COMP718: Ontologies and Knowledge Bases

Lectures 9 and 10: ontology-based data access

C. Maria Keet
School of Mathematics, Statistics, and Computer Science,
University of KwaZulu-Natal, South Africa

April 17, 2012

The following slides are heavily based on David Toman's slides of his seminar at UKZN d.d. 29-3-2011; slides used with permission

Ontology-Based Data Access: Options

David Toman

D.R. Cheriton School of Computer Science,
University of Waterloo, Canada

Joint work with:

R. Kontchakov, C. Lutz, F. Wolter, and M. Zakharyaschev
E. Franconi and G. Weddell

Queries and Ontologies

Ontology-based Data Access

Enriches query answers over *explicitly represented data* using *background knowledge* (captured using an *ontology*.)

Example

- Bob is a BOSS (explicit data)
- Every BOSS is an EMPloyee (ontology)

List all EMPloyees $\Rightarrow \{\text{Bob}\}$ (query)

Setup

$$(\mathcal{A}, \mathcal{T}) \xrightarrow{Q} \mathcal{A}'$$

$\mathcal{A}$	"the data"	set of *ground tuples*: $BOSS(Bob)$
$\mathcal{T}$	"the knowledge"	FO† sentences: $\forall x.BOSS(x) \rightarrow EMP(x)$
Q	"the question"	a FO† formula: $EMP(x)$

† or an appropriate fragment of FO

What is this good for?

1. Enriches explicit data with background knowledge
2. Physical Data Independence

Interpretations and Models, Data and Queries

Interpretation $\mathcal{I}$:

- A Domain Δ of *objects*
- An Interpretation Function $(\cdot)^{\mathcal{I}}$ that maps

  constants to objects and predicates to sets of tuples of objects

Models

A *model* of a *formula (set of formulas)* is an interpretation that makes the formula (all formulas in the set) true.

What does $\mathcal{A} = \{\mathtt{Emp}(Bob), \mathtt{Emp}(Sue)\}$ mean?

OWA: $Bob^{\mathcal{I}} \in \mathtt{Emp}^{\mathcal{I}}, Sue^{\mathcal{I}} \in \mathtt{Emp}^{\mathcal{I}}$ (KR folks)

CWA: $\{Bob^{\mathcal{I}}, Sue^{\mathcal{I}}\} = \mathtt{Emp}^{\mathcal{I}}$ (DB folks)

How do we Answer Queries: The Simple Answer

Logical Implication

A *set of formulas* entails ($\models$) *another formula* if every *model* of the former is also model of the later.

Definition (Query Answering)

$$Q(\mathcal{A}, \mathcal{T}) = \{\vec{a} \mid \mathcal{T} \cup \mathcal{A} \models Q[\vec{a}]\}$$

Operationally (with *standard names*):

$$Q(\mathcal{A}, \mathcal{T}) = \underbrace{\bigcap_{\mathcal{I} \models \mathcal{T} \cup \mathcal{A}}}_{\boxed{\text{this is a problem}}} \underbrace{Q(\mathcal{I})}_{\boxed{\text{but this is not}}}$$

Running rather Slowly, Eh?

Example

- relations: "$ColNode(x, y)$" and "$Edge(x, y)$";
- ontology: $\forall x.Node(x) \rightarrow \exists y.ColNode(x, y)$,
  $\forall x, y.ColNode(x, y) \rightarrow Colour(y)$;
- the data: a graph $(Node^{\mathcal{I}}, Edge^{\mathcal{I}})$, and
  $Colour^{\mathcal{I}} = \{r, g, b\}$.

What does the following query say?

$$\exists x, y, z.Edge(x, y) \wedge ColNode(x, z) \wedge ColNode(y, z)$$

"the graph $(\mathtt{Node}, \mathtt{Edge})$ is NOT 3-colourable"

How do we Answer Queries Efficiently?

Problem

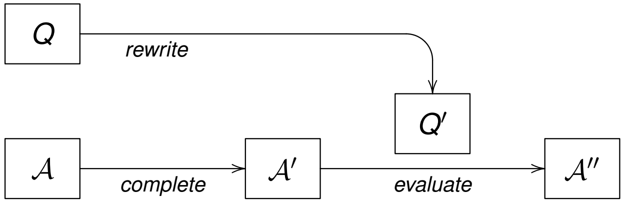
The KB has TOO MANY MODELS (so we have to look at many)

1. $(\mathcal{T}, \mathcal{A})$ have exactly one model $\mathcal{I}$: then $Q(\mathcal{A}, \mathcal{T}) = Q(\mathcal{I})$
   . . . this is how *people will think about query answering* anyway!
2. $(\mathcal{T}, \mathcal{A})$ have many models, say $\mathcal{I}_j$ $(j \in J)$:

   Option I: restrict $\mathcal{T}$ to make it feasible: *(simple) Horn theories*
   $\Rightarrow$ canonical (Herbrand) models (and small ones!)
   $\Rightarrow$ but this works well *only for positive queries!*

   Option II: restrict Q to make it feasible: those
   for which it doesn't matter which model is used
   $\Rightarrow$ e.g., safe queries in Codd's relational model

Option I



v1.0: *rewrite*: incorporate $\mathcal{T}$ into Q,
complete: an identity ($\mathcal{A}' = \mathcal{A}$)
...[Calvanese et al.]

v2.0: *rewrite*: rewrite independently of $\mathcal{T} \cup \mathcal{A}$,
complete: incorporate $\mathcal{T}$ into $\mathcal{A}$
...[Lutz et al.]

How to make $\mathcal{T}$ Easy?

Definition (DL-Lite$_{\text{horn}}$)

roles: $R ::= P \mid P^{-}$, concepts: $C ::= \bot \mid A \mid \exists R$.

1. An *ontology (TBox)* is a finite set $\mathcal{T}$ of *concept* inclusions $C_1 \sqcap \cdots \sqcap C_n \sqsubseteq C$;
2. The *Data (ABox)* is a finite set $\mathcal{A}$ of *concept* and *role* assertions $C(a)$ and $R(a, b)$;
3. A Conjunctive Query (CQ):
   an existentially quantified finite conjunction of atoms.

The Master Plan (v1.0)

IDEA:

1. Incorporate the background knowledge (i.e., $\mathcal{T}$) into the *query*.
2. Use the *rewritten query* against the ABox $\mathcal{A}$
   $\Rightarrow$ and use a relational system to do this *efficiently*.

Example

$\mathcal{T} = \{EMP \sqsubseteq \exists MANAGES, \exists MANAGES^{-} \sqsubseteq BOSS, BOSS \sqsubseteq EMP\}$

$\mathcal{A} = \{BOSS(Bob), EMP(Sue)\}$

$$Q(x,z) \leftarrow \exists y.MANAGES(x,y) \wedge MANAGES(z,y)$$
$$Q(x,x) \leftarrow \exists y.MANAGES(x,y) \qquad (factor)$$
$$Q(x,x) \leftarrow EMP(x) \qquad \mathcal{T}(1)$$
$$Q(x,x) \leftarrow BOSS(x) \qquad \mathcal{T}(3)$$

The Master Plan (v2.0)

IDEA:

1. Incorporate the background knowledge (i.e., $\mathcal{T}$) into the *data*.
   $\Rightarrow$ make *implicit knowledge explicit (data completion)*.
2. Use the *data completion* (only) to answer queries
   $\Rightarrow$ and use a relational system to do this *efficiently*.

Example

$\mathcal{T} = \{BOSS \sqsubseteq EMP\}$, $\mathcal{A} = \{BOSS(Bob)\}$, $Q \equiv EMP(x)$

1. $\mathcal{I}_{\mathcal{K}} = \{BOSS(Bob), EMP(Bob)\}$ (data completion)
2. $Q(\mathcal{I}_{\mathcal{K}}) = \{Bob\}$ (relational query)

Canonical Interpretations

ABox completion: the Canonical Interpretation $\mathcal{I}_\mathcal{K}$

$A^{\mathcal{I}_\mathcal{K}} = \{a \in \mathsf{Ind}(\mathcal{A}) \mid \mathcal{K} \models A(a)\} \cup \{c_R \in \Delta^{\mathcal{I}_\mathcal{K}} \mid \mathcal{T} \models \exists R^- \sqsubseteq A\}$,

$P^{\mathcal{I}_\mathcal{K}} = \{(a,b) \in \mathsf{Ind}(\mathcal{A}) \times \mathsf{Ind}(\mathcal{A}) \mid P(a,b) \in \mathcal{A}\} \cup$

$\quad \{(d, c_P) \in \Delta^{\mathcal{I}_\mathcal{K}} \times \mathsf{N}_\mathsf{I}^\mathcal{T} \mid d \rightsquigarrow c_P\} \cup \{(c_{P^-}, d) \in \mathsf{N}_\mathsf{I}^\mathcal{T} \times \Delta^{\mathcal{I}_\mathcal{K}} \mid d \rightsquigarrow c_{P^-}\}$

... c_R's only used "when necessary" (for *generating* roles)

Example

$\mathcal{T} = \{EMP \sqsubseteq \exists MANAGES, \exists MANAGES^- \sqsubseteq BOSS, BOSS \sqsubseteq EMP\}$

$\mathcal{A} = \{BOSS(Bob), EMP(Sue)\}$

Then $EMP^{\mathcal{I}_\mathcal{K}} = \{Bob, Sue, \}$, $BOSS^{\mathcal{I}_\mathcal{K}} = \{Bob, \}$, and
$MANAGES^{\mathcal{I}_\mathcal{K}} = \{(Bob,), (Sue,), (,)\}$.

Lemma

- $q_A^\mathcal{T}$ *s.t.* $\mathsf{ans}(q_A^\mathcal{T}, \mathcal{A}) = A^{\mathcal{I}_\mathcal{K}}$, *and*
- $q^\mathcal{T}$ s.t. ans$(q^\mathcal{T}$, $\mathcal{A}) = P^{\mathcal{I}_\mathcal{K}}$

Query Rewriting (TBox-free)

Example

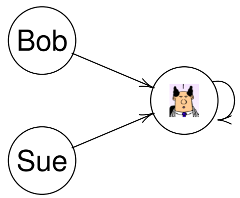
$\mathcal{T} = \{EMP \sqsubseteq \exists MANAGES, \exists MANAGES^- \sqsubseteq BOSS, BOSS \sqsubseteq EMP\}$

$\mathcal{A} = \{EMP(Bob), EMP(Sue)\}$

Queries:

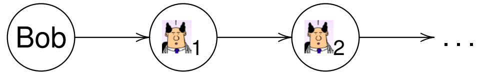
1. $\exists v.MANAGES(v, v)$
2. $\exists y.MANAGES(x, y) \wedge MANAGES(z, y)$

$\mathcal{I}_\mathcal{K}$:



$Q_1(\mathcal{I}_\mathcal{K}) = \text{true}$
$Q_2(\mathcal{I}_\mathcal{K}) = \{(Bob, Sue)\}$

$\mathcal{U}_\mathcal{K}$:



$Q_1(\mathcal{U}_\mathcal{K}) = \text{false}$

v1.0 vs. v2.0

	v1.0 (query rewriting)	v2.0 (data completion)
Queries	rewriting is exponential in $\|Q\|$	data only grows polynomially in $\|\mathcal{A}\|$
Updates	applies to original data	needs rematerialize data completion

Option II: Exact Answers

IDEA:

Restrict *queries* to those
whose answer does NOT depend on the choice of model of $\mathcal{T} \cup \mathcal{A}$:

$$\textit{for all } \mathcal{I}, \mathcal{J} \models \mathcal{T} \cup \mathcal{A} \textit{ we have } Q(\mathcal{I}) = Q(\mathcal{J})$$

In practice—given $\mathcal{T}$, Q, and *FIXED signature for $\mathcal{A}$*:

$$\textit{for all } \mathcal{I}, \mathcal{J} \models \mathcal{T} \cup \mathcal{A} \textit{ we have } Q(\mathcal{I}) = Q(\mathcal{J}) \qquad (*)$$

for *every choice of $\mathcal{A}$ over the FIXED signature*.

Advantages: no restrictions of $\mathcal{T}$ and Q
(modulo deciding whether the condition $(*)$ holds)

Issues: how does this help us??
a FO rewriting *over $\mathcal{A}$* exists $\Rightarrow$ a relational query

Beth Definability and Interpolation

How do we test for $(*)$?

Beth Definability

Q satisfies $(*)$ if

$$\mathcal{T} \cup \mathcal{T}' \models Q \rightarrow Q'$$

where $\mathcal{T}'$ (Q') is $\mathcal{T}$ (Q) in which symbols *NOT in* $\mathcal{A}$ are *primed*.

. . . this only works under CWA!

How do we rewrite Q?

Craig Interpolation

$\models \varphi \rightarrow \psi$ then $\models \varphi \rightarrow \gamma \rightarrow \psi$,

where γ only uses non-logical symbols common to φ and ψ.

Exercise: use the above to show $\mathcal{T} \cup \mathcal{T}' \models Q \rightarrow P \rightarrow Q'$

Observations

- Either Option I+OWA or
  Option II+CWA(+standard names), but not both
- Applications:
  - KR (mostly Option I and OWA)
    - ⇒ Medical ontologies and patient records, (Bio-)sciences in general
    - ⇒ Information Integration
  - DB (almost exclusively Option II and CWA)
    - ⇒ Physical Design and Data Structures
    - ⇒ Query Optimization, Materialized Views, etc.

References

Option I, v1.0: D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Tractable reasoning and efficient query answering in description logics: The DL-Lite family. J. of Automated Reasoning, 39(3):385-429, 2007.

Option I, v2.0: C. Lutz, D. Toman, and F. Wolter. Conjunctive query answering in the description logic EL using a relational database system. In Proc. IJCAI, 2070-2075, 2009.

R. Kontchakov, C. Lutz, D. Toman, F. Wolter, and M. Zakharyaschev. The combined approach to query answering in DL-Lite. In Proc. KR, 2010.

Option II: D. Toman and G. Weddell. Fundamentals of Physical Design and Query Compilation. Morgan and Claypool, Synthesis lectures, *Data Management Series*. 2011.