Methods and Tools for Developing Ontology-Based Data Management Solutions

Concepts for Ontology-Based Data Management

Domenico Lembo, Valerio Santarelli, **Domenico Fabio Savo**

SEMANTICS 2018
Vienna, Austria, September 11, 2018

Finding data...

Fragment of a relational table in a Bank Information system:

CUC	TS_START	TS_END	ID_GRUP	FLAG_CP	FLAG_CF	FATTURATO	FLAG_FATT
124589	30-lug-2004	1-gen-9999	92736	S	N	195000,00	N
140904	15-mag-2001	15-giu-2005	35060	N	N	230600,00	N
124589	5-mag-2001	30-lug-2004	92736	N	S	195000,00	S
-452901	13-mag-2001	27-lug-2004	92770	S	N	392000,00	N
129008	10-mag-2001	1-gen-9999	62010	N	S	247000,00	S
-472900	10-mag-2001	1-gen-9999	62010	S	N	0 00	N
130976	7-mag-2001	9-lug-2003	75680				

Finding data...

Negative value denotes a holding

CUC	TS_START	TS_END	ID_GRUP	FLAG_CP	FLAG_CF	FATTURATO	FLAG_FATT
124589	30-lug-2004	1-gen-9999	92736	S	N	195000,00	N
140904	15-mag-2001	15-giu-2005	35060	N	N	230600,00	N
124589	5-mag-2001	30-lug-2004	92736	N	S	195000,00	S
-452901	13-mag-2001	27-lug-2004	92770	S	N	392000,00	N
129008	10-mag-2001	1-gen-9999	62010	N	S	247000,00	S
-472900	10-mag-2001	1-gen-9999	62010	S	N	0 00	N
130976	7-mag-2001	9-lug-2003	75680				

Finding data...

S means that the customer is the leader of the group it belongs to

S means that the customer is the head of the group it belongs to

CUC	TS_START	TS_END	ID_GRUP	FLAG_CP	FLAG_CF	FATTURATO	FLAG_FATT
124589	30-lug-2004	1-gen-9999	92736	S	N	195000,00	N
140904	15-mag-2001	15-giu-2005	35060	N	N	230600,00	N
124589	5-mag-2001	30-lug-2004	92736	N	S	195000,00	S
-452901	13-mag-2001	27-lug-2004	92770	S	N	392000,00	N
129008	10-mag-2001	1-gen-9999	62010	N	S	247000,00	S
-472900	10-mag-2001	1-gen-9999	62010	S	N	0 00	N
130976	7-mag-2001	9-lug-2003	75680				

Finding data...

N means that the FATTURATO field is not valid

CUC	TS_START	TS_END	ID_GRUP	FLAG_CP	FLAG_CF	FATTURATO	FLAG_FATT	
124589	30-lug-2004	1-gen-9999	92736	S	N	195000,00	N	
140904	15-mag-2001	15-giu-2005	35060	N	N	230600,00	N	
124589	5-mag-2001	30-lug-2004	92736	N	S	195000,00	S	
-452901	13-mag-2001	27-lug-2004	92770	S	N	392000,00	N	
129008	10-mag-2001	1-gen-9999	62010	N	S	247000,00	S	
-472900	10-mag-2001	1-gen-9999	62010	S	N	0 00	N	
130976	7-mag-2001	9-lug-2003	75680					

Finding data...the problem

- The example shows that the meaning of data in a table is not always clear.
- Understanding such meaning is crucial if one wants to correctly manage the data in the table and extract information out of it.

- To make matters worse...typically, information systems in the real world use different heterogeneous data sources, both internal and external to the organization.

Organizations spend a significant amount time and money trying to govern the resources (data, meta-data, services, processes, etc.) in their information systems.

Finding data...the problem

- The example shows that the meaning of data in a table is not always clear.
- Understanding such meaning is crucial if one wants to correctly manage the data in the table and extract information out of it.

- To make matters worse...typically, information systems in the real world use different heterogeneous data sources, both internal and external to the organization.

Organizations spend a significant amount time and money trying to govern the resources (data, meta-data, services, processes, etc.) in their information systems.

Managing data through the lens of an ontology

Manage data by adopting principles and techniques studied in Knowledge Representation.

- Provide a conceptual, high level representation of the domain of interest in terms of an **ontology**.
- **Data** may reside where they are (no need to move data).
- **Map** the ontology to the data sources.
- Specify all information requests to the data in terms of the ontology.
- Use inference services to **automatically translate the requests** into queries to the data sources.

Ontology-based Data Management (OBDM)

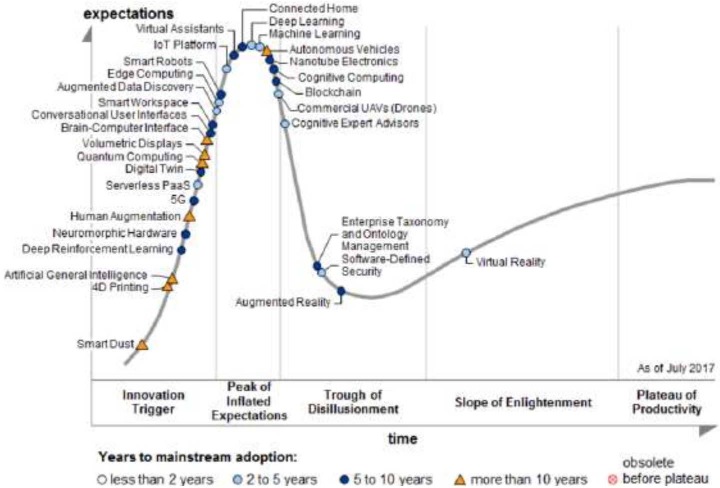
Managing data through the lens of an ontology

Manage data by adopting principles and techniques studied in Knowledge Representation.

- Provide a conceptual, high level representation of the domain of interest in terms of an **ontology**.
- **Data** may reside where they are (no need to move data).
- **Map** the ontology to the data sources.
- Specify all information requests to the data in terms of the ontology.
- Use inference services to **automatically translate the requests** into queries to the data sources.

Ontology-based Data Management (OBDM)

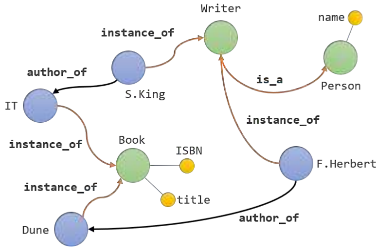
According to Gartner, AI technologies will be the most disruptive class of technologies in digital business in the next decade, and Enterprise Ontology Management will be a mainstream technology in 5 to 10 years.[1]



[1] http://www.gartner.com/newsroom/id/3784363

What is an ontology?

Ontologies are logical specifications describing the things of a domain of knowledge and the relationships between them.



They are typically specified in languages that allow abstraction away from data structures and implementation strategies.

Commonly used languages are:

- Description Logics
- OWL 2 (W3C ontology standard language)

Description Logic Ontologies

Description Logics (DLs) are a family of logic languages used for representing the knowledge of a domain.

The domain of interest is described in terms of **objects** (individuals), **concepts** (sets of objects), and **roles** (binary relations between objects).

A DL ontology is a pair $\langle \mathcal{T}, \mathcal{A} \rangle$, where $\mathcal{T}$ is the Terminological Box (TBox), containing assertions that describe the general aspects of the domain, while $\mathcal{A}$ is the Assertional Box (ABox) providing extensional information on the domain (i.e., data).

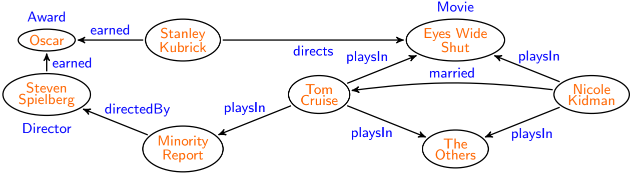
TBox $\mathcal{T}$:

$\exists \mathsf{playsIn} \sqsubseteq \mathsf{Actor}$ $\quad \exists \mathsf{directs} \sqsubseteq \mathsf{Director}$ $\quad \cdots$

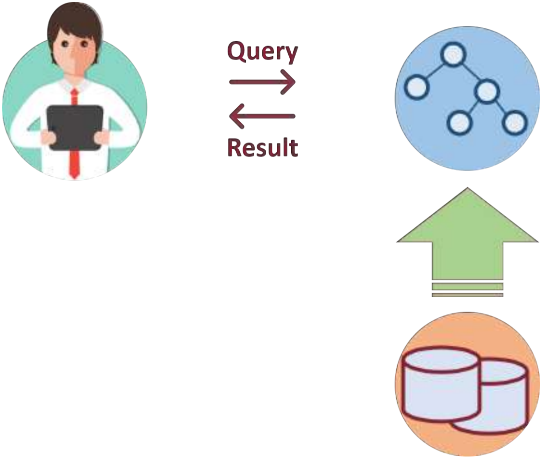
$\exists \mathsf{playsIn}^- \sqsubseteq \mathsf{Movie}$ $\quad \mathsf{Movie} \sqsubseteq \exists \mathsf{directedBy}$

$\mathsf{married} \equiv \mathsf{married}^-$ $\quad \mathsf{directs} \equiv \mathsf{directedBy}^-$

ABox $\mathcal{A}$:



Ontology-based data management: architecture



Ontology provides a global vocabulary and is used as a conceptual view

Mappings semantically link sources and ontology

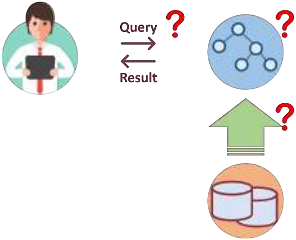
Data Sources external and heterogeneous

Designing a proper OBDM framework

Which is the “right” query language?

Which is the “right” ontology language?

Which is the “right” mapping language?



Trade-off expressive power vs. efficiency of query answering!

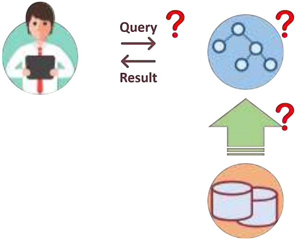
We access big data, so **efficiency w.r.t. the data is crucial**.

Designing a proper OBDM framework

Which is the “right” query language?

Which is the “right” ontology language?

Which is the “right” mapping language?

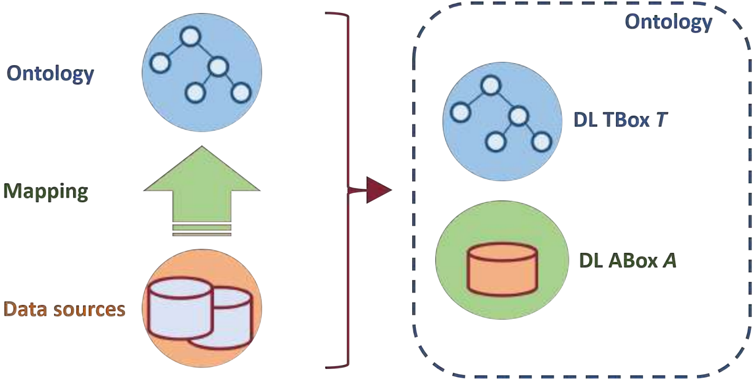


Trade-off expressive power vs. efficiency of query answering!

We access big data, so **efficiency w.r.t. the data is crucial**.

Abstracting from the mapping

Here, we abstract away from the mappings, and consider **ontology-based query answering** (as opposed to data access).

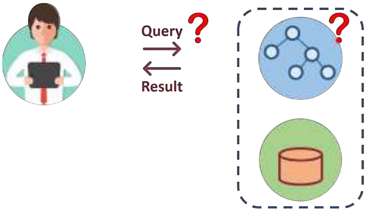


One can see the **ABox** as a database whose predicates are the same of the ontology and containing both objects and values.

Ontology-based query answering

Which is the "right" query language?

Which is the "right" ontology language?



We are in a setting of incomplete information.

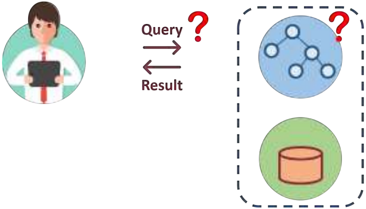
Certain answers

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

Ontology-based query answering

Which is the "right" query language?

Which is the "right" ontology language?



We are in a setting of incomplete information.

Certain answers

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

Which is the right query language?

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

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

Conjunctive queries

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

Ontology-based query answering – Example

TBox $\mathcal{T}$:

$$\exists \text{playsIn} \sqsubseteq \text{Actor} \qquad \exists \text{directs} \sqsubseteq \text{Director} \qquad \cdots$$
$$\exists \text{playsIn}^{-} \sqsubseteq \text{Movie} \qquad \text{Movie} \sqsubseteq \exists \text{directedBy}$$
$$\text{married} \equiv \text{married}^{-} \qquad \text{directs} \equiv \text{directedBy}^{-}$$

ABox $\mathcal{A}$:

Leveraging the TBox axioms, we infer additional ABox facts.

Query: Return all pairs of married actors who played in movies whose directors have earned the same award.

$$\mathbf{q}(x_1, x_2) \leftarrow \exists m_1, m_2, d_1, d_2, a.\ \text{Actor}(x_1) \wedge \text{playsIn}(x_1, m_1) \wedge \text{Movie}(m_1) \wedge \text{directedBy}(m_1, d_1) \wedge \text{Director}(d_1) \wedge \text{earned}(d_1, a) \wedge \text{Actor}(x_2) \wedge \text{playsIn}(x_2, m_2) \wedge \text{Movie}(m_2) \wedge \text{directedBy}(m_2, d_2) \wedge \text{Director}(d_2) \wedge \text{earned}(d_2, a) \wedge \text{married}(x_1, x_2) \wedge \text{Award}(a)$$

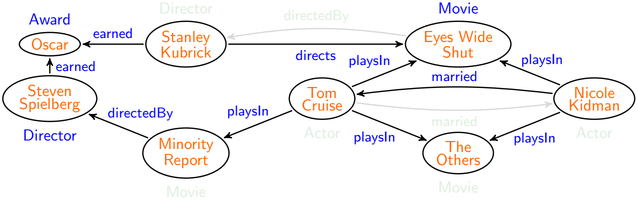
We have that the pair ('Tom Cruise', 'Nicole Kidman') is in $cert(\mathbf{q}, \langle \mathcal{T}, \mathcal{A} \rangle)$.

Ontology-based query answering – Example

TBox $\mathcal{T}$:

$\exists$playsIn $\sqsubseteq$ Actor $\exists$directs $\sqsubseteq$ Director ...
$\exists$playsIn$^-$ $\sqsubseteq$ Movie Movie $\sqsubseteq$ $\exists$directedBy
married $\equiv$ married$^-$ directs $\equiv$ directedBy$^-$

ABox $\mathcal{A}$:



Leveraging the TBox axioms, we infer additional ABox facts.

Query: Return all pairs of married actors who played in movies whose directors have earned the same award.

$\mathbf{q}(x_1, x_2) \leftarrow \exists m_1, m_2, d_1, d_2, a.$ Actor$(x_1) \land$ playsIn$(x_1, m_1) \land$ Movie$(m_1) \land$
directedBy$(m_1, d_1) \land$ Director$(d_1) \land$ earned$(d_1, a) \land$
Actor$(x_2) \land$ playsIn$(x_2, m_2) \land$ Movie$(m_2) \land$
directedBy$(m_2, d_2) \land$ Director$(d_2) \land$ earned$(d_2, a) \land$
married$(x_1, x_2) \land$ Award(a)

We have that the pair ('Tom Cruise', 'Nicole Kidman') is in $cert(\mathbf{q}, \langle \mathcal{T}, \mathcal{A} \rangle)$.

Ontology-based query answering – Example

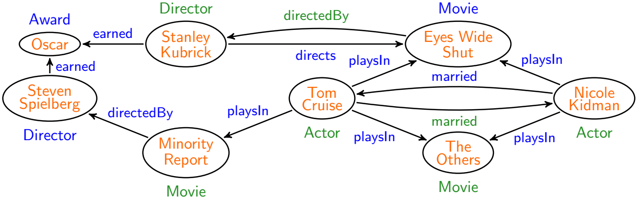
TBox $\mathcal{T}$:

$\exists \text{playsIn} \sqsubseteq \text{Actor}$ $\quad$ $\exists \text{directs} \sqsubseteq \text{Director}$ $\quad \cdots$

$\exists \text{playsIn}^- \sqsubseteq \text{Movie}$ $\quad$ $\text{Movie} \sqsubseteq \exists \text{directedBy}$

$\text{married} \equiv \text{married}^-$ $\quad$ $\text{directs} \equiv \text{directedBy}^-$

ABox $\mathcal{A}$:



Leveraging the TBox axioms, we infer additional ABox facts.

Query: Return all pairs of married actors who played in movies whose directors have earned the same award.

$$\mathbf{q}(x_1, x_2) \leftarrow \exists m_1, m_2, d_1, d_2, a.\ \text{Actor}(x_1) \wedge \text{playsIn}(x_1, m_1) \wedge \text{Movie}(m_1) \wedge \text{directedBy}(m_1, d_1) \wedge \text{Director}(d_1) \wedge \text{earned}(d_1, a) \wedge \text{Actor}(x_2) \wedge \text{playsIn}(x_2, m_2) \wedge \text{Movie}(m_2) \wedge \text{directedBy}(m_2, d_2) \wedge \text{Director}(d_2) \wedge \text{earned}(d_2, a) \wedge \text{married}(x_1, x_2) \wedge \text{Award}(a)$$

We have that the pair ('Tom Cruise', 'Nicole Kidman') is in $cert(\mathbf{q}, \langle \mathcal{T}, \mathcal{A} \rangle)$.

Ontology-based query answering – Example

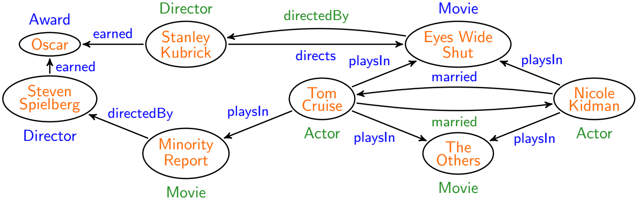
TBox $\mathcal{T}$:

$\exists$playsIn $\sqsubseteq$ Actor
$\exists$playsIn$^-$ $\sqsubseteq$ Movie
married $\equiv$ married$^-$

$\exists$directs $\sqsubseteq$ Director
Movie $\sqsubseteq$ $\exists$directedBy
directs $\equiv$ directedBy$^-$

...

ABox $\mathcal{A}$:



Leveraging the TBox axioms, we infer additional ABox facts.

Query: Return all pairs of married actors who played in movies whose directors have earned the same award.

$\mathbf{q}(x_1, x_2) \leftarrow \exists m_1, m_2, d_1, d_2, a.$ Actor$(x_1) \wedge$ playsIn$(x_1, m_1) \wedge$ Movie$(m_1) \wedge$
directedBy$(m_1, d_1) \wedge$ Director$(d_1) \wedge$ earned$(d_1, a) \wedge$
Actor$(x_2) \wedge$ playsIn$(x_2, m_2) \wedge$ Movie$(m_2) \wedge$
directedBy$(m_2, d_2) \wedge$ Director$(d_2) \wedge$ earned$(d_2, a) \wedge$
married$(x_1, x_2) \wedge$ Award(a)

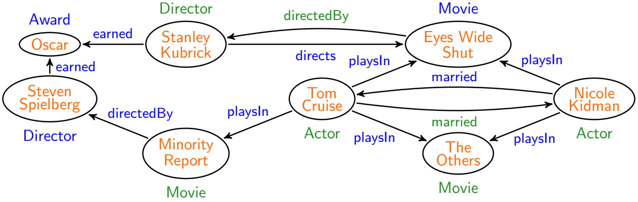
We have that the pair ('Tom Cruise', 'Nicole Kidman') is in $cert(\mathbf{q}, \langle\mathcal{T}, \mathcal{A}\rangle)$.

Ontology-based query answering – Example

TBox $\mathcal{T}$:

$$\exists \mathsf{playsIn} \sqsubseteq \mathsf{Actor} \qquad \exists \mathsf{directs} \sqsubseteq \mathsf{Director} \qquad \cdots$$
$$\exists \mathsf{playsIn}^- \sqsubseteq \mathsf{Movie} \qquad \mathsf{Movie} \sqsubseteq \exists \mathsf{directedBy}$$
$$\mathsf{married} \equiv \mathsf{married}^- \qquad \mathsf{directs} \equiv \mathsf{directedBy}^-$$

ABox $\mathcal{A}$:



Leveraging the TBox axioms, we infer additional ABox facts.

Query: Return all pairs of married actors who played in movies whose directors have earned the same award.

$$\mathbf{q}(x_1, x_2) \leftarrow \exists m_1, m_2, d_1, d_2, a.\ \mathsf{Actor}(x_1) \land \mathsf{playsIn}(x_1, m_1) \land \mathsf{Movie}(m_1) \land \mathsf{directedBy}(m_1, d_1) \land \mathsf{Director}(d_1) \land \mathsf{earned}(d_1, a) \land \mathsf{Actor}(x_2) \land \mathsf{playsIn}(x_2, m_2) \land \mathsf{Movie}(m_2) \land \mathsf{directedBy}(m_2, d_2) \land \mathsf{Director}(d_2) \land \mathsf{earned}(d_2, a) \land \mathsf{married}(x_1, x_2) \land \mathsf{Award}(a)$$

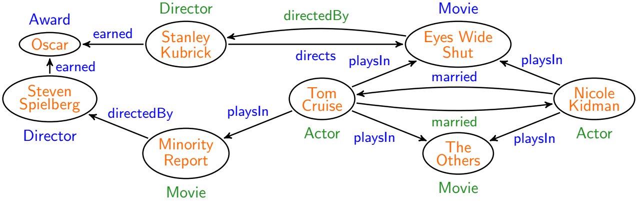
We have that the pair ('Tom Cruise', 'Nicole Kidman') is in $cert(\mathbf{q}, \langle \mathcal{T}, \mathcal{A} \rangle)$.

Ontology-based query answering – Example

TBox $\mathcal{T}$:

$$\exists\mathsf{playsIn} \sqsubseteq \mathsf{Actor} \qquad \exists\mathsf{directs} \sqsubseteq \mathsf{Director} \qquad \cdots$$
$$\exists\mathsf{playsIn}^- \sqsubseteq \mathsf{Movie} \qquad \mathsf{Movie} \sqsubseteq \exists\mathsf{directedBy}$$
$$\mathsf{married} \equiv \mathsf{married}^- \qquad \mathsf{directs} \equiv \mathsf{directedBy}^-$$

ABox $\mathcal{A}$:



Leveraging the TBox axioms, we infer additional ABox facts.

Query: Return all pairs of married actors who played in movies whose directors have earned the same award.

$$\mathbf{q}(x_1, x_2) \leftarrow \exists m_1, m_2, d_1, d_2, a.\ \mathsf{Actor}(x_1) \wedge \mathsf{playsIn}(x_1, m_1) \wedge \mathsf{Movie}(m_1) \wedge \mathsf{directedBy}(m_1, d_1) \wedge \mathsf{Director}(d_1) \wedge \mathsf{earned}(d_1, a) \wedge \mathsf{Actor}(x_2) \wedge \mathsf{playsIn}(x_2, m_2) \wedge \mathsf{Movie}(m_2) \wedge \mathsf{directedBy}(m_2, d_2) \wedge \mathsf{Director}(d_2) \wedge \mathsf{earned}(d_2, a) \wedge \mathsf{married}(x_1, x_2) \wedge \mathsf{Award}(a)$$

We have that the pair ('Tom Cruise', 'Nicole Kidman') is in $cert(\mathbf{q}, \langle\mathcal{T}, \mathcal{A}\rangle)$.

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
OWL 2	$\geqslant$ 2ExpTime	coNP-hard

Challenges

- Can we find interesting DLs for which the query answering problem can be solved efficiently (in AC^0 wrt data complexity)?
- If yes, can we delegate query answering over ontologies to a relational engine?

Complexity of conjunctive query answering in DLs

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

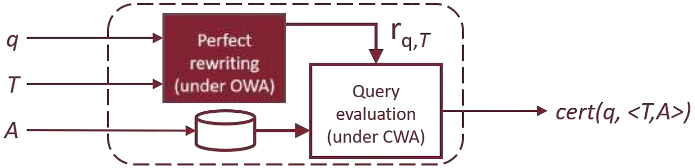
	Combined complexity	Data complexity
Plain databases	NP-complete	AC^0
OWL 2	$\geqslant$ 2ExpTime	coNP-hard

Challenges

- Can we find interesting DLs for which the query answering problem can be solved efficiently (in AC^0 wrt data complexity)?
- If yes, can we delegate query answering over ontologies to a relational engine?

To be able to deal with data efficiently, we need to separate the contribution of the data $\mathcal{A}$ from the contribution of **q** and $\mathcal{T}$.

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



Query answering is done in two phases:

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

$$\rightsquigarrow \quad cert(\mathbf{q}, \langle \mathcal{T}, \mathcal{A} \rangle)$$

The expressiveness of the ontology language affects the query language into which we are able to express the rewriting $r_{q,\mathcal{T}}$..

FOL-rewritability

We are especially interested in ontology languages for which the rewriting of CQs w.r.t. to $\mathcal{T}$ can be expressed in first-order logic (**FOL-rewritability**).

Because:

- the rewriting can be expressed in SQL, and so...
- query evaluation can be delegated to a relational DBMS.

FOL-rewritability

We are especially interested in ontology languages for which the rewriting of CQs w.r.t. to $\mathcal{T}$ can be expressed in first-order logic (**FOL-rewritability**).

Because:

- the rewriting can be expressed in SQL, and so...
- query evaluation can be delegated to a relational DBMS.

Question

- Can we find interesting DLs for which query answering is FOL-rewritable?

The *DL-Lite* family

DL-Lite is 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.

Nevertheless they have the right expressive power: capture the essential features of conceptual modeling formalisms (except completeness in hierarchies).

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 2.

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)*	$(\textbf{funct}\ livesIn)$ $(\textbf{funct}\ isFatherOf^-)$

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

End Part I

(*) thanks to Freepik.com for some of the icons used in the presentation. (icons created by Sapann-Design - Freepik.com).