The Combined Approach to Ontology-Based Data Access

R. Kontchakov, C. Lutz, D. Toman, F.Wolter and M. Zakharyaschev

Presented by Amer Mouawad
University of Waterloo

July 8, 2013

Ontology-Based Data Access (OBDA)

Motivation:

- Data enrichment (through inference)
- Separation of concerns: Users are generally not interested in how or where data is stored
- Provide a user-oriented view of the data
- Queries are formulated in the language of the ontology

Ontology-Based Data Access (OBDA)

Motivation:

- Data enrichment (through inference)
- Separation of concerns: Users are generally not interested in how or where data is stored
- Provide a user-oriented view of the data
- Queries are formulated in the language of the ontology

Notation:

- $\mathcal{T}$ is given by a finite set of sentences of FO logic
- $\mathcal{D}$ is given by a finite set of ground atoms $P(a_1, ..., a_n)$
- $a_1, ..., a_n$ are constants
- A query $q(\vec{x})$ is an FO-formula with free variables $\vec{x}$

Ontology-Based Data Access (OBDA)

Example:

- All MEN are MORTAL (ontology)
- Socrates is a MAN (explicit data)
- List all mortals => {Socrates}

Ontology-Based Data Access (OBDA)

Example:

- All MEN are MORTAL (ontology)
- Socrates is a MAN (explicit data)
- List all mortals => {Socrates}

Problems:

- $\mathcal{D}$ is incomplete
- Potentially infinite set of possible models of $\mathcal{T}$ and $\mathcal{D}$
- $q(\vec{x})$ must be true in every FO-model $\mathcal{M}$ of $\mathcal{T}$ and $\mathcal{D}$ (certain answers as opposed to RDBMS)
- OBDA should scale to large amounts of data and be as efficient as RDBMS

Ontology-Based Data Access (OBDA)

Given $\mathcal{T}$, $\mathcal{D}$, and $q(\vec{x})$, the general problem is to compute a finite FO model $\mathcal{D}'$ and an FO query $q'(\vec{x})$ such that the following properties hold:

- **(ans)**: $\vec{a}$ is an answer to $q'(\vec{x})$ over $\mathcal{D}'$ iff $\vec{a}$ is a *certain answer* to $q(\vec{x})$ over $\mathcal{T}$ and $\mathcal{D}$
- **(dat)**: $\mathcal{D}'$ is computable in polynomial time in $\mathcal{D}$ and does not depend on $q(\vec{x})$
- **(que)**: $q'(\vec{x})$ does not depend on $\mathcal{D}$

Ontology-Based Data Access (OBDA)

Given $\mathcal{T}$, $\mathcal{D}$, and $q(\vec{x})$, the general problem is to compute a finite FO model $\mathcal{D}'$ and an FO query $q'(\vec{x})$ such that the following properties hold:

- **(ans)**: $\vec{a}$ is an answer to $q'(\vec{x})$ over $\mathcal{D}'$ iff $\vec{a}$ is a *certain answer* to $q(\vec{x})$ over $\mathcal{T}$ and $\mathcal{D}$
- **(dat)**: $\mathcal{D}'$ is computable in polynomial time in $\mathcal{D}$ and does not depend on $q(\vec{x})$
- **(que)**: $q'(\vec{x})$ does not depend on $\mathcal{D}$

Various refinements of these conditions have been studied. Replacing **(dat)** by $\mathcal{D}' = \mathcal{D}$ is one example which guarantees the same data complexity as in RDBMSs but rewritten queries may be exponential in the size of q (Calvanese et al., 2007)

Ontology-Based Data Access (OBDA)

This paper suggests the use of two different conditions:

- **(dat')**: $\mathcal{D}'$ is computable in polynomial time in both $\mathcal{T}$ and $\mathcal{D}$, preferably using RDBMSs
- **(que')**: $q'(\vec{x})$ is polynomial in $\mathcal{T}$ and $q(\vec{x})$

Notes: Source data has to be manipulated, no exponential blowups.

Description Logic: DL-Lite$_{horn}$

Reminder:

- Concepts (unary predicates in FO)
- Domains and ranges of roles (binary relations in FO)
- Roles R and concepts C are built from concept names A_i and role names P_i, $i \geq 0$, according to the following syntax rules:
  - $R ::= P_i \mid P_i^-$
  - $C ::= \bot \mid \top \mid A_i \mid \exists R$

- A DL-Lite$_{horn}$ TBox, $\mathcal{T}$, is a finite set of concept inclusions
- A DL-Lite$_{horn}$ ABox, $\mathcal{A}$, is a finite set of concept and role assertions, which is used to store instance data

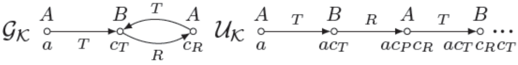
Description Logic: DL-Lite$_{horn}$

- A DL-Lite$_{horn}$ knowledge base (KB) is a pair $\mathcal{K} = (\mathcal{T}, \mathcal{A})$
- An interpretation $\mathcal{I}$ is a model of a KB if $\mathcal{I} \models \alpha$ for all $\alpha \in \mathcal{T} \cup \mathcal{A}$
- $\mathcal{K} \models \alpha$ whenever $\mathcal{I} \models \alpha$ for all models $\mathcal{I}$ of $\mathcal{K}$
- $\mathcal{K}$ is consistent if it has a model

Description Logic: DL-LITE_{horn}

- A DL-LITE_{horn} knowledge base (KB) is a pair $\mathcal{K} = (\mathcal{T}, \mathcal{A})$
- An interpretation $\mathcal{I}$ is a model of a KB if $\mathcal{I} \models \alpha$ for all $\alpha \in \mathcal{T} \cup \mathcal{A}$
- $\mathcal{K} \models \alpha$ whenever $\mathcal{I} \models \alpha$ for all models $\mathcal{I}$ of $\mathcal{K}$
- $\mathcal{K}$ is consistent if it has a model

Consider $\mathcal{K} = (\mathcal{T}\ , \{\mathcal{A}\ (a)\})$ where
$\mathcal{T} = \{\mathcal{A} \sqsubseteq \exists\mathcal{T}, \exists\mathcal{T}^- \sqsubseteq \mathcal{B}, \mathcal{B} \sqsubseteq \exists\mathcal{R}, \exists\mathcal{R}^- \sqsubseteq \mathcal{A}\}$ and
let $q(x) = \exists y, z(T(x,y) \wedge R(y,z) \wedge T(z,y))$
$\Rightarrow$ a is an answer to $q(x)$ in $\mathcal{G}_{\mathcal{K}}$, but not a certain answer to $q(x)$ over $\mathcal{K}$



Description Logic: DL-LITE$_{horn}$

Problem: Given a DL-LITE$_{horn}$ knowledge base $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ and a conjunctive query $q(\vec{x})$, compute (in poly time if possible) a finite FO-structure $\mathcal{G}_{\mathcal{K}}$, independently from $q(\vec{x})$, and an FO-query $q'(\vec{x})$, independently from A, such that **(dat')**, **(dat')**, and **(ans)** hold: for every tuple $\vec{a} \subseteq Ind(\mathcal{A})$, $\vec{a} \in cert(q, \mathcal{K})$ iff $\vec{a} \in ans(q', \mathcal{G}_{\mathcal{K}})$

Description Logic: DL-Lite$_{horn}$

Problem: Given a DL-Lite$_{horn}$ knowledge base $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ and a conjunctive query $q(\vec{x})$, compute (in poly time if possible) a finite FO-structure $\mathcal{G}_{\mathcal{K}}$, independently from $q(\vec{x})$, and an FO-query $q'(\vec{x})$, independently from A, such that **(dat')**, **(dat')**, and **(ans)** hold: for every tuple $\vec{a} \subseteq Ind(\mathcal{A})$, $\vec{a} \in cert(q, \mathcal{K})$ iff $\vec{a} \in ans(q', \mathcal{G}_{\mathcal{K}})$

$\Rightarrow$ The key to the solution is the existence of canonical models for Horn theories which give all correct answers to CQs

Canonical Models

Some definitions for a KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$:

- $N^{\mathcal{T}} = \{c_P, c_{P^-} \mid P$ is a role name in $\mathcal{T}\}$ is a set of "new" individual names (disjoint from $Ind(\mathcal{A})$).
- A role R is called generating in $\mathcal{K}$ if there exist $a \in Ind(\mathcal{A})$ and $R_0, ..., R_n = R$ such that:
  - **(agen)**: $\mathcal{K} \models \exists R_0(a)$ but $R_0(a, b) \notin \mathcal{A}$ for all $b \in Ind(\mathcal{A})$ (written as $a \rightsquigarrow c_{R_0}$)
  - **(rgen)**: for $i \leq n$, $\mathcal{T} \models \exists R_i^- \sqsubseteq R_{i+1}$ and $R_i^- \neq R_{i+1}$ (written as $c_{R_i} \rightsquigarrow c_{R_{i+1}}$)

Canonical Models

The model $\mathcal{G}_{\mathcal{K}}$ for $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ is defined as follows:

$$\Delta^{\mathcal{G}_{\mathcal{K}}} = Ind(\mathcal{A}) \cup \{c_R \mid R \in N^{\mathcal{T}}, R \text{ is generating in } \mathcal{K}\}$$

$$a^{\mathcal{G}_{\mathcal{K}}} = a, \text{ for all } a \in Ind(\mathcal{A})$$

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

$$\begin{aligned} P^{\mathcal{G}_{\mathcal{K}}} = {} & \{(a, b) \in Ind(\mathcal{A}) \times Ind(\mathcal{A}) \mid P(a, b) \in \mathcal{A}\} \\ & \cup \{(d, c_P) \in \Delta^{\mathcal{G}_{\mathcal{K}}} \times N^{\mathcal{T}} \mid d \rightsquigarrow c_P\} \\ & \cup \{(c_{P^-}, d) \in N^{\mathcal{T}} \times \Delta^{\mathcal{G}_{\mathcal{K}}} \mid c_{P^-} \rightsquigarrow d\} \end{aligned}$$

Canonical Models

The model $\mathcal{G}_{\mathcal{K}}$

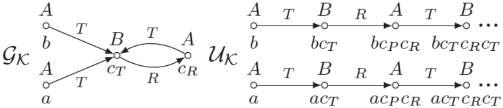
- can be built in time polynomial in $|\mathcal{K}|$ and thus satisfies **(dat')**
- is not in general a model of $\mathcal{K}$ (finiteness)
- does NOT always give correct answers to queries (without modifications)

Canonical Models

The model $\mathcal{G}_{\mathcal{K}}$

- can be built in time polynomial in $|\mathcal{K}|$ and thus satisfies **(dat')**
- is not in general a model of $\mathcal{K}$ (finiteness)
- does NOT always give correct answers to queries (without modifications)

Another example: $\mathcal{K} = (\mathcal{T}, \{A(a), A(b)\})$ where
$\mathcal{T} = \{\mathcal{A} \sqsubseteq \exists\mathcal{T}, \exists\mathcal{T}^- \sqsubseteq \mathcal{B}, \mathcal{B} \sqsubseteq \exists\mathcal{R}, \exists\mathcal{R}^- \sqsubseteq \mathcal{A}\}$ and
let $q(x_1, x_2) = \exists y(T(x_1, y) \wedge T(x_2, y))$
$\Rightarrow$ (a, b) is an answer to $q(x)$ in $\mathcal{G}_{\mathcal{K}}$, but not a certain answer to $q(x)$ over $\mathcal{K}$



Canonical Models

The solution to the problem is two-fold:

- First, it is showed that by "unraveling" $\mathcal{G}_{\mathcal{K}}$ into a (possibly infinite) homomorphic model $\mathcal{U}_{\mathcal{K}}$, we can guarantee $cert(q, \mathcal{K}) = ans(q, \mathcal{U}_{\mathcal{K}}) \subseteq ans(q, \mathcal{G}_{\mathcal{K}})$ for every consistent DL-LITE$_{horn}$ KB $\mathcal{K}$ and every positive existential query q.
- Secondly, a query rewriting algorithm is proposed which converts any q into some q' such that $ans(q', \mathcal{G}_{\mathcal{K}}) = ans(q, \mathcal{U}_{\mathcal{K}})$.

Conjunctive Query Answering

- We are given a CQ $q(\vec{x}) = \exists \vec{y}.\sigma(\vec{x}, \vec{y})$ and the goal is to find a rewriting, $q^\star$, such that
  (i) for every DL-Lite$_{horn}$ KB $\mathcal{K}$, $cert(q, \mathcal{K}) = ans(q^\star, \mathcal{G}_\mathcal{K})$ and
  (ii) the size of $q^\star$ is polynomial in the size of q.

Conjunctive Query Answering

- We are given a CQ $q(\vec{x}) = \exists\vec{y}.\sigma(\vec{x},\vec{y})$ and the goal is to find a rewriting, $q^\star$, such that
  (i) for every DL-LITE$_{horn}$ KB $\mathcal{K}$, $cert(q,\mathcal{K}) = ans(q^\star,\mathcal{G}_\mathcal{K})$ and
  (ii) the size of $q^\star$ is polynomial in the size of q.
- $q^\star = \exists\vec{y}(\sigma \wedge \sigma_1 \wedge \sigma_2 \wedge \sigma_3)$
  (i) where σ_1, σ_2, and σ_3 are boolean combinations of equalities $t_1 = t_2$
  (ii) and t_i is either a term in q or a constant $c_R \in N^\mathcal{T}$.

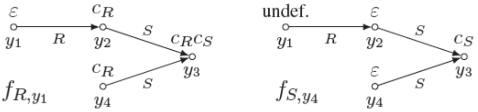
Conjunctive Query Answering

- $\sigma_1 = \bigwedge_{x \in \vec{x}} \bigwedge_{c_R \in N^{\mathcal{T}}} (x \neq c_R)$
  - σ_1 guarantees that no tuples in the answer can contain an "unknown" or "null" value
  - The size of σ_1 is polynomial in q and $\mathcal{T}$ **(que')**.

Conjunctive Query Answering

- Let $N^* = N^{\mathcal{T}} \cup \epsilon$ (the empty string).
- Let q be a CQ and $R(t, t') \in q$.
- Identify q with the set of its atoms and use $P^-(t, t') \in q$ as a synonym of $P(t', t) \in q$.
- A partial function f : terms of $q \to N^*$ is a tree-witness for (R, t) if its domain is minimal such that $f(t) = \epsilon$ and for all $S(s, s') \in q$
  - If $f(s) = \epsilon$, then $f(s') = c_R$ (provided $S = R$)
  - If $f(s) = \omega c_T$, then $f(s') = \begin{cases} \omega, & if\ T = S^- \\ \omega c_T c_S & otherwise \end{cases}$

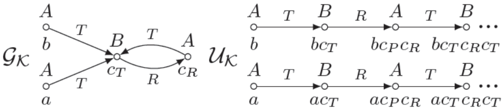
Example Let $q = \{R(y_1, y_2), S(y_2, y_3), S(y_4, y_3)\}$. Then the tree witnesses for (R, y_1) and (S, y_4) in q are:



Conjunctive Query Answering

- $\sigma_2 = \bigwedge_{R(t,t') \in q,\ tw\ (R,t)\ exists}((t' = c_R) \to \bigwedge_{f_{R,t}(s)=\epsilon}(s = t))$
  - σ_2 guarantees that no tuples in the answer were the result of a "join" on null or unknown values
  - The size of σ_2 is polynomial in q and $\mathcal{T}$ **(que')** (poly-time for tree-witness testing).

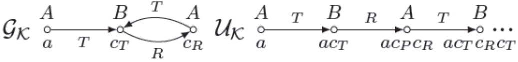
Back to our example where $q(x_1, x_2) = \exists y(T(x_1, y) \wedge T(x_2, y))$
$\Rightarrow$ As $f_{T,x_1}(x2) = \epsilon$, we have $(y = c_T) \to (x1 = x2)$ in σ_2, which prevents the spurious (a, b) answer



Conjunctive Query Answering

- $\sigma_3 = \bigwedge_{R(t,t')\in q,\ tw\ (R,t)\ !exists}(t' \neq c_R)$
  - If the tree witness for (R,t) does not exist, then there are two paths from $R(t,t')$ to some term $s \in q$. σ_3 guarantees that reaching such a term cannot be through null or unknown values
  - The size of σ_3 is polynomial in q and $\mathcal{T}$ **(que')** (poly-time for tree-witness testing).

Back to our example where
$q(x) = \exists y, z(T(x,y) \wedge R(y,z) \wedge T(z,y))$
$\Rightarrow$ There exist no tree witnesses for (R,y), (R^-,z), (T,z) and (T^-,y). This gives four conjuncts $(z \neq c_R)$, $(y \neq c_{R^-})$, $(y \neq c_T)$ and $(z \neq c_{T^-})$ which prevent the spurious answer (a)



Conclusion

- Using the combined approach, query rewriting can be done without an exponential blowup
- Experimental evidence suggest that the efficiency of this technique is comparable to RDBMSs
- By generating the model using classical views, all the power of current RDBMSs can be exploited