Managing open data through ontologies

Maurizio Lenzerini

joint work with *Gianluca Cima, Antonella Poggi*

Dipartimento di Ingegneria Informatica
Automatica e Gestionale Antonio Ruberti

SAPIENZA
UNIVERSITÀ DI ROMA

ISIP 2019

13th Int. Workshop on Information Search, Integration, and Personalization

Heraklion, Greece, May 9-10, 2019

Open data

What is Open Data?

"Open data is data that anyone can access, use or share."[1]

Why Open Data?

- **interoperability**
- **transparency in government**
- **improving public services**
- **creating social and commercial value**
- **essential element for the data-driven society**

[1] Open Data Institute (`https://theodi.org/`)

Open data semantics

- Current practices for publishing Open Data focus essentially on providing the dataset (extensional information), often in simple format as CSV.
- Dataset are mostly documented through description in natural language.
- As a consequence, the semantics of datasets is not formally expressed in a machine-readable form.

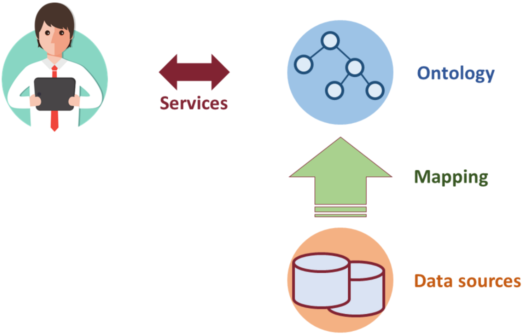
In this talk we explore the use of the paradigm of **Ontology-based Data Management (OBDM)** as a means for publishing open data that are

- high-quality
- semantically annotated

Outline

Outline

Ontology-based data management: architecture



Based on three main components:

- Ontology, a declarative, logic-based specification of the domain of interest, used as a unified, conceptual view for clients
- Data sources, representing external, independent, heterogeneous, storage (or, more generally, computational) structures
- Mappings, used to semantically link data at the sources to the ontology

Formal framework of ontology-based data management

An ontology-based data management (OBDM) **specification** [Poggi et al 2008] is a triple $\mathcal{J} = \langle \mathcal{O}, \mathcal{M}, \mathcal{S} \rangle$, where

- $\mathcal{O}$ is the ontology, expressed as a logical theory (here, is a TBox in a **Description Logic**)
- $\mathcal{S}$ is a (federated) relational database schema representing the data sources
- $\mathcal{M}$ is a set of mapping assertions, each of the form

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

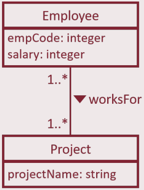
  where
  - $\Phi(\vec{x})$ is a query over $\mathcal{S}$, returning values for $\vec{x}$
  - $\Psi(\vec{x})$ is a query over $\mathcal{O}$, whose free variables are from $\vec{x}$.

An OBDM **system** is a pair $(\mathcal{J}, D)$, where $\mathcal{J} = \langle \mathcal{O}, \mathcal{M}, \mathcal{S} \rangle$ is an OBDM specification, and D is a source database, i.e., an instance for $\mathcal{S}$.

Ontology-based data management specification – Example

Ontology $\mathcal{O}$ (TBox)

Employee $\sqsubseteq$ $\exists$worksFor
Employee $\sqsubseteq$ $\exists$empCode
Employee $\sqsubseteq$ $\exists$salary
Project $\sqsubseteq$ $\exists$worksFor$^-$
Project $\sqsubseteq$ $\exists$projectName
$\exists$worksFor $\sqsubseteq$ Employee
$\exists$worksFor$^-$ $\sqsubseteq$ Project



Federated schema of the DB $\mathcal{S}$

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

D_2[*Code*: String, *Salary*: Int]
Employee's Code with salary

D_3[*Code*: String, *SSN*: String]
Employee's Code with SSN

...

Mapping $\mathcal{M}$

M_1:
```
SELECT SSN,PrName
FROM D1
```
$\leadsto$
Employee(**pers**(*SSN*)),
Project(**proj**(*PrName*)),
projectName(**proj**(*PrName*), *PrName*),
workFor(**pers**(*SSN*), **proj**(*PrName*))

M_2:
```
SELECT SSN,Salary
FROM D2, D3
WHERE D2.Code = D3.Code
```
$\leadsto$
Employee(**pers**(*SSN*)),
salary(**pers**(*SSN*), *Salary*)

Ontology-based data management: Semantics

Let $\mathcal{J} = \langle \mathcal{O}, \mathcal{M}, \mathcal{S} \rangle$ be an OBDM specification, and let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ be an interpretation for the ontology $\mathcal{O}$.

Def.: Semantics

The semantics of $\mathcal{J}$ is given with respect to a legal instance D of $\mathcal{S}$.
$\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ is a model of $(\mathcal{J}, D)$ if:

- $\mathcal{I}$ satisfies all axioms of $\mathcal{O}$, i.e., is a model of $\mathcal{O}$;
- $\mathcal{I}$ satisfies $\mathcal{M}$ wrt D, i.e., satisfies every assertion $\Phi(\vec{x}) \leadsto \Psi(\vec{x})$ in $\mathcal{M}$ wrt D, which means that the sentence $\forall \vec{x}\ (\Phi(\vec{x}) \rightarrow \Psi(\vec{x}))$ is true in $\mathcal{I} \cup D$.

$(\mathcal{J}, D)$ is satisfiable or consistent, if it admits at least one model.

Certain answer

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

Outline

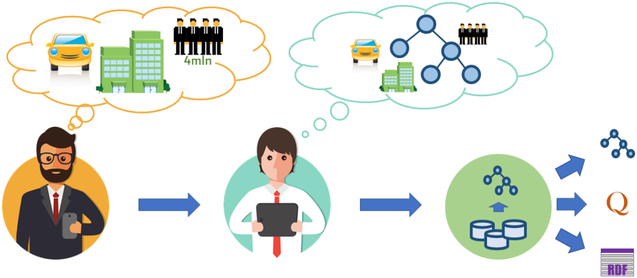
OBDM and open data

Once we have defined an **OBDM** specification over our data sources, we can exploit it for the tasks needed for publishing **open data**, at least in two scenarios:

1. Top-down: we want to publish a dataset and we know hot to specify its content in terms of the ontology (open data publication and annotation)
2. Bottom-up: we have an existing dataset, and we want to produce a semantic description of it (open data annotation)

⇒ If the IT-expert is familiar with OBDM, the dataset to publish is specified by referring to the domain's elements (the ontology).

Example: Mr White (the manager) wants to publish data (number of vehicles per city) regarding vehicles registered this year in all those regions where the number of citizens is greater than 4 mln. Mr. White himself, or Johnny (the OBDM-expert), know how to formulate the request in terms of the ontology.



Scenario 1: The top-down approach (cont'd)

The way to proceed is:

1. express the request of the dataset we want to open in terms of a **query** over the **ontology**;
2. evaluate such query over the OBDM system (compute the certain answers);
3. express the result in RDF by using the query expression and the ontology;
4. publish the **ontology**, the **query**, and the **dataset**.

Scenario 2: The bottom-up approach

⇒ The dataset is already published, and the query over the source used to produce it is known; we now want to automatically associate to it a semantic annotation.

The way to proceed is:

1. given the **query** over the **data source** used to produce the dataset, a mechanism is needed in order translate the source query into the ontology, so as to derive a **query** over the **ontology**;
2. use such query as a semantic annotation of the dataset

What do we need in the two scenarios

Once we have the OBDM specification $\mathcal{J} = \langle \mathcal{O}, \mathcal{M}, \mathcal{S} \rangle$, we need

1. In scenario 1:
   a method that, given a query $q_{\mathcal{O}}$ over the ontology $\mathcal{O}$, computes a query $q_{\mathcal{S}}$ such that, for every database D for $\mathcal{S}$, $q_{\mathcal{S}}^{D}$ coincides with the certain answers $cert(q_{\mathcal{O}}, \mathcal{J}, D)$

   $\longrightarrow$ Ontology-to-Source query rewriting

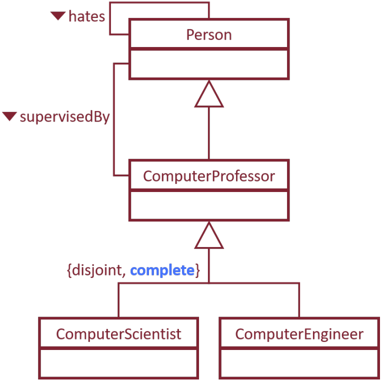
2. In scenario 2:
   a method that, given a query $q_{\mathcal{S}}$ over the source schema $\mathcal{S}$, computes a query $q_{\mathcal{O}}$ such that, for every for every database D for $\mathcal{S}$, $cert(q_{\mathcal{O}}, \mathcal{J}, D)$ coincides with $q_{\mathcal{S}}^{D}$

   $\longrightarrow$ Source-to-Ontology query rewriting

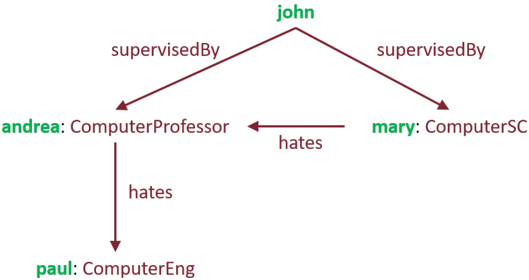
Outline

Ontology-to-Source rewriting

Depending of the expressive power of the query, ontology and mapping language, the problem may becom very involved.



Note that ComputerProfessor is partitioned into ComputerScientist and ComputerEngineer.

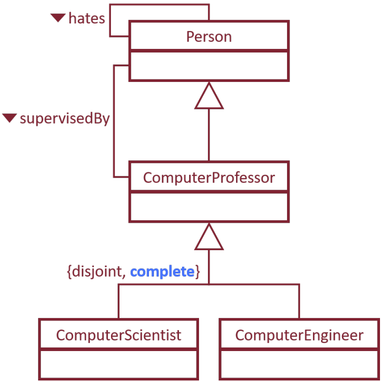


Query:

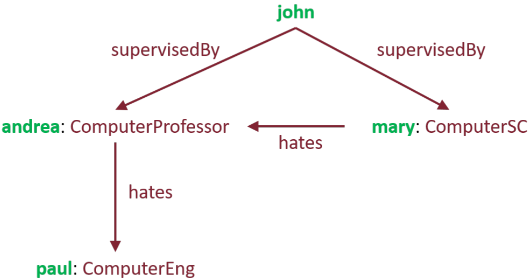
$q(x) \leftarrow \exists y, z.\, \mathsf{supervisedBy}(x, y), \mathsf{ComputerSC}(y), \mathsf{hates}(y, z), \mathsf{ComputerEng}(z)$

Answer: ???

Depending of the expressive power of the query, ontology and mapping language, the problem may becom very involved.



Note that ComputerProfessor is partitioned into ComputerScientist and ComputerEngineer.



Query:

$q(x) \leftarrow \exists y, z.\, \text{supervisedBy}(x, y), \text{ComputerSC}(y), \text{hates}(y, z), \text{ComputerEng}(z)$

Answer: { john }   To obtain this answer, we need to reasoning by cases

$\longrightarrow$ **No First-order query is an Ontology-to-Source rewriting of q**

Complexity of conjunctive query answering in DLs

Studied extensively for (unions of) CQs and various ontology languages:

	Combined complexity	Data complexity
Plain databases	NP-complete	AC^0 (1)
OWL 2	???	coNP-hard (2)

(1) This makes it possible to scale with the data.

(2) Already for a TBox with a single disjunction (see example above).

Question

Can we find interesting DLs for which we can always rewrite the query over the ontology into a FOL query over the source? A lot of research work/answers in the last decade!

Complexity of conjunctive query answering in DLs

Studied extensively for (unions of) CQs and various ontology languages:

	Combined complexity	Data complexity
Plain databases	NP-complete	AC^0 (1)
OWL 2	???	coNP-hard (2)

(1) This makes it possible to scale with the data.

(2) Already for a TBox with a single disjunction (see example above).

Question

Can we find interesting DLs for which we can always rewrite the query over the ontology into a FOL query over the source? A lot of research work/answers in the last decade!

The *DL-Lite* family

DL-Lite is a family [Calvanese et al 2007] 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.

Nevertheless they have the right expressive power to capture the essential features of conceptual modeling formalisms.

DL-Lite provides robust foundations for **Ontology-Based Data Management**.

The *DL-Lite* family is at the basis of the OWL 2 QL profile of the W3C standard Web Ontology Language OWL.

Capturing basic ontology constructs in *DL-Lite*

Modeling construct	*DL-Lite*
ISA between classes	$Student \sqsubseteq Person$
... and or relations	$isMatherOf \sqsubseteq isParentOf$
Disjointness between classes	$Student \sqsubseteq \neg Professor$
... and or relations	$isMatherOf \sqsubseteq \neg isFatherOf$
Domain of properties	$\exists livesIn \sqsubseteq Person$
Range of properties	$\exists livesIn^{-} \sqsubseteq City$
Mandatory participation *(min card = 1)*	$Person \sqsubseteq \exists livesIn$ $City \sqsubseteq \exists livesIn-$
Functionality of relations *(max card = 1)*	(**funct** $livesIn$) (**funct** $isFatherOf^{-}$)

Note: *DL-Lite* distinguishes between abstract objects and data values as well (**we can represent concept attributes**) (ignored here).

Query answering in *DL-Lite*-based OBDM systems

In a *DL-Lite*-based OBDM specification $\langle \mathcal{O}, \mathcal{M}, \mathcal{S} \rangle$

- $\mathcal{O}$ is expressed in *DL-Lite*
- $\mathcal{M}$ is a set of GAV mapping assertions (the right-hand side Ψ is a conjunctive query (CQ) without existential variables).
- queries over $\mathcal{O}$ are unions of conjunctive queries (UCQs).

Query answering is performed through ontology-to-source rewriting:

Given a (U)CQ q, OBDM specification $\mathcal{J} = \langle \mathcal{O}, \mathcal{M}, \mathcal{S} \rangle$, database D, we compute $cert(q, \mathcal{J}, D)$ as follows:

1. we compute the **ontology rewriting** $r_{q,\mathcal{O}}$ of q using $\mathcal{O}$;
2. we compute the **mapping rewriting** r of $r_{q,\mathcal{O}}$ using $\mathcal{M}$;
3. evaluate the UCQ r directly over D.

Correctness of this procedure shows **FOL-rewritability** of query answering in *DL-Lite*

Computational complexity of query answering

Proposition

Query answering on a *DL-Lite* ontology-based data management system $(\langle \mathcal{O}, \mathcal{M}, \mathcal{S} \rangle, D)$ of the kind considered so far is

1. PTIME in the size of the ontology $\mathcal{O}$ and the mappings $\mathcal{M}$.
2. AC^0 in the size of the database D, in fact FOL-rewritable.
3. Exponential in the size of the query, more precisely NP-complete.

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

Can we go beyond *DL-Lite* and remain in AC^0?

The DLs of the *DL-Lite* family are essentially the maximally expressive DLs enjoying these nice computational properties.

Outline

From source queries to ontology queries

Problem

Given an OBDM specification $\mathcal{J} = \langle \mathcal{O}, \mathcal{M}, \mathcal{S} \rangle$ and a query $q_{\mathcal{S}}$ over the sources schema $\mathcal{S}$, which query $q_{\mathcal{O}}$ over the ontology $\mathcal{O}$ best characterizes $q_{\mathcal{S}}$ under $\mathcal{J}$?

Note that this problem is relevant also for other tasks related to the management of the information system of an organization, e.g.,

- the task of providing the semantics of the various data sources;
- the task of providing the semantics of data services expressed over the sources (reverse engineering);
- checking whether the ontology and the mapping assertions are "adequate" for answering source queries through the OBDM system.

Perfect Source-to-Ontology (S-to-O) rewriting

Let $\mathcal{J} = \langle \mathcal{O}, \mathcal{M}, \mathcal{S} \rangle$ be an OBDM specification, and let $q_{\mathcal{S}}$ be a query over $\mathcal{S}$.

Definition (Perfect S-to-O rewriting)

A query $q_{\mathcal{O}}$ over $\mathcal{O}$ is a **perfect** S-to-O $\mathcal{J}$-rewriting of $q_{\mathcal{S}}$, if for every database D legal for $\mathcal{S}$ such that $\langle \mathcal{J}, D \rangle$ is consistent, we have that

$$q_{\mathcal{S}}^{D} = cert(q_{\mathcal{O}}, \mathcal{J}, D)$$

$q_{\mathcal{S}}^{D} = cert(q_{\mathcal{O}}, \mathcal{J}, D)$	
Ontology-to-Source rewriting	**Source-to-Ontology rewriting**
input: $q_{\mathcal{O}}$	**input**: $q_{\mathcal{S}}$
output: $q_{\mathcal{S}}$	**output**: $q_{\mathcal{O}}$

Source-to-Ontology rewriting - Example

Example

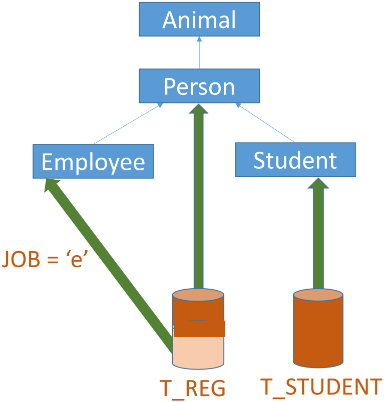
Consider the OBDM specification $\mathcal{J} = \langle \mathcal{O}, \mathcal{M}, \mathcal{S} \rangle$, where

- $\mathcal{O} = \{ \ Student \sqsubseteq Person,$
  $Employee \sqsubseteq Person,$
  $Person \sqsubseteq Animal \ \}$
  $Animal \sqsubseteq \neg University \ \}$
- $\mathcal{S}$ contains two tables
  - T_REG(ID,JOB)
  - T_STUDENT(ID,UNIVERSITY)
- $\mathcal{M}$ is as follows:

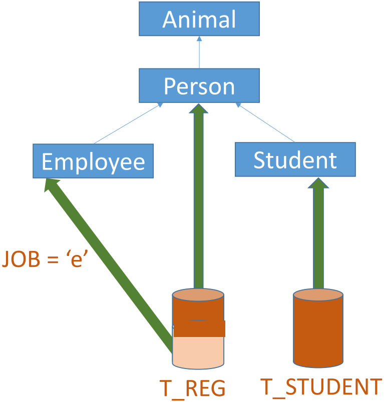
m_1:	select ID as x from T_REG	$\rightsquigarrow$	$Person(x)$
m_2:	select ID as x from T_REG where JOB = 'e'	$\rightsquigarrow$	$Employee(x)$
m_3:	select ID as x from T_STUDENT	$\rightsquigarrow$	$Student(x)$

Source-to-Ontology rewriting - Example

Source query $Q_{\mathcal{S}}$:
select ID as x from T_REG



Source-to-Ontology rewriting - Example

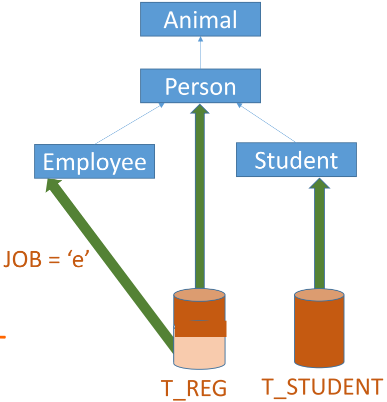


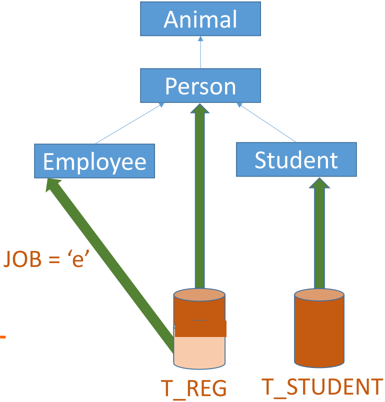
Source query $Q_{\mathcal{S}}$:
select ID as x from T_REG

- perfect S-to-O $\mathcal{J}$-rewriting of $Q_{\mathcal{S}}$: none

Source-to-Ontology rewriting - Example

Source query $Q_{\mathcal{S}}$:
select ID as x from T_REG
Source query $Q'_{\mathcal{S}}$:
select ID as x from T_STUDENT





Source query $Q_{\mathcal{S}}$:
select ID from T_REG
Source query $Q'_{\mathcal{S}}$:
select ID as x from T_STUDENT

- perfect S-to-O $\mathcal{J}$-rewriting of $Q_{\mathcal{S}}$: none
- perfect S-to-O $\mathcal{J}$-rewriting of $Q'_{\mathcal{S}}$: $Student(x)$

Sound and complete S-to-O rewritings

Let $\mathcal{J} = \langle \mathcal{O}, \mathcal{M}, \mathcal{S} \rangle$ be an OBDM specification, and let $q_{\mathcal{S}}$ be a query over $\mathcal{S}$.

Definition (Sound S-to-O rewriting)

A query $q_{\mathcal{O}}$ over $\mathcal{O}$ is a **sound** S-to-O $\mathcal{J}$-rewriting of $q_{\mathcal{S}}$ if for every database D legal for $\mathcal{S}$ such that $\langle \mathcal{J}, D \rangle$ is consistent, we have that

$$cert(q_{\mathcal{O}}, \mathcal{J}, D) \subseteq q_{\mathcal{S}}^{D}$$

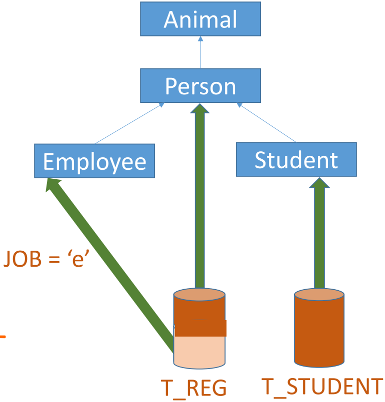
Definition (Complete S-to-O rewriting)

A query $q_{\mathcal{O}}$ over $\mathcal{O}$ is a **complete** S-to-O $\mathcal{J}$-rewriting of $q_{\mathcal{S}}$ if for every database D legal for $\mathcal{S}$ such that $\langle \mathcal{J}, D \rangle$ is consistent, we have that

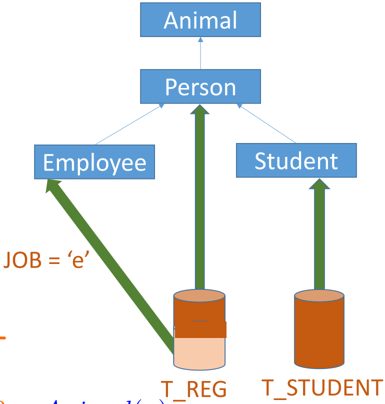
$$q_{\mathcal{S}}^{D} \subseteq cert(q_{\mathcal{O}}, \mathcal{J}, D)$$

Source-to-Ontology rewritings - Example

Source query $Q_{\mathcal{S}}$:
select ID as x from T_REG
Source query $Q'_{\mathcal{S}}$:
select ID as x from T_STUDENT



Source-to-Ontology rewritings - Example



Source query $Q_{\mathcal{S}}$:
select ID as x from T_REG
Source query $Q'_{\mathcal{S}}$:
select ID as x from T_STUDENT

- complete S-to-O $\mathcal{J}$-rewriting of $Q_{\mathcal{S}}$: $Animal(x)$
- complete S-to-O $\mathcal{J}$-rewriting of $Q_{\mathcal{S}}$: $Person(x)$
- sound S-to-O $\mathcal{J}$-rewriting of $Q_{\mathcal{S}}$: $Employee(x)$
- sound S-to-O $\mathcal{J}$-rewriting of $Q_{\mathcal{S}}$: $Employee(x), University(x)$
- perfect S-to-O $\mathcal{J}$-rewriting of $Q_{\mathcal{S}}$: none
- perfect S-to-O $\mathcal{J}$-rewriting of $Q'_{\mathcal{S}}$: $Student(x)$

Minimal and maximal S-to-O rewritings

Let $\mathcal{L}$ be a class of queries.

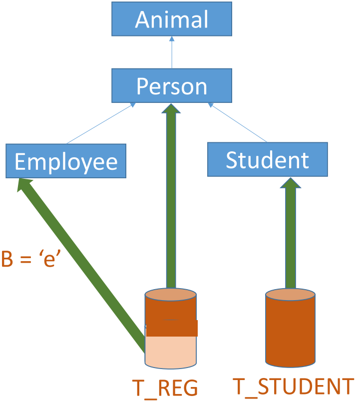
Definition

If $q_{\mathcal{O}} \in \mathcal{L}$ is a sound $\mathcal{S}$-to-$\mathcal{O}$ $\mathcal{J}$-rewriting of $q_{\mathcal{S}}$, then $q_{\mathcal{O}}$ is *$\mathcal{L}$-maximally sound* if no $q' \in \mathcal{L}$ exists such that (i) q' is a sound $\mathcal{S}$-to-$\mathcal{O}$ $\mathcal{J}$-rewriting of $q_{\mathcal{S}}$, (ii) $cert_{q_{\mathcal{O}},\mathcal{J}} \sqsubseteq cert_{q',\mathcal{J}}$, and (iii) there exists an $\mathcal{S}$-database s.t. $cert^{D}_{q_{\mathcal{O}},\mathcal{J}} \subset cert^{D}_{q',\mathcal{J}}$.

Definition

If $q_{\mathcal{O}} \in \mathcal{L}$ is a complete $\mathcal{S}$-to-$\mathcal{O}$ $\mathcal{J}$-rewriting of $q_{\mathcal{S}}$, then $q_{\mathcal{O}}$ is *$\mathcal{L}$-minimally complete* if no $q' \in \mathcal{L}$ exists such that (i) q' is a complete $\mathcal{S}$-to-$\mathcal{O}$ $\mathcal{J}$-rewriting of $q_{\mathcal{S}}$, (ii) $cert_{q',\mathcal{J}} \sqsubseteq cert_{q_{\mathcal{O}},\mathcal{J}}$, and (iii) there exists an $\mathcal{S}$-database s.t. $cert^{D}_{q',\mathcal{J}} \subset cert^{D}_{q_{\mathcal{O}},\mathcal{J}}$.

Source-to-Ontology rewriting - Example



Source query $Q_{\mathcal{S}}$:
select ID as x from T_REG
Source query $Q'_{\mathcal{S}}$:
select ID as x from T_STUDENT

- UCQ-minimally complete S-to-O $\mathcal{J}$-rewriting of $Q_{\mathcal{S}}$: $Person(x)$
- UCQ-maximally sound S-to-O $\mathcal{J}$-rewriting of $Q_{\mathcal{S}}$: $Employee(x)$
- perfect S-to-O $\mathcal{J}$-rewriting of $Q_{\mathcal{S}}$: none
- perfect S-to-O $\mathcal{J}$-rewriting of $Q'_{\mathcal{S}}$: $Student(x)$

Can we compute the UCQ-maximally complete S-to-O rewriting?

We now focus on the problem of computing the **UCQ-maximally complete S-to-O rewriting** of a $q_{\mathcal{S}}$ over the source schema $\mathcal{S}$ with respect to an OBDM specification $\mathcal{J} = \langle \mathcal{O}, \mathcal{M}, \mathcal{S} \rangle$, where

- $q_{\mathcal{S}}$ is a **CQ**
- $\mathcal{O}$ a **DL-Lite** ontology, and
- $\mathcal{M}$ a set of **GLAV** mapping assertions of the form

$$\forall \vec{x}\, \forall \vec{y}\, (\Phi_{\mathcal{S}}(\vec{x}, \vec{y}) \;\rightarrow\; \exists \vec{z}\, \Psi_{\mathcal{O}}(\vec{x}, \vec{z}))$$

where $\Phi_{\mathcal{S}}(\vec{x})$ is a CQ over $\mathcal{S}$ and $\Psi_{\mathcal{O}}$ is a CQ over the ontology $\mathcal{O}$.

Note that computing such rewriting solves also the problem of computing the perfect rewriting (it is sufficient to check the result for soundness)

How to compute it: basic idea

There is a very strict correlation between CQs and database. Given a CQ q over a schema $\mathcal{S}$ it is possible to construct in linear time a database D_q of $\mathcal{S}$ that fully captures q, and **viceversa**:

- every constant in q becomes a value in D_q;
- every variable in q becomes a *labeled null* in D_q;
- every atom $R_i(\vec{u}) \in q$ becomes a fact (tuple) in D_q.

Example

Let $\mathcal{S}$ contain the tables **Tab1(id,city)** and **Tab2(id,city)**, and consider the following CQ $q_\mathcal{S}$ over $\mathcal{S}$

$$q_\mathcal{S}(x) \leftarrow \text{Tab1}(x, y), \text{Tab2}(\text{'sara'}, y)$$

The $\mathcal{S}$-database $D_{q_\mathcal{S}}$ associated to the query $q_\mathcal{S}$ is:

Tab1(**x**,**y**), Tab2(**'sara'**,**y**)

where **x**, **y** are labeled nulls.

How to compute it: basic idea (cont'd)

Roughly speaking, the database $D_{q_\mathcal{S}}$ of $\mathcal{S}$ associated to $q_\mathcal{S}$ is representative of those instances of $\mathcal{S}$ on which the evaluation of $q_\mathcal{S}$ is non-empty.

Given the OBDM specification $\mathcal{J} = \langle \mathcal{O}, \mathcal{M}, \mathcal{S} \rangle$, we denote by $\mathcal{M}(D_{q_\mathcal{S}})$ the set of $\mathcal{O}$-facts (ABox, in DL terminology) obtained by chasing $D_{q_\mathcal{S}}$ with the mapping assertions in $\mathcal{M}$.

Intuition

The query $q_\mathcal{O}$ over the ontology corresponding to the UCQ-minimally complete $\mathcal{J}$-rewriting of $q_\mathcal{S}$ is the query associated to $\mathcal{M}(D_{q_\mathcal{S}})$.

$$Q_\mathcal{S} \xrightarrow{\text{compute the associated DB}} D_{Q_\mathcal{S}} \xrightarrow{\text{compute the chase}} \mathcal{M}(D_{Q_\mathcal{S}}) \xrightarrow{\text{compute the associated query}} Q_\mathcal{O}$$

How to compute it: basic idea (cont'd)

Example

Suppose to have the following mapping:

m_1: Tab1(x,y) $\rightsquigarrow$ $Person(x), livesIn(x, y)$

m_2: Tab2(x,y) $\rightsquigarrow$ $Person(x), worksIn(x, y)$

and consider the instance $D_{Q_\mathcal{S}}$ of $\mathcal{S}$ associated to the query $Q_\mathcal{S}$:

Tab1(**x**,**y**), Tab2(**'sara'**,**y**),

It is easy to see that $\mathcal{M}(D_{Q_\mathcal{S}})$ is the **ABox** containing the following assertions:

$$Person(\mathbf{x}),\ livesIn(\mathbf{x}, \mathbf{y}),\ Person(\text{'sara'}),\ worksIn(\text{'sara'}, \mathbf{y})$$

From $\mathcal{M}(D_{Q_\mathcal{S}})$ we can construct the following query over the ontology, denoting *all the Persons living in the same city where the person 'sara' works*

$$Q_\mathcal{O}(x) \leftarrow Person(x),\ livesIn(x, y),\ Person(\text{'sara'}),\ worksIn(\text{'sara'}, y)$$

UCQ-minimally complete S-to-O rewriting: algorithm

Algorithm: FindMinimallyCompleteRewriting
Input: a *DL-Lite* OBDM specification $\langle \mathcal{O}, \mathcal{M}, \mathcal{S} \rangle$, a CQ $q_{\mathcal{S}}$ over $\mathcal{S}$
Output: a CQ $q_{\mathcal{O}}$ over $\mathcal{O}$

1. Compute D_{q_s} from q_s (i.e., the database, possibly with incomplete information, associated to the query q_s).
2. Chase D_{q_s} wrt $\mathcal{M}$ to produce an ABox A with variables.
3. If $A = \emptyset$ (i.e., no atoms in A), then return the query $\{tup(q_s) \mid \top(tup(q_s))\}$.
4. Chase A wrt $\neg Q^{1\neq}_{sat}$; if the chase fails, then return the query $\{tup(Q_s) \mid \bot(tup(Q_s))\}$; otherwise, let A' be the instance produced, and let ψ be the set of equality applied to the variables in D_{q_s} by the chase.
5. Evaluate $q^{0\neq}_{sat}$ over A'; if the answer is $\{()\}$ (i.e., $J \models q^{0\neq}_{sat}$), then return the query $\{tup(q_s) \mid \bot(tup(q_s))\}$; otherwise, let $q_{A'}$ be the boolean conjunctive query associated to A'.
6. Return $\{\psi(tup(q_s)) \mid Q_{A'}(\psi(tup(Q_s)) \wedge \top(\overline{w})\}$, where $\overline{w}$ is the tuple composed by all terms in $\psi(tup(q_s))$ not appearing in A'.

UCQ-minimally complete S-to-O rewriting: properties

Let $\mathcal{J} = \langle \mathcal{O}, \mathcal{M}, \mathcal{S} \rangle$ be a *DL-Lite* OBDM specification, and let $q_{\mathcal{S}}$ be a CQ over $\mathcal{S}$.

- **FindMinimallyCompleteRewriting**$(\mathcal{J}, q_{\mathcal{S}})$ terminates and runs in:
  - (i) PTIME in the size of $q_{\mathcal{S}}$;
  - (ii) PTIME in the size of $\mathcal{O}$;
  - (iii) EXPTIME in the size of $\mathcal{M}$.
- The query returned by **FindMinimallyCompleteRewriting**$(\mathcal{J}, q_{\mathcal{S}})$ is a **UCQ-minimally complete S-to-O $\mathcal{J}$-rewriting** of $q_{\mathcal{S}}$.
- The UCQ-minimally complete S-to-O $\mathcal{J}$-rewriting of $q_{\mathcal{S}}$ is **unique** up to logical equivalence, and can be expressed as a conjunctive query.
- A perfect S-to-O $\mathcal{J}$-rewriting of $q_{\mathcal{S}}$ exists if and only if the query $q_{\mathcal{O}}$ returned by **FindMinimallyCompleteRewriting**$(\mathcal{J}, q_{\mathcal{S}})$ is a sound S-to-O $\mathcal{J}$-rewriting of $q_{\mathcal{S}}$, in which case $q_{\mathcal{O}}$ is the perfect S-to-O $\mathcal{J}$-rewriting of $q_{\mathcal{S}}$.

Outline

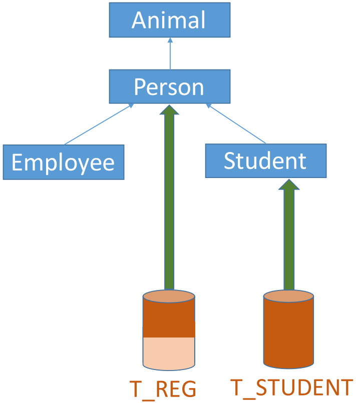
Future work

We argue that OBDM provides a suitable formal basis for a new, well-founded methodology for open data publishing, and that in this context, S-to-O rewriting is an interesting notion.

Many challenges remain, for example:

- Studying UCQ-maximally sound S-to-O rewritings.
- If we relax the assumption that $\mathcal{L} = \text{UCQ}$, can we have/compute more accurate S-to-O rewritings?
- We have assumed that the query language used to express $q_{\mathcal{S}}$ is the language of CQs. This is too limited: Open data often requires aggregation, negation, universal quantification, etc. Can we compute the S-to-O rewriting of queries with such operators?

Maximally sound s-to-t rewritings - Example



Source query $q_{\mathcal{S}}$:
select ID as x from T_REG

- UCQ-maximally sound S-to-O $\mathcal{J}$-rewriting of $q_{\mathcal{S}}$: empty query
- "best" sound s-to-t rewriting of $q_{\mathcal{S}}$?

Thank you for your attention!