- Current system uses forward chaining:
  - Naive chase: 7 days
  - Optimized chase: 40 mins
  - Storage cost: 16GB base data + 70GB of derived data
  - Split second responses
- Pure query rewriting based approaches cannot cope with the size of the ontology.
- Semantic index-based rewriting:
  - DAG computation and indexing: 5 mins
  - Cost: 16 GB (no additional storage needed)
  - Rewriting reduces to a single query, split second responses

Outline

Conclusions

- Ontology-based data access are challenging problems with great practical relevance.
- In this setting, the size of the data is a critical parameter that must guide technological choices.
- Theoretical foundations provide a solid basis for system development.
- Practical deployment of this technology in real world scenarios is ongoing, but requires further research.
- Adoption of a holistic approach, considering all components of OBDA systems seems the only way to cope with real-world challenges.

Further research directions

- Extensions of the ontology languages, e.g., towards n-ary relations [Calì *et al.*, 2009; Baget *et al.*, 2011; Gottlob and Schwentick, 2012].
- Dealing with inconsistency in the ontology.
- Ontology-based update.
- Coping with evolution of data in the presence of ontological constraints.
- Dealing with different kinds of data, besides relational sources: XML, graph-structured data, RDF and linked data –
- Close connection to work carried out in the Semantic Web on Triple Stores.

Some advertising

~~ontop~~ OBDA framework developed at the Free Univ. of Bozen-Bolzano.

`http://obda.inf.unibz.it/protege-plugin/`

New EU IP Project starting in Nov. 2012 Optique

unibz.it

Optique use cases

Statoil Exploration

- OBDA technologies developed in Optique could free as much as 30% of experts' time.
- Potential savings: €70,000,000 per year

Siemens Energy Services

- OBDA technologies developed in Optique could reduce data acquisition time by at least 25%.
- Potential savings: €1,000,000/year/service-center
  = €50,000,000/year

Thanks

Thanks to the many people that contributed to this work:

- Alessandro Artale
- Elena Botoeva
- Giuseppe De Giacomo
- Roman Kontschakov
- Domenico Lembo
- Maurizio Lenzerini
- Antonella Poggi
- Mariano Rodriguez Muro
- Riccardo Rosati
- Michael Zakhariaschev
- many students

References I

[Agrawal *et al.*, 1989] Rakesh Agrawal, Alexander Borgida, and H. V. Jagadish.
Efficient management of transitive relationships in large data and knowledge bases.
In *Proc. of the ACM SIGMOD Int. Conf. on Management of Data*, pages 253–262, 1989.

[Artale *et al.*, 2009] Alessandro Artale, Diego C., Roman Kontchakov, and Michael Zakharyaschev.
The *DL-Lite* family and relations.
J. of Artificial Intelligence Research, 36:1–69, 2009.

[Baget *et al.*, 2011] Jean-François Baget, Michel Leclère, Marie-Laure Mugnier, and Eric Salvat.
On rules with existential variables: Walking the decidability line.
Artificial Intelligence, 175(9–10):1620–1654, 2011.

[Berardi *et al.*, 2005] Daniela Berardi, Diego C., and Giuseppe De Giacomo.
Reasoning on UML class diagrams.
Artificial Intelligence, 168(1–2):70–118, 2005.

[Bergamaschi and Sartori, 1992] Sonia Bergamaschi and Claudio Sartori.
On taxonomic reasoning in conceptual design.
ACM Trans. on Database Systems, 17(3):385–422, 1992.

References II

[Borgida and Brachman, 2003] Alexander Borgida and Ronald J. Brachman.
Conceptual modeling with description logics.
In Franz Baader, Diego C., Deborah McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors, *The Description Logic Handbook: Theory, Implementation and Applications*, chapter 10, pages 349–372. Cambridge University Press, 2003.

[Borgida, 1995] Alexander Borgida.
Description logics in data management.
IEEE Trans. on Knowledge and Data Engineering, 7(5):671–682, 1995.

[C. *et al.*, 1998] Diego C., Giuseppe De Giacomo, and Maurizio Lenzerini.
On the decidability of query containment under constraints.
In *Proc. of the 17th ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems (PODS'98)*, pages 149–158, 1998.

[C. *et al.*, 1999] Diego C., Maurizio Lenzerini, and Daniele Nardi.
Unifying class-based representation formalisms.
J. of Artificial Intelligence Research, 11:199–240, 1999.

References III

[C. *et al.*, 2004] Diego C., Giuseppe De Giacomo, Maurizio Lenzerini, Riccardo Rosati, and Guido Vetere.
DL-Lite: Practical reasoning for rich DLs.
In *Proc. of the 17th Int. Workshop on Description Logic (DL 2004)*, volume 104 of *CEUR Electronic Workshop Proceedings*, `http://ceur-ws.org/`, 2004.

[C. *et al.*, 2005a] Diego C., Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati.
Tailoring OWL for data intensive ontologies.
In *Proc. of the 1st Int. Workshop on OWL: Experiences and Directions (OWLED 2005)*, volume 188 of *CEUR Electronic Workshop Proceedings*, `http://ceur-ws.org/`, 2005.

[C. *et al.*, 2005b] Diego C., Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati.
DL-Lite: Tractable description logics for ontologies.
In *Proc. of the 20th Nat. Conf. on Artificial Intelligence (AAAI 2005)*, pages 602–607, 2005.

References IV

[C. *et al.*, 2006] Diego C., Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati.
Data complexity of query answering in description logics.
In *Proc. of the 10th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2006)*, pages 260–270, 2006.

[C. *et al.*, 2007a] Diego C., Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati.
Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family.
J. of Automated Reasoning, 39(3):385–429, 2007.

[C. *et al.*, 2007b] Diego C., Giuseppe De Giacomo, Maurizio Lenzerini, and Riccardo Rosati.
Actions and programs over description logic ontologies.
In *Proc. of the 20th Int. Workshop on Description Logic (DL 2007)*, volume 250 of *CEUR Electronic Workshop Proceedings*, `http://ceur-ws.org/`, pages 29–40, 2007.

[C. *et al.*, 2007c] Diego C., Thomas Eiter, and Magdalena Ortiz.
Answering regular path queries in expressive description logics: An automata-theoretic approach.
In *Proc. of the 22nd AAAI Conf. on Artificial Intelligence (AAAI 2007)*, pages 391–396, 2007.

References V

[C. *et al.*, 2008a] Diego C., Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, Antonella Poggi, Riccardo Rosati, and Marco Ruzzi.
Data integration through *DL-Lite*$_{\mathcal{A}}$ ontologies.
In Klaus-Dieter Schewe and Bernhard Thalheim, editors, *Revised Selected Papers of the 3rd Int. Workshop on Semantics in Data and Knowledge Bases (SDKB 2008)*, volume 4925 of *Lecture Notes in Computer Science*, pages 26–47. Springer, 2008.

[C. *et al.*, 2008b] Diego C., Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati.
Path-based identification constraints in description logics.
In *Proc. of the 11th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2008)*, pages 231–241, 2008.

[C. *et al.*, 2008c] Diego C., Giuseppe De Giacomo, and Maurizio Lenzerini.
Conjunctive query containment and answering under description logics constraints.
ACM Trans. on Computational Logic, 9(3):22.1–22.31, 2008.

[Calì *et al.*, 2009] Andrea Calì, Georg Gottlob, and Thomas Lukasiewicz.
Datalog$^{\pm}$: a unified approach to ontologies and integrity constraints.
In *Proc. of the 12th Int. Conf. on Database Theory (ICDT 2009)*, pages 14–30, 2009.

unibz.it

References VI

[DeHaan *et al.*, 2003] David DeHaan, David Toman, Mariano P. Consens, and M. Tamer Özsu.
A comprehensive XQuery to SQL translation using dynamic interval encoding.
In *Proc. of the ACM SIGMOD Int. Conf. on Management of Data*, pages 623–634, 2003.

[Eiter *et al.*, 2008] Thomas Eiter, Georg Gottlob, Magdalena Ortiz, and Mantas Šimkus.
Query answering in the description logic Horn-$\mathcal{SHIQ}$.
In *Proc. of the 11th Eur. Conference on Logics in Artificial Intelligence (JELIA 2008)*, pages 166–179, 2008.

[Eiter *et al.*, 2009] Thomas Eiter, Carsten Lutz, Magdalena Ortiz, and Mantas Šimkus.
Query answering in description logics with transitive roles.
In *Proc. of the 21st Int. Joint Conf. on Artificial Intelligence (IJCAI 2009)*, pages 759–764, 2009.

[Glimm *et al.*, 2008a] Birte Glimm, Ian Horrocks, Carsten Lutz, and Uli Sattler.
Conjunctive query answering for the description logic $\mathcal{SHIQ}$.
J. of Artificial Intelligence Research, 31:151–198, 2008.

References VII

[Glimm *et al.*, 2008b] Birte Glimm, Ian Horrocks, and Ulrike Sattler.
Unions of conjunctive queries in $\mathcal{SHOQ}$.
In *Proc. of the 11th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2008)*, pages 252–262, 2008.

[Gottlob and Schwentick, 2012] Georg Gottlob and Thomas Schwentick.
Rewriting ontological queries into small nonrecursive Datalog programs.
In *Proc. of the 13th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2012)*, pages 254–263, 2012.

[Kikot *et al.*, 2012a] Stanislav Kikot, Roman Kontchakov, V. Podolskii, and Michael Zakharyaschev.
Long rewritings, short rewritings.
In *Proc. of the 25th Int. Workshop on Description Logic (DL 2012)*, volume 846 of *CEUR Electronic Workshop Proceedings*, `http://ceur-ws.org/`, 2012.

[Kikot *et al.*, 2012b] Stanislav Kikot, Roman Kontchakov, and Michael Zakharyaschev.
Conjunctive query answering with OWL 2 QL.
In *Proc. of the 13th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2012)*, pages 275–285, 2012.

References VIII

[Kontchakov *et al.*, 2010] Roman Kontchakov, Carsten Lutz, David Toman, Frank Wolter, and Michael Zakharyaschev.
The combined approach to query answering in *DL-Lite*.
In *Proc. of the 12th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2010)*, pages 247–257, 2010.

[Lenzerini and Nobili, 1990] Maurizio Lenzerini and Paolo Nobili.
On the satisfiability of dependency constraints in entity-relationship schemata.
Information Systems, 15(4):453–461, 1990.

[Levy and Rousset, 1998] Alon Y. Levy and Marie-Christine Rousset.
Combining Horn rules and description logics in CARIN.
Artificial Intelligence, 104(1–2):165–209, 1998.

[Lutz, 2008] Carsten Lutz.
The complexity of conjunctive query answering in expressive description logics.
In *Proc. of the 4th Int. Joint Conf. on Automated Reasoning (IJCAR 2008)*, volume 5195 of *Lecture Notes in Artificial Intelligence*, pages 179–193. Springer, 2008.

References IX

[Ortiz *et al.*, 2006] Maria Magdalena Ortiz, Diego C., and Thomas Eiter.
Characterizing data complexity for conjunctive query answering in expressive description logics.
In *Proc. of the 21st Nat. Conf. on Artificial Intelligence (AAAI 2006)*, pages 275–280, 2006.

[Ortiz *et al.*, 2008] Magdalena Ortiz, Diego C., and Thomas Eiter.
Data complexity of query answering in expressive description logics via tableaux.
J. of Automated Reasoning, 41(1):61–98, 2008.

[Pérez-Urbina *et al.*, 2010] Héctor Pérez-Urbina, Boris Motik, and Ian Horrocks.
Tractable query answering and rewriting under description logic constraints.
J. of Applied Logic, 8(2):186–209, 2010.

[Poggi *et al.*, 2008] Antonella Poggi, Domenico Lembo, Diego C., Giuseppe De Giacomo, Maurizio Lenzerini, and Riccardo Rosati.
Linking data to ontologies.
J. on Data Semantics, X:133–173, 2008.

[Queralt *et al.*, 2012] Anna Queralt, Alessandro Artale, Diego C., and Ernest Teniente.
OCL-Lite: Finite reasoning on UML/OCL conceptual schemas.
Data and Knowledge Engineering, 73:1–22, 2012.

References X

[Rodríguez-Muro and C., 2011] Mariano Rodríguez-Muro and Diego C.
Dependencies to optimize ontology based data access.
In *Proc. of the 24th Int. Workshop on Description Logic (DL 2011)*, volume 745 of *CEUR Electronic Workshop Proceedings*, `http://ceur-ws.org/`, 2011.

[Rodriguez-Muro and C., 2012] Mariano Rodriguez-Muro and Diego C.
High performance query answering over *DL-Lite* ontologies.
In *Proc. of the 13th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2012)*, pages 308–318, 2012.

[Rodríguez-Muro, 2010] Mariano Rodríguez-Muro.
Tools and Techniques for Ontology Based Data Access in Lightweight Description Logics.
PhD thesis, KRDB Research Centre for Knowledge and Data, Free University of Bozen-Bolzano, 2010.

[Rosati and Almatelli, 2010] Riccardo Rosati and Alessandro Almatelli.
Improving query answering over *DL-Lite* ontologies.
In *Proc. of the 12th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2010)*, pages 290–300, 2010.

References XI

[Rosati, 2012] Riccardo Rosati.

Query rewriting under extensional constraints in *DL-Lite*.

In *Proc. of the 25th Int. Workshop on Description Logic (DL 2012)*, volume 846 of *CEUR Electronic Workshop Proceedings*, `http://ceur-ws.org/`, 2012.