Query Inseparability for Description Logic Knowledge Bases

Elena Botoeva\textsuperscript{1} Roman Kontchakov\textsuperscript{2} Vladislav Ryzhikov\textsuperscript{1} \\
Frank Wolter\textsuperscript{3} Michael Zakharyaschev\textsuperscript{2}

\textsuperscript{1}Faculty of Computer Science, Free University of Bozen-Bolzano, Italy
\textsuperscript{2}Department of Computer Science, Birkbeck, University of London, UK
\textsuperscript{3}Department of Computer Science, University of Liverpool, UK

July 21 \\
KR 2014 \\
Vienna
Query Answering Over Knowledge Bases

<table>
<thead>
<tr>
<th>Category</th>
<th>Size</th>
<th>Color</th>
<th>Brand</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Choose size ▼</td>
<td>Choose color ▼</td>
<td>Choose brand ▼</td>
</tr>
</tbody>
</table>

**Sandals: (3 products found)**

- **heel_sand**
  - Heeled
  - €79
- **wedge_sand**
  - Platform
  - €69
- **brown_sand**
  - Classic
  - €50
Query Answering Over Knowledge Bases

Viewed as a knowledge base \((T, A)\) and a query \(q\):

\[ q(x) \leftarrow \text{Sandals}(x) \]
Query Answering Over Knowledge Bases

Viewed as a knowledge base \((T, A)\) and a query \(q\):

\[ q(x) \leftarrow \text{Sandals}(x), \text{hasSize}(x, 37), \text{hasBrand}(x, \text{geox}) \]
Motivation: Module Extraction

knowledge base $\mathcal{K}$

Give me all $B$ and $D$ such that ...
Motivation: Module Extraction

knowledge base $\mathcal{K}$

Give me all $B$ and $D$ such that . . .
Motivation: Module Extraction

knowledge base $\mathcal{K}$

module $\mathcal{K}'$

Give me all $B$ and $D$ such that . . .
Motivation: Module Extraction

knowledge base $\mathcal{K}$

Give me all $B$ and $D$ such that . . .

\[
\begin{align*}
b_1 & \quad d_1 \\
b_2 & \quad d_2 \\
\ldots
\end{align*}
\]
Motivation: Knowledge Exchange

I want a translation of $\mathcal{K}_1$ in $\Sigma_2$ to ask queries
Motivation: Knowledge Exchange

source schema $\Sigma_1$

mapping $M$

target schema $\Sigma_2$

source KB $K_1$

target KB $K_2$

Universal UCQ-solution

I want a translation of $K_1$ in $\Sigma_2$ to ask queries $K_2$
\[ \mathcal{K}_1 \text{-query entails } \mathcal{K}_2 \text{ if } \]
\[ \mathcal{K}_2 \models q(\bar{a}) \text{ implies } \mathcal{K}_1 \models q(\bar{a}), \]
for each CQ \( q(\bar{x}) \) over \( \Sigma \) and each tuple \( \bar{a} \subseteq \text{ind}(\mathcal{K}_2) \).
• $\mathcal{K}_1$ $\Sigma$-query entails $\mathcal{K}_2$ if
  
  $\mathcal{K}_2 \models q(\bar{a})$ implies $\mathcal{K}_1 \models q(\bar{a})$, 
  for each CQ $q(\bar{x})$ over $\Sigma$ and each tuple $\bar{a} \subseteq \text{ind}(\mathcal{K}_2)$.

• $\mathcal{K}_1$ and $\mathcal{K}_2$ are $\Sigma$-query inseparable, $\mathcal{K}_1 \equiv_{\Sigma} \mathcal{K}_2$, if
  
  $\mathcal{K}_1$ $\Sigma$-query entails $\mathcal{K}_2$ and $\mathcal{K}_2$ $\Sigma$-query entails $\mathcal{K}_1$
• \( \mathcal{K}_1 \) \( \Sigma \)-query entails \( \mathcal{K}_2 \) if
  \[ \mathcal{K}_2 \models q(\bar{a}) \text{ implies } \mathcal{K}_1 \models q(\bar{a}), \]
  for each \( \text{CQ } q(\bar{x}) \) over \( \Sigma \) and each tuple \( \bar{a} \subseteq \text{ind}(\mathcal{K}_2) \).

• \( \mathcal{K}_1 \) and \( \mathcal{K}_2 \) are \( \Sigma \)-query inseparable, \( \mathcal{K}_1 \equiv_{\Sigma} \mathcal{K}_2 \), if
  \[ \mathcal{K}_1 \text{ \( \Sigma \)-query entails } \mathcal{K}_2 \text{ \ and \ } \mathcal{K}_2 \text{ \( \Sigma \)-query entails } \mathcal{K}_1 \]

Then,

• \( \mathcal{K}' \subseteq \mathcal{K} \) is a \( \Sigma \)-module of \( \mathcal{K} \) if
  \[ \mathcal{K}' \equiv_{\Sigma} \mathcal{K}. \]
• $\mathcal{K}_1$ $\Sigma$-query entails $\mathcal{K}_2$ if
  $$\mathcal{K}_2 \models q(\vec{a}) \text{ implies } \mathcal{K}_1 \models q(\vec{a}),$$
  for each CQ $q(\vec{x})$ over $\Sigma$ and each tuple $\vec{a} \subseteq \text{ind}(\mathcal{K}_2)$.

• $\mathcal{K}_1$ and $\mathcal{K}_2$ are $\Sigma$-query inseparable, $\mathcal{K}_1 \equiv_\Sigma \mathcal{K}_2$, if
  $\mathcal{K}_1$ $\Sigma$-query entails $\mathcal{K}_2$ and $\mathcal{K}_2$ $\Sigma$-query entails $\mathcal{K}_1$

Then,

• $\mathcal{K}' \subseteq \mathcal{K}$ is a $\Sigma$-module of $\mathcal{K}$ if
  $$\mathcal{K}' \equiv_\Sigma \mathcal{K}.$$

• $\mathcal{K}_2$ is a universal UCQ-solution for $\mathcal{K}_1$ under $\mathcal{M}$ if
  $$\mathcal{K}_2 \equiv_{\Sigma_2} \mathcal{K}_1 \cup \mathcal{M}.$$
Horn Description Logics

Description Logics (DLs) represent knowledge in terms of **concepts** (unary predicates) and **roles** (binary predicates).

\[
\begin{align*}
\text{Horn-ALCHI} & \\
\text{Horn-ALCH} & \quad \text{Horn-ALCI} & \quad \text{DL-Lite}_\text{horn}^H \\
\text{OWL 2 EL} & \quad \text{Horn-ALC} & \quad \text{DL-Lite}_\text{horn}^H \\
\mathcal{ELH} & \quad P_1 \sqsubseteq P_2 & \quad \mathcal{EL}^{H} \\
A_1 \sqsubseteq \forall P.A_2, A \sqsubseteq \perp & \quad A, \exists P.A, \sqsubseteq & \quad A, \exists R, R := P \mid P^- \\
\end{align*}
\]
Horn Description Logics

Description Logics (DLs) represent knowledge in terms of concepts (unary predicates) and roles (binary predicates).

Data complexity of CQ-answering
How We Tackle \( \Sigma \)-Query Entailment

We rely on two fundamental instruments:

1. **Materialisation**, as an abstract way to characterize all answers to CQs over a KB.

   A materialisation of a KB \( \mathcal{K} \) is an interpretation \( \mathcal{M} \) such that
   \[
   \mathcal{K} \models q(\vec{a}) \iff \mathcal{M} \models q(\vec{a}),
   \]
   for each CQ \( q(\vec{x}) \) and each tuple \( \vec{a} \subseteq \text{ind}(\mathcal{K}) \).

2. **Reachability Games**, as a technique for obtaining upper-bounds.
Materialisations

Horn DLs enjoy **materialisations** (chase, canonical models).
Materialisations

Horn DLs enjoy **materialisations** (chase, canonical models).

Let $\mathcal{K} = \langle T, A \rangle$

$A = \{ B(a) \}$
Materialisations

Horn DLs enjoy **materialisations** (chase, canonical models).

Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$

$\mathcal{A} = \{ B(a) \}$

$\mathcal{T} = \{ B \sqsubseteq \exists P. \exists R. (\exists S \sqcap \exists Q) \}$

$\forall x. \ (B(x) \rightarrow \exists y, z, u, v. \ P(x, y), R(y, z), \ S(z, u), Q(z, v))$
Materialisations
Horn DLs enjoy **materialisations** (chase, canonical models).

Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$

$\mathcal{A} = \{ B(a) \}$

$\mathcal{T} = \{ B \sqsubseteq \exists P. \exists R. (\exists S \cap \exists Q) \}$

$\exists S^{-} \sqsubseteq \exists T. \exists S$

\[ \forall x. (B(x) \rightarrow \exists y, z, u, v. \ P(x, y), R(y, z), \ S(z, u), Q(z, v)) \]

\[ \forall x. (\exists u. \ S(u, x) \rightarrow \exists y, z. \ T(x, y), S(y, z)) \]
Materialisations

Horn DLs enjoy **materialisations** (chase, canonical models).

Let \( \mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle \)

\[ \mathcal{A} = \{ B(a) \} \]
\[ \mathcal{T} = \{ B \sqsubseteq \exists P. \exists R. (\exists S \cap \exists Q) \exists S^- \sqsubseteq \exists T. \exists S \} \]

\[ \forall x. (B(x) \rightarrow \exists y, z, u, v. P(x, y), R(y, z), S(z, u), Q(z, v)) \]
\[ \forall x. (\exists u. S(u, x) \rightarrow \exists y, z. T(x, y), S(y, z)) \]
Materialisations

Horn DLs enjoy **materialisations** (chase, canonical models).

Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$

$\mathcal{A} = \{ B(a) \}$

$\mathcal{T} = \{ B \sqsubseteq \exists P. \exists R. (\exists S \cap \exists Q) \}$

$\exists S^- \sqsubseteq \exists T. \exists S$

$\exists Q^- \sqsubseteq \exists Q$

\[
\forall x. (B(x) \rightarrow \exists y, z, u, v. \ P(x, y), R(y, z), \ S(z, u), Q(z, v))
\]

\[
\forall x. (\exists u. \ S(u, x) \rightarrow \exists y, z. \ T(x, y), S(y, z))
\]

\[
\forall x. (\exists y. \ Q(y, x) \rightarrow \exists z. \ Q(x, z))
\]
Materialisations

Horn DLs enjoy **materialisations** (chase, canonical models).

Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$

$\mathcal{A} = \{ B(a) \}$

$\mathcal{T} = \{ B \sqsubseteq \exists P. \exists R. (\exists S \sqcap \exists Q)$

$\exists S^{-} \sqsubseteq \exists T. \exists S$

$\exists Q^{-} \sqsubseteq \exists Q \}$

$\Sigma = \{ Q, R, S, T \}$

$\forall x. (B(x) \rightarrow \exists y, z, u, v.\ P(x, y), R(y, z), S(z, u), Q(z, v))$

$\forall x. (\exists u. S(u, x) \rightarrow \exists y, z. T(x, y), S(y, z))$

$\forall x. (\exists y. Q(y, x) \rightarrow \exists z. Q(x, z))$
Semantic Characterization of Σ-Query Entailment

Assume KBs $\mathcal{K}_1$ and $\mathcal{K}_2$ with materialisations $\mathcal{M}_1$ and $\mathcal{M}_2$.

**Theorem**

$\mathcal{K}_1$ Σ-query entails $\mathcal{K}_2$ iff $\mathcal{M}_2$ is finitely Σ-homomorphically embeddable into $\mathcal{M}_1$. 

Elena Botoeva (FUB) Query Inseparability for Description Logic Knowledge Bases 9/16
Homomorphisms as Games

The problem of finding a homomorphism can be seen as a game.

This game can be straightforwardly encoded as a Reachability Game (G, F).

However such encoding is impossible in practice:

1. Materialisations are infinite, in general;
2. Or of exponential size, even for DL-Lite core.
Homomorphisms as Games

The problem of finding a homomorphism can be seen as a game.
Homomorphisms as Games

The problem of finding a homomorphism can be seen as a game.
Homomorphisms as Games

The problem of finding a homomorphism can be seen as a game.

This game can be straightforwardly encoded as a Reachability Game $(G, F)$. However such encoding is impossible in practice:

1. Materialisations are infinite, in general;
2. Or of exponential size, even for DL-Lite core.
Homomorphisms as Games

The problem of finding a homomorphism can be seen as a game.

This game can be straightforwardly encoded as a Reachability Game $(G, F)$. However such encoding is impossible in practice:

1. Materialisations are infinite, in general;
2. Or of exponential size, even for DL-Lite core.
Homomorphisms as Games

The problem of finding a homomorphism can be seen as a game.

This game can be straightforwardly encoded as a Reachability Game \((G, F)\).

However such encoding is impossible in practice:
1. Materialisations are infinite, in general;
2. Or of exponential size, even for \(DL-Lite\) core.
Homomorphisms as Games

The problem of finding a homomorphism can be seen as a game.

This game can be straightforwardly encoded as a Reachability Game \((G, F)\).

However such encoding is impossible in practice:
1. Materialisations are infinite, in general;
2. Or of exponential size, even for DL-Lite core.

Elena Botoeva (FUB) Query Inseparability for Description Logic Knowledge Bases 10/16
Homomorphisms as Games

The problem of finding a homomorphism can be seen as a game.
Homomorphisms as Games

The problem of finding a homomorphism can be seen as a game.
Homomorphisms as Games

The problem of finding a homomorphism can be seen as a game.

This game can be straightforwardly encoded as a Reachability Game \((G, F)\).

However such encoding is impossible in practice:

1. Materialisations are infinite, in general;
2. Or of exponential size, even for DL-Lite core.
Homomorphisms as Games

The problem of finding a homomorphism can be seen as a game.
Homomorphisms as Games

The problem of finding a homomorphism can be seen as a game.
Homomorphisms as Games

The problem of finding a homomorphism can be seen as a game.
Homomorphisms as Games

The problem of finding a homomorphism can be seen as a game.

This game can be straightforwardly encoded as a **Reachability Game** \((G, F)\).
Homomorphisms as Games

The problem of finding a homomorphism can be seen as a game.

This game can be straightforwardly encoded as a Reachability Game \((G, F)\).
Homomorphisms as Games

The problem of finding a homomorphism can be seen as a game.

This game can be straightforwardly encoded as a Reachability Game \((G, F)\).

However such encoding is impossible in practice:

1. Materialisations are infinite, in general;
2. Or of exponential size, even for DL-Lite\textsubscript{core}.
Homomorphisms on Infinite Materialisations

$\mathcal{M}_2$ is finitely $\Sigma$-homomorphically embeddable into $\mathcal{M}_1$. 

How to play in this case? Instead of materialisations, we play on finite generating structures.
Homomorphisms on Infinite Materialisations

$\mathcal{M}_2$ is finitely $\Sigma$-homomorphically embeddable into $\mathcal{M}_1$. 

$\mathcal{M}_2^\Sigma$ 

$\mathcal{M}_1^\Sigma$

How to play in this case? Instead of materialisations, we play on finite generating structures.
Homomorphisms on Infinite Materialisations

$\mathcal{M}_2$ is finitely $\Sigma$-homomorphically embeddable into $\mathcal{M}_1$. 

How to play is this case? Instead of materialisations, we play on finite generating structures.
Homomorphisms on Infinite Materialisations

$\mathcal{M}_2$ is finitely $\Sigma$-homomorphically embeddable into $\mathcal{M}_1$.

How to play is this case?
Homomorphisms on Infinite Materialisations

$\mathcal{M}_2$ is finitely $\Sigma$-homomorphically embeddable into $\mathcal{M}_1$.

How to play is this case?

Instead of materialisations, we play on finite generating structures.
Finite Generating Structures

Materialisation $\mathcal{M}$

Generating structure $\mathcal{G}$ for $\mathcal{M}$
Finite Generating Structures

\[ \mathcal{M} \]

Materialisation \( \mathcal{M} \)

Generating structure \( \mathcal{G} \) for \( \mathcal{M} \)

\[ [\mathcal{EL}, \mathcal{ELH}], [\text{DL-Lite}_{\text{core}}, \text{DL-Lite}^\mathcal{H}], [\text{Horn-ALC}, \text{Horn-ALCHI}] : \text{generating structures of polynomial size} \]

\[ \text{[Horn-ALC, Horn-ALCHI]} : \text{generating structures exponential size} \]
The Upper Bound

For KBs $\mathcal{K}_1$, $\mathcal{K}_2$, and a signature $\Sigma$, we construct a reachability game $G_\Sigma(\mathcal{G}_2, \mathcal{G}_1) = (G, F)$ such that

**Player 1** has a winning strategy against **Player 2** in $G_\Sigma(\mathcal{G}_2, \mathcal{G}_1)$ iff

$\mathcal{M}_2$ is finitely $\Sigma$-homomorphically embeddable into $\mathcal{M}_1$.

where the size of $G$ is

- Polynomial in the size of $\mathcal{G}_2$ and $\mathcal{G}_1$, if the logic admits only **forward strategies** (without inverses)

- Polynomial in the size of $\mathcal{G}_2$ and $\mathcal{G}_1$, if the logic admits only **backward+forward strategies** ($DL$-$Lite$ without $\mathcal{H}$)

- Exponential in the size of $\mathcal{G}_2$, if the logic admits **arbitrary strategies**.
The Summary of the Results
The Summary of the Results

\[ I+(E \text{ or } H) \]

- arbitrary strategy

- forward strategy
  - no Inverses

- backward+forward strategy
  - no role Hierarchy

- Horn-ALCHI
- Horn-ALCH
- Horn-ALCI

- DL-Lite_{\text{horn}}^H
- DL-Lite_{\text{core}}^H

- \( E \Lambda H \)
- \( E \Lambda \)
- \( E \)
The Summary of the Results

- **Horn-ALCHI**
  - 2EXPTIME
  - arbitrary strategy

- **Horn-ALC**
  - EXPTIME

- **Horn-ALCH**
  - EXPTIME

- **ELH**
  - EXPTIME

- **EL**
  - PTIME

- **Horn-ALC**
  - PTIME

- **EL**
  - EXPTIME

- **DL-Lite^H_{horn}**

- **DL-Lite^H_{core}**

- **DL-Lite^H_{horn}**

- **DL-Lite^H_{core}**

- forward strategy
  - no Inverses

- backward + forward strategy
  - no role Hierarchy
Future Work

- approximate module extraction using forward strategies

- KB Inseparability for more expressive DLs: *Horn-SHIQ* and *ALC*

- TBox Inseparability
Thank you for your attention!