Description Logics

Franz Baader
Theoretical Computer Science
TU Dresden
Germany

1. Motivation and introduction to Description Logics
2. Tableau-based reasoning procedures
3. Automata-based reasoning procedures
4. Complexity of reasoning in Description Logics
5. Reasoning in inexpressive Description Logics
6. Query answering in inexpressive Description Logics
Query Answering in databases from a logical point of view

The problem:

- A database is a finite first-order interpretation (i.e., a finite relational structure).
- A query is a first-order formula with some free variables (the answer variables). \[ \phi(x_1, \ldots, x_n) \]
- An answer tuple assigns elements of the interpretation to the free variables such that the query is satisfied.

\[ \mathcal{I} \]

Answer tuples:
\[ \{ (d_1, \ldots, d_n) \mid \mathcal{I} \models \phi(d_1, \ldots, d_n) \} \]
Query Answering in databases from a logical point of view

The problem:

- A database is a finite first-order interpretation (i.e., a finite relational structure).
- A query is a first-order formula with some free variables (the answer variables).
- An answer tuple assigns elements of the interpretation to the free variables such that the query is satisfied.

Its complexity: deciding whether there is an answer tuple is

- PSpace-complete w.r.t. combined complexity, i.e., w.r.t. the combined size of I and $\phi$. Usually I is very large and $\phi$ quite small.
- In $AC^0$ w.r.t. data complexity, i.e., w.r.t. the size of I only ($\phi$ fixed). $AC^0 \subseteq \text{LogSpace} \subseteq P$
Query Answering in databases from a logical point of view

The problem:

- A database is a finite first-order interpretation (i.e., a finite relational structure).

- A query is a first-order formula with some free variables (the answer variables).

- An answer tuple assigns elements of the interpretation to the free variables such that the query is satisfied.

In practice:

- highly efficient relational database engines available

- that scale very well to huge databases
Conjunctive queries

subclass of FOL queries

A conjunctive query (CQ)
is an existentially quantified conjunction of atoms:

$$\exists y, z.\ Q(x) \land P(x, y) \land P(y, z) \land P(z, x)$$

A union of conjunctive queries (UCQ) is a disjunction of CQs.

Complexity of CQs and UCQs:

- deciding whether there is an answer tuple is
  - NP-complete w.r.t. combined complexity.
  - In $AC^0$ w.r.t. data complexity.
Conjunctive queries

Example that shows NP-hardness w.r.t.
combined complexity

three-colorability

Conjunctive query:

\[ \exists x_1, x_2, x_3, x_4, x_5, x_6. \]
\[ E(x_1, x_2) \land E(x_2, x_3) \land \\
E(x_1, x_4) \land E(x_2, x_5) \land E(x_3, x_6) \land \\
E(x_4, x_5) \land E(x_5, x_6) \]

Database:

\[ E(\text{red, blue}) \]
\[ E(\text{red, green}) \]
\[ E(\text{green, red}) \]

The empty tuple () is an answer tuple iff the graph is three-colorable.
Generalizes answering (unions of) conjunctive queries in two directions:

- Presence of a TBox $\mathcal{T}$:
  predicates used in the CQs are constrained by TBox axioms

- Incompleteness:
  CQs evaluated over an ABox $\mathcal{A}$ rather than an interpretation
  (no closed-word assumption).

We want to compute certain answers of $\phi(x_1, \ldots, x_n)$ over $\mathcal{A}$ w.r.t. $\mathcal{T}$:

- a tuple $(a_1, \ldots, a_n)$ of individuals occurring in $\mathcal{A}$ such that
- $(a_1^\mathcal{T}, \ldots, a_n^\mathcal{T})$ is an answer tuple of $\phi(x_1, \ldots, x_n)$ over $\mathcal{I}$
- for all models $\mathcal{I}$ of $\mathcal{T}$ and $\mathcal{A}$. 
Ontology-Based Data Access (OBDA)

- One usually assumes that the ABox $\mathcal{A}$ is atomic, i.e., contains only atomic assertions of the form $A(a), r(a, b)$ for concept names $A$ and role names $r$.

- Data complexity: complexity of computing certain answers in the size of the ABox only (TBox and query assumed to be fixed).

- Combined complexity: complexity of computing certain answers in the size of the ABox, TBox, and query.

- The instance problem can be seen as a special case: $\mathcal{A} \models_{\mathcal{T}} A(e)$ iff $(e)$ is a certain answer of $A(x)$ over $\mathcal{A}$ w.r.t. $\mathcal{T}$.

  Combined complexity of computing certain answers at least as high as the complexity of the instance problem.
Ontology-Based Data Access for expressive DLs

For the DL $\mathcal{ALC}$, deciding whether there is a certain answer is

- ExpTime-complete w.r.t. combined complexity. \textit{same as instance problem}
- coNP-complete w.r.t. data complexity. \textit{not even tractable}

Adding inverse roles increases the combined complexity:
For the DL $\mathcal{ALCI}$, deciding whether there is a certain answer is

- 2ExpTime-complete w.r.t. combined complexity. \textit{higher than instance problem}
- coNP-complete w.r.t. data complexity.
Ontology-Based Data Access for inexpressive DLs

In order to deal with very large ABoxes, tractability is not sufficient.

Goal

Find DLs for which computing certain answers can be reduced to answering FOL queries using a relational database system.
Query answering using relational DB technology

TBox \rightarrow \text{Reformulation} \rightarrow \text{FOL query} \rightarrow \text{Evaluate with RDB system} \rightarrow \text{Certain answers}

CQ

ABox \rightarrow \text{view as} \rightarrow \text{RDB}

\text{FOL-reducibility holds if this is possible}
Ontology-Based Data Access for inexpressive DLs

In order to deal with very large ABoxes, tractability is not sufficient.

Goal

Find DLs for which computing certain answers can be achieved using a relational database system.

directed toward the DL-Lite family
**DL-Lite** \(_{\text{core}}\)

the basic member of the **DL-Lite family**

[Calvanese et al.; 2007]

<table>
<thead>
<tr>
<th>Concept Type</th>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Concept names</td>
<td>(A)</td>
<td></td>
</tr>
<tr>
<td>Basic concepts</td>
<td>(B)</td>
<td></td>
</tr>
<tr>
<td>General concepts</td>
<td>(C)</td>
<td></td>
</tr>
</tbody>
</table>

\[
B \rightarrow A \mid \exists r. \top \mid \exists r^{-1}. \top
\]

\[
C \rightarrow B \mid \neg B
\]

**GCI**s

\[
B \sqsubseteq C
\]

\[
\exists \text{has\_child}. \top \sqsubseteq \neg \text{Spinster}
\]

\[
\exists \text{has\_child}. \top \sqsubseteq \text{Parent}
\]

\[
\text{Parent} \sqsubseteq \text{Human}
\]

\[
\text{Human} \sqsubseteq \exists \text{has\_child}^{-1}. \top
\]

**ABox**

\[
A(a) \quad r(a, b)
\]

\[
\text{Woman}(\text{LINDA})
\]

\[
\text{has\_child}(\text{LINDA}, \text{JAMES'})
\]

\[
\text{Beatle}(\text{PAUL})
\]

\[
\text{has\_child}(\text{PAUL}, \text{JAMES'})
\]
Conjunctive query answering over a DL-Lite$_{core}$ ontology

$\exists y, z_1, z_2. \text{Woman}(x) \land \text{has\_child}(x, y) \land \text{has\_child}(z_1, y) \land \text{Human}(z_1) \land \text{has\_child}(z_2, z_1)$

free variable

certain answer: (LINDA)

TBox

$\exists \text{has\_child}. \top \sqsubseteq \lnot \text{Spinster}$

$\exists \text{has\_child}. \top \sqsubseteq \text{Parent}$

$\text{Parent} \sqsubseteq \text{Human}$

$\text{Human} \sqsubseteq \exists \text{has\_child}^{-1}. \top$

ABox

ontology

$\text{Woman}(\text{LINDA})$

$\text{has\_child}(\text{LINDA, JAMES'})$

$\text{Beatle}(\text{PAUL})$

$\text{has\_child}(\text{PAUL, JAMES'})$
FOL-reducibility of DL-Lite_{core} [Calvanese et al.; 2007]

TBox \rightarrow CQ

Reformulation

FOL query

Query reformulation generates a union of conjunctive queries by

- using GCIs with basic concepts on right-hand side as rewrite rules from right to left,

- which generate a new CQ in the union by rewriting an atom in an already obtained CQ.
FOL-reducibility of $\text{DL-Lite}_{\text{core}}$

$\exists y, z_1, z_2. \text{Woman}(x) \land \text{has\_child}(x, y) \land \text{has\_child}(z_1, y) \land \text{Human}(z_1) \land \text{has\_child}(z_2, z_1)$

$\exists y, z_1. \text{Woman}(x) \land \text{has\_child}(x, y) \land \text{has\_child}(z_1, y) \land \text{Human}(z_1) \land \text{Human}(z_1)$

TBox

$\exists \text{has\_child}. \top \sqsubseteq \neg \text{Spinster}$

$\exists \text{has\_child}. \top \sqsubseteq \text{Parent}$

$\text{Parent} \sqsubseteq \text{Human}$

$\text{Human} \sqsubseteq \exists \text{has\_child}^{-1}. \top$
FOL-reducibility of DL-Lite$_{core}$

\[
\exists y, z_1, z_2. \text{Woman}(x) \land \text{has\_child}(x, y) \land \text{has\_child}(z_1, y) \land \text{Human}(z_1) \land \text{has\_child}(z_2, z_1)
\]

\[
\exists y, z_1. \text{Woman}(x) \land \text{has\_child}(x, y) \land \text{has\_child}(z_1, y) \land \text{Human}(z_1)
\]

\[
\exists y, z_1. \text{Woman}(x) \land \text{has\_child}(x, y) \land \text{has\_child}(z_1, y) \land \text{Parent}(z_1)
\]

TBox

\[
\exists \text{has\_child}. \top \sqsubseteq \neg \text{Spinster}
\]

\[
\text{Parent} \sqsubseteq \text{Human}
\]

\[
\exists \text{has\_child}. \top \sqsubseteq \text{Parent}
\]

\[
\text{Human} \sqsubseteq \exists \text{has\_child}^{-1}. \top
\]
FOL-reducibility of DL-Lite$_{core}$

\[ \exists y, z_1, z_2. \, \text{Woman}(x) \land \text{has\_child}(x, y) \land \text{has\_child}(z_1, y) \land \text{Human}(z_1) \land \text{has\_child}(z_2, z_1) \]

\[ \exists y, z_1. \, \text{Woman}(x) \land \text{has\_child}(x, y) \land \text{has\_child}(z_1, y) \land \text{Human}(z_1) \]

\[ \exists y, z_1. \, \text{Woman}(x) \land \text{has\_child}(x, y) \land \text{has\_child}(z_1, y) \land \text{Parent}(z_1) \]

\[ \exists y, z_1, z_3. \, \text{Woman}(x) \land \text{has\_child}(x, y) \land \text{has\_child}(z_1, y) \land \text{has\_child}(z_1, z_3) \]

\[
\text{TBox} \quad \exists \text{has\_child}. \top \sqsubseteq \lnot \text{Spinster} \\
\text{Parent} \sqsubseteq \text{Human} \\
\exists \text{has\_child}. \top \sqsubseteq \text{Parent} \\
\text{Human} \sqsubseteq \exists \text{has\_child}^{-1}. \top
\]
FOL-reducibility of DL-Lite$_{core}$

\[ \exists y, z_1, z_2. \text{Woman}(x) \land \text{has\_child}(x, y) \land \text{has\_child}(z_1, y) \land \text{Human}(z_1) \land \text{has\_child}(z_2, z_1) \]

\[ \exists y, z_1. \text{Woman}(x) \land \text{has\_child}(x, y) \land \text{has\_child}(z_1, y) \land \text{Human}(z_1) \]

\[ \exists y, z_1. \text{Woman}(x) \land \text{has\_child}(x, y) \land \text{has\_child}(z_1, y) \land \text{Parent}(z_1) \]

\[ \exists y, z_1, z_3. \text{Woman}(x) \land \text{has\_child}(x, y) \land \text{has\_child}(z_1, y) \land \text{has\_child}(z_1, z_3) \]

**ABox**

| Woman(LINDA) | has\_child(LINDA, JAMES) |
| Beatle(PAUL) | has\_child(PAUL, JAMES) |

**TBox**

| \exists \text{has\_child}. \top \sqsubseteq \neg \text{Spinster} | \exists \text{has\_child}. \top \sqsubseteq \text{Parent} |
| Parent \sqsubseteq \text{Human} | Human \sqsubseteq \exists \text{has\_child}^{-1}. \top |

Dresden
FOL-reducibility of DL-Lite_{core}

\exists y, z_1, z_2. Woman(x) \land has\_child(x, y) \land has\_child(z_1, y) \land Human(z_1) \land has\_child(z_2, z_1)

\exists y, z_1. Woman(x) \land has\_child(x, y) \land has\_child(z_1, y) \land Human(z_1)

\exists y, z_1. Woman(x) \land has\_child(x, y) \land has\_child(z_1, y) \land Parent(z_1)

\exists y, z_1, z_3. Woman(x) \land has\_child(x, y) \land has\_child(z_1, y) \land has\_child(z_1, z_3)

RDB

\begin{align*}
\text{Woman(LINDA)} & \quad \text{has\_child(LINDA, JAMES)} \\
\text{Beatle(PAUL)} & \quad \text{has\_child(PAUL, JAMES)}
\end{align*}

answer tuple: \((LINDA)\)
FOL-reducibility of DL-Lite$_{core}$

Some subtleties

- When rewriting with existential restrictions, the variable that “is lost” should not occur anywhere else.

$$\exists y, z_1, z_2. \text{Woman}(x) \land \text{has\_child}(x, y) \land \text{has\_child}(z_1, y) \land \text{Human}(z_1) \land \text{has\_child}(z_2, z_1)$$

  $$\text{Human} \sqsubseteq \exists \text{has\_child}^{-1} \cdot \top$$

- To satisfy this constraint, one sometimes needs to unify atoms.

  $$\exists y, z_1. \text{has\_child}(x, y) \land \text{has\_child}(z_1, y)$$

  $$\text{Parent} \sqsubseteq \exists \text{has\_child} \cdot \top$$

Unification replaces $z_1$ by $x$:  

$$\exists y. \text{has\_child}(x, y)$$

$$\text{Parent}(x)$$
FOL-reducibility for the DL-Lite family of DLs

- DL-Lite$_{core}$ and its extensions DL-Lite$_R$ and DL-Lite$_F$ are FOL-reducible.

  additional role inclusion axioms:
  
  \[
  r_1 \sqsubseteq r_2 \\
  r_1 \sqsubseteq \neg r_2
  \]

  additional functionality axioms:
  
  \[
  \top \sqsubseteq (\leq 1 r) \\
  \top \sqsubseteq (\leq 1 r^{-1})
  \]

- FOL-reducibility implies a data complexity in \(AC^0\) for query answering, and thus in particular tractability w.r.t. data complexity.
FOL-reducibility for the DL-Lite family of DLs

- DL-Lite$_{core}$ and its extensions DL-Lite$_R$ and DL-Lite$_F$ are FOL-reducible.
  
  **additional role inclusion axioms:**
  
  \[ r_1 \sqsubseteq r_2 \]
  
  \[ r_1 \sqsubseteq \neg r_2 \]

  **additional functionality axioms:**
  
  \[ T \sqsubseteq (\leq 1 \ r) \]
  
  \[ T \sqsubseteq (\leq 1 \ r^{-1}) \]

- FOL-reducibility implies a data complexity in \( AC^0 \) for query answering, and thus in particular tractability w.r.t. data complexity.

- DL-Lite$_R$ is the formal basis for the OWL 2 QL profile of the new OWL 2 standard

- Approach implemented in the QUONTO system.
Query answering in $\mathcal{EL}$

- Computing certain answers w.r.t. $\mathcal{EL}$-TBoxes is tractable w.r.t. data complexity.

- More precisely, it is PTime-complete, and thus not in $AC^0$.

- Thus, query answering in $\mathcal{EL}$ is not FOL-reducible.

Can we still use RDB technology for query evaluation?
Query answering in $\mathcal{E}\mathcal{L}$ using relational DB technology beyond FOL-reducibility

Query \quad \Rightarrow \quad \text{Rewrite} \quad \Rightarrow \quad \text{FOL query}

GCIs \quad \Rightarrow \quad \text{Rewrite} \quad \Rightarrow \quad \text{Evaluate with RDB system} \quad \Rightarrow \quad \text{RDB}

ABox \quad \Rightarrow \quad \text{Rewrite} \quad \Rightarrow \quad \text{Evaluate with RDB system} \quad \Rightarrow \quad \text{RDB}

Answer tuples

works for both $\mathcal{E}\mathcal{L}$ and DL-Lite [Lutz et al.; 2009]
[Calvanese et al.; 2007]

- Query reformulation independent of ABox.
- ABox need not be changed.
- Size of reformulated query may grow exponentially.

[Lutz et al.; 2009]

- Query rewriting independent of ABox and GCIs.
- ABox needs to be changed.
- ABox rewriting independent of query.
- Both ABox and query rewriting polynomial.
Description Logics

Franz Baader
Theoretical Computer Science
TU Dresden
Germany

Literature:

