Finding Good Proofs for Ontology-Mediated Query Answers

(extended abstract of DL paper) // XLoKR’22, Haifa, Israel, 31st July 2022
Motivation

Finding Good Proofs for Ontology-Mediated Query Answers

Center for Perspicuous Computing (CPEC) // © Stefan Borgwardt
XLoKR'22, Haifa, Israel, 31st July 2022
Motivation

Ontologies, Proofs

plus Data, Queries
Motivation

Ontologies, Proofs

plus Data, Queries

Algorithms, Complexity
Outline

Preliminaries
  Description Logics
  Proofs
  Ontology-Mediated Queries

Proof Systems
  Deriving CQs
  Deriving Ground Atoms

Complexity

Summary
Description Logics (DLs)

DLs are fragments of first-order logic with a funny syntax.
Description Logics (DLs)

DLs are fragments of first-order logic with a funny syntax.

A DL TBox/ontology $\mathcal{T}$:

<table>
<thead>
<tr>
<th>Existential rules (implicit $\forall$ omitted)</th>
<th>DL syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{Spoke}(x) \rightarrow \exists y. \text{partOf}(x, y) \land \text{Wheel}(y)$</td>
<td>$\text{Spoke} \sqsubseteq \exists \text{partOf.Wheel}$</td>
</tr>
</tbody>
</table>
**Description Logics (DLs)**

DLs are fragments of first-order logic with a funny syntax.

A DL TBox/ontology $\mathcal{T}$:

<table>
<thead>
<tr>
<th>existential rules (implicit $\forall$ omitted)</th>
<th>DL syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{Spoke}(x) \rightarrow \exists y. \text{partOf}(x, y) \land \text{Wheel}(y)$</td>
<td>$\text{Spoke} \sqsubseteq \exists \text{partOf}.\text{Wheel}$</td>
</tr>
<tr>
<td>$\text{partOf}(x, y) \rightarrow \text{hasPart}(y, x)$</td>
<td>$\text{partOf} \sqsubseteq \text{hasPart}^{-}$</td>
</tr>
</tbody>
</table>
Description Logics (DLs)

DLs are fragments of first-order logic with a funny syntax.

A DL TBox/ontology $\mathcal{T}$:

<table>
<thead>
<tr>
<th>existential rules (implicit $\forall$ omitted)</th>
<th>DL syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{Spoke}(x) \rightarrow \exists y. \text{partOf}(x, y) \land \text{Wheel}(y)$</td>
<td>$\text{Spoke} \sqsubseteq \exists \text{partOf}. \text{Wheel}$</td>
</tr>
<tr>
<td>$\text{partOf}(x, y) \rightarrow \text{hasPart}(y, x)$</td>
<td>$\text{partOf} \sqsubseteq \text{hasPart}^-$</td>
</tr>
<tr>
<td>$\text{hasPart}(x, y) \rightarrow \exists z. \text{attachedTo}(y, z)$</td>
<td>$\exists \text{hasPart}^- \sqsubseteq \exists \text{attachedTo}$</td>
</tr>
</tbody>
</table>
Description Logics (DLs)

DLs are fragments of first-order logic with a funny syntax.

A DL TBox/ontology $\mathcal{T}$:

<table>
<thead>
<tr>
<th>Existential rules (implicit $\forall$ omitted)</th>
<th>DL syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spoke($x$) $\rightarrow$ $\exists y$. partOf($x$, $y$) $\land$ Wheel($y$)</td>
<td>Spoke $\sqsubseteq \exists$partOf.Wheel</td>
</tr>
<tr>
<td>partOf($x$, $y$) $\rightarrow$ hasPart($y$, $x$)</td>
<td>partOf $\sqsubseteq$ hasPart$^\neg$</td>
</tr>
<tr>
<td>hasPart($x$, $y$) $\rightarrow$ $\exists z$. attachedTo($y$, $z$)</td>
<td>$\exists$hasPart$^\neg$ $\sqsubseteq \exists$attachedTo</td>
</tr>
<tr>
<td>Wheel($x$) $\rightarrow$ $\exists y$. partOf($x$, $y$) $\land$ Bike($y$)</td>
<td>Wheel $\sqsubseteq \exists$partOf.Bike</td>
</tr>
</tbody>
</table>
Description Logics (DLs) \( DL-Lite^R \)

DLs are fragments of first-order logic with a funny syntax.

A \( DL \) \( DL-Lite^R \) TBox/ontology \( T \):

existential rules (implicit \( \forall \) omitted)

\[
\begin{align*}
\text{Spoke}(x) & \rightarrow \exists y. \text{partOfW}(x, y) \\
\text{partOf}(x, y) & \rightarrow \text{hasPart}(y, x) \\
\text{hasPart}(x, y) & \rightarrow \exists z. \text{attachedTo}(y, z) \\
\text{Wheel}(x) & \rightarrow \exists y. \text{partOfB}(x, y) \\
\text{partOfW}(x, y) & \rightarrow \text{partOf}(x, y) \\
\text{partOfW}(x, y) & \rightarrow \text{Wheel}(y) \\
\text{partOfB}(x, y) & \rightarrow \text{partOf}(x, y) \\
\text{partOfB}(x, y) & \rightarrow \text{Bike}(y)
\end{align*}
\]
Proofs

(Alrabbaa, Baader, Borgwardt, Koopmann, and Kovtunova 2021)

**Entailment**: $\mathcal{T}$ implies a new sentence $\alpha$, e.g.,

\[
\text{partOfW}(x, y) \rightarrow \exists z. \text{partOfB}(y, z)
\]

\[
\exists \text{partOfW} \sqsubseteq \exists \text{partOfB}
\]

\[
\mathcal{T}
\]

\[
\top
\]

\[
\alpha
\]
Proofs

(Alrabbaa, Baader, Borgwardt, Koopmann, and Kovtunova 2021)

Entailment: $\mathcal{T}$ implies a new sentence $\alpha$, e.g.,
\[
\text{partOfW}(x, y) \rightarrow \exists z. \text{partOfB}(y, z)
\]

Justification: Minimal subset of $\mathcal{T}$ entailing $\alpha$
Proofs

(Alrabbaa, Baader, Borgwardt, Koopmann, and Kovtunova 2021)

**Entailment:** $\mathcal{T}$ implies a new sentence $\alpha$, e.g.,
$$\text{partOfW}(x, y) \rightarrow \exists z. \text{partOfB}(y, z)$$

**Justification:** Minimal subset of $\mathcal{T}$ entailing $\alpha$

**Inference step:** List of premises + conclusion
Proofs (Alrabbaa, Baader, Borgwardt, Koopmann, and Kovtunova 2021)

Entailment: $T$ implies a new sentence $\alpha$, e.g.,
\[ \text{partOfW}(x, y) \rightarrow \exists z. \text{partOfB}(y, z) \]

Justification: Minimal subset of $T$ entailing $\alpha$

Inference step: List of premises + conclusion

Proof: Acyclic, connected, non-redundant hypergraph with sink $\alpha$
Proofs

(Alrabbaa, Baader, Borgwardt, Koopmann, and Kovtunova 2021)

**Entailment:** $\mathcal{T}$ implies a new sentence $\alpha$, e.g.,

$$\text{partOfW}(x, y) \rightarrow \exists z. \text{partOfB}(y, z) \quad \exists \text{partOfW} \subseteq \exists \text{partOfB}$$

**Justification:** Minimal subset of $\mathcal{T}$ entailing $\alpha$

**Inference step:** List of premises + conclusion

**Proof:** Acyclic, connected, non-redundant hypergraph with sink $\alpha$

**Deriver:** Provides hypergraphs $\mathcal{D}(\mathcal{T}, \alpha)$ of all permissible inference steps for $\mathcal{T} \models \alpha$
Proofs (Alrabbaa, Baader, Borgwardt, Koopmann, and Kovtunova 2021)

Entailment: $\mathcal{T}$ implies a new sentence $\alpha$, e.g.,
$$\text{partOfW}(x, y) \rightarrow \exists z \cdot \text{partOfB}(y, z)$$
$\exists \text{partOfW} \not\subseteq \exists \text{partOfB}$

Justification: Minimal subset of $\mathcal{T}$ entailing $\alpha$

Inference step: List of premises + conclusion

Proof: Acyclic, connected, non-redundant hypergraph with sink $\alpha$

Deriver: Provides hypergraphs $\mathcal{D}(\mathcal{T}, \alpha)$ of all permissible inference steps for $\mathcal{T} \models \alpha$

Goal: Find optimal proofs in $\mathcal{D}(\mathcal{T}, \alpha)$
Proofs

(Alrabbaa, Baader, Borgwardt, Koopmann, and Kovtunova 2021)

**Entailment:** $T$ implies a new sentence $\alpha$, e.g.,

\[
\text{partOfW}(x, y) \rightarrow \exists z. \text{partOfB}(y, z)
\]

**Justification:** Minimal subset of $T$ entailing $\alpha$

**Inference step:** List of premises + conclusion

**Proof:** Acyclic, connected, non-redundant hypergraph with sink $\alpha$

**Deriver:** Provides hypergraphs $D(T, \alpha)$ of all permissible inference steps for $T \models \alpha$

**Goal:** Find optimal proofs in $D(T, \alpha)$

**Measures:** Size (5)
Proofs

(Alrabbaa, Baader, Borgwardt, Koopmann, and Kovtunova 2021)

Entailment: $\mathcal{T}$ implies a new sentence $\alpha$, e.g.,
$$\text{partOfW}(x, y) \rightarrow \exists z. \text{partOfB}(y, z) \quad \exists \text{partOfW} \subseteq \exists \text{partOfB}$$

Justification: Minimal subset of $\mathcal{T}$ entailing $\alpha$

Inference step: List of premises + conclusion

Proof: Acyclic, connected, non-redundant hypergraph with sink $\alpha$

Deriver: Provides hypergraphs $\mathcal{D}(\mathcal{T}, \alpha)$ of all permissible inference steps for $\mathcal{T} \models \alpha$

Goal: Find optimal proofs in $\mathcal{D}(\mathcal{T}, \alpha)$

Measures: Size (5), tree size (6)
Proofs

(Alrabbaa, Baader, Borgwardt, Koopmann, and Kovtunova 2021)

**Entailment:** \( T \) implies a new sentence \( \alpha \), e.g.,
\[
\text{partOfW}(x, y) \rightarrow \exists z. \text{partOfB}(y, z)
\]

**Justification:** Minimal subset of \( T \) entailing \( \alpha \)

**Inference step:** List of premises + conclusion

**Proof:** Acyclic, connected, non-redundant hypergraph with sink \( \alpha \)

**Deriver:** Provides hypergraphs \( \mathcal{D}(T, \alpha) \) of all permissible inference steps for \( T \models \alpha \)

**Goal:** Find optimal proofs in \( \mathcal{D}(T, \alpha) \)

**Measures:** Size (5), tree size (6)

**Decision problem:** Decide whether there exists a proof of measure \( \leq n \)
Ontology-Mediated Queries

Ontology-mediated queries generalize database queries
Ontology-Mediated Queries

Ontology-mediated queries generalize database queries

ABox $A$: Spoke($a$), partOf($a, b$), …
Ontology-Mediated Queries

Ontology-mediated queries generalize database queries

ABox $A$: Spoke($a$), partOf($a$, $b$), ...

Conjunctive query $q(x)$: $\exists y, z, v, w, w'.
\text{partOfW}(x, y) \land \text{partOfB}(y, z) \land \text{hasPart}(y, v) \land \text{attachedTo}(v, w) \land \text{attachedTo}(v, w')$
Ontology-Mediated Queries

Ontology-mediated queries generalize database queries

ABox $\mathcal{A}$: Spoke($a$), partOf($a$, $b$), ...

Conjunctive query $q(x)$: $\exists y, z, v, w, w'$. 
partOfW($x$, $y$) \land partOfB($y$, $z$) \land hasPart($y$, $v$) \land attachedTo($v$, $w$) \land attachedTo($v$, $w'$)

Ontology-mediated query answering (OMQA): $\mathcal{T} \cup \mathcal{A} \models q(a)$
Ontology-Mediated Queries

Ontology-mediated queries generalize database queries

\[ \text{ABox } \mathcal{A}: \text{Spoke}(a), \text{partOf}(a, b), \ldots \]

Conjunctive query \( q(x) \): \[ \exists y, z, v, w, w'. \]
\[ \text{partOfW}(x, y) \land \text{partOfB}(y, z) \land \text{hasPart}(y, v) \land \text{attachedTo}(v, w) \land \text{attachedTo}(v, w') \]

Ontology-mediated query answering (OMQA): \( \mathcal{T} \cup \mathcal{A} \models q(a) \)

For \( DL-Lite_R \), this problem is

- in \( AC^0 \) in data complexity (\( \mathcal{T} \) and \( q(a) \) are fixed)
- \textbf{NP-complete} in combined complexity
- \textbf{P-complete} for tree-shaped queries
Ontology-Mediated Queries

Ontology-mediated queries generalize database queries

**ABox** $\mathcal{A}$: Spoke($a$), partOf($a, b$), ...

Conjunctive query $q(x)$: $\exists y, z, v, w, w'$. 

\[
\text{partOfW}(x, y) \land \text{partOfB}(y, z) \land \text{hasPart}(y, v) \land \text{attachedTo}(v, w) \land \text{attachedTo}(v, w')
\]

Ontology-mediated query answering (OMQA): $\mathcal{T} \cup \mathcal{A} \models q(a)$

For $DL-Lite_R$, this problem is

- in $AC^0$ in data complexity ($\mathcal{T}$ and $q(a)$ are fixed)
- NP-complete in combined complexity
- P-complete for tree-shaped queries

What is the complexity of explaining this entailment?
Outline

Preliminaries
  Description Logics
  Proofs
  Ontology-Mediated Queries

Proof Systems
  Deriving CQs
  Deriving Ground Atoms

Complexity

Summary
Deriving CQs ($\mathcal{Q}_{cq}$) (Stefanoni 2011; Croce and Lenzerini 2018)

What are the inference rules for proving $\mathcal{T} \cup \mathcal{A} \models q(a)$ (in DL-Lite$_R$)?
Deriving CQs ($\mathcal{D}_{cq}$) (Stefanoni 2011; Croce and Lenzerini 2018)

What are the inference rules for proving $\mathcal{T} \cup \mathcal{A} \models q(a)$ (in DL-Lite$_R$)?

- **Spoke($a$)**
  - $\exists y. \text{partOfW}(x, y)$
  - $\exists y. \text{partOfW}(a, y)$

- **Spoke($x$)**
  - $\exists y. \text{partOfW}(x, y)$
Deriving CQs (Ωcq)

What are the inference rules for proving $\mathcal{T} \cup \mathcal{A} \models q(a)$ (in DL-Lite$R$)?

- **Spoke($a$)**
  - $\exists y. \text{partOfW}(a, y)$

- $\exists y. \text{partOfW}(x, y)$
  - $\text{partOfW}(x, y) \rightarrow \exists z. \text{partofB}(y, z)$
  - $\exists y, z. \text{partOfW}(a, y) \land \text{partOfB}(y, z)$
Deriving CQs (Ω_{cq})

What are the inference rules for proving $\mathcal{T} \cup \mathcal{A} \models q(a)$ (in $DL-Lite_R$)?

[[Diagram of inference rules]]

Finding Good Proofs for Ontology-Mediated Query Answers
Center for Perspicuous Computing (CPEC) // © Stefan Borgwardt
XLoKR'22, Haifa, Israel, 31st July 2022
Deriving CQs ($\mathcal{D}_{cq}$) (Stefanoni 2011; Croce and Lenzerini 2018)

What are the inference rules for proving $T \cup A \models q(a)$ (in DL-Lite$_R$)?

- Spoke($a$) → Spoke($x$) → $\exists y. \text{partOfW}(x, y)$
- $\exists y. \text{partOfW}(a, y)$ → $\text{partOfW}(x, y) \rightarrow \exists z. \text{partofB}(y, z)$
- $\text{partOfW}(x, y) \rightarrow \text{hasPart}(y, x)$ → $\exists y, z. \text{partOfW}(a, y) \land \text{partOfB}(y, z)$
- $\exists y, z. \text{pW}(a, y) \land \text{pB}(y, z) \land \text{hP}(y, a)$ → $\text{hP}(x, y) \rightarrow \exists z. \text{aT}(y, z)$
- $\exists y, z, w. \text{pW}(a, y) \land \text{pB}(y, z) \land \text{hP}(y, a) \land \text{aT}(a, w)$
Deriving CQs ($\mathcal{O}_{cq}$)

(Stefanoni 2011; Croce and Lenzerini 2018)

What are the inference rules for proving $\mathcal{T} \cup \mathcal{A} \models q(a)$ (in DL-Lite$_R$)?

- Spoke($a$)
- Spoke($x$) $\rightarrow$ $\exists y. \text{partOfW}(x, y)$
- $\exists y. \text{partOfW}(a, y)$
- $\text{partOfW}(x, y) \rightarrow \exists z. \text{partOfB}(y, z)$
- $\text{partOfW}(x, y) \rightarrow \exists z. \text{partOfB}(y, z)$
- $\exists y. \text{partOfW}(a, y) \land \text{partOfB}(y, z)$
- $\exists y, z. \text{partOfW}(a, y) \land \text{partOfB}(y, z)$
- $\exists y, z. \text{partOfW}(a, y) \land \text{partOfB}(y, z) \land \text{hP}(y, a)$
- $\exists y, z. \text{partOfW}(a, y) \land \text{partOfB}(y, z) \land \text{hP}(y, a) \land \text{aT}(a, w)$
- $\exists y, z, w, w'. \text{pW}(a, y) \land \text{pB}(y, z) \land \text{hP}(y, a) \land \text{aT}(a, w) \land \text{aT}(a, w')$
Deriving CQs ($\mathcal{O}_{cq}$)

What are the inference rules for proving $\mathcal{T} \cup \mathcal{A} \models q(a)$ (in DL-Lite$_R$)?

- Spoke($a$)
- Spoke($x$) → $\exists y$. partOfW($x, y$)
- $\exists y$. partOfW($a, y$)
- partOfW($x, y$) → $\exists z$. partOfB($y, z$)
- partOfW($x, y$) → hasPart($y, x$)
- $\exists y, z$. partOfW($a, y$) ∧ partOfB($y, z$)
- $\exists y, z, pW(a, y) \land pB(y, z) \land hP(y, a)$
- hP($x, y$) → $\exists z$. aT($y, z$)
- aT($x, y$) → $\exists y$. aT($x, y$)
- $\exists y, z, w$. pW($a, y$) ∧ pB($y, z$) ∧ hP($y, a$) ∧ aT($a, w$)
- $\exists y, z, w, w'$. pW($a, y$) ∧ pB($y, z$) ∧ hP($y, a$) ∧ aT($a, w$) ∧ aT($a, w'$)
- $\exists y, z, v, w, w'$. pW($a, y$) ∧ pB($y, z$) ∧ hP($y, v$) ∧ aT($v, w$) ∧ aT($v, w'$)
Deriving Ground Atoms ($\mathcal{D}_{sk}$) (Borgida, Calvanese, and Rodriguez-Muro 2008)

We can use Skolem functions to make proofs more modular and compact.
We can use Skolem functions to make proofs more modular and compact.

\( \text{Spoke}(a) \)  \( \rightarrow \text{partOfW}(a, \text{wheel}(a)) \)
We can use Skolem functions to make proofs more modular and compact.

\[ \text{Spoke}(a) \quad \text{Spoke}(x) \rightarrow \text{partOfW}(x, \text{wheel}(x)) \]
\[ \text{partOfW}(a, \text{wheel}(a)) \quad \text{pW}(x, y) \rightarrow \text{pB}(y, \text{bi}(y)) \]
\[ \text{partOfB}(\text{wheel}(a), \text{bike}(\text{wheel}(a))) \]
Deriving Ground Atoms ($\mathcal{D}_{sk}$) (Borgida, Calvanese, and Rodriguez-Muro 2008)

We can use Skolem functions to make proofs more modular and compact.

$$\text{Spoke}(a) \quad \text{Spoke}(x) \rightarrow \text{partOfW}(x, \text{wheel}(x))$$

$$\text{pW}(x, y) \rightarrow \text{hP}(y, x)$$

$$\text{partOfW}(a, \text{wheel}(a))$$

$$\text{hasPart}(\text{wheel}(a), a)$$

$$\text{pW}(x, y) \rightarrow \text{pB}(y, \text{bi}(y))$$

$$\text{partOfB}(\text{wheel}(a), \text{bike}(\text{wheel}(a)))$$
Deriving Ground Atoms ($\mathcal{D}_{sk}$) (Borgida, Calvanese, and Rodriguez-Muro 2008)

We can use **Skolem functions** to make proofs more modular and compact.

![Diagram](attachment:image.png)
Deriving Ground Atoms ($\mathcal{D}_{sk}$) (Borgida, Calvanese, and Rodriguez-Muro 2008)

We can use Skolem functions to make proofs more modular and compact.

We can use Skolem functions to make proofs more modular and compact.

```
\begin{align*}
\text{Spoke}(a) & \quad \text{Spoke}(x) \rightarrow \text{partOfW}(x, \text{wheel}(x)) \\
\text{pW}(x, y) \rightarrow \text{hP}(y, x) & \quad \text{partOfW}(a, \text{wheel}(a)) \\
\text{hP}(x, y) \rightarrow \text{aT}(y, \text{at}(y)) & \quad \text{hasPart}(\text{wheel}(a), a) \\
\text{attachedTo}(a, \text{attachment}(a)) &
\end{align*}
```

```
\begin{align*}
\{ \text{pW}(a, \text{wh}(a)) \land \text{pB}(\text{wh}(a), \text{bi}(\text{wh}(a))) \land \text{hP}(\text{wh}(a), a) \land \text{aT}(a, \text{at}(a)) \land \text{aT}(a, \text{at}(a)) \}
\end{align*}
```
Deriving Ground Atoms ($\mathcal{D}_{sk}$) (Borgida, Calvanese, and Rodriguez-Muro 2008)

We can use Skolem functions to make proofs more modular and compact.

\[
\exists y, z, v, w, w'. pW(a, y) \land pB(y, z) \land hP(y, v) \land aT(v, w) \land aT(v, w')
\]
We can use **Skolem functions** to make proofs more modular and compact.

\[
\exists y, z, v, w, w'. pW(a, y) \land pB(y, z) \land hP(y, v) \land aT(v, w) \land aT(v, w')
\]
Outline

Preliminaries
  Description Logics
  Proofs
  Ontology-Mediated Queries

Proof Systems
  Deriving CQs
  Deriving Ground Atoms

Complexity

Summary
The Complexity of Finding Small (Tree) Proofs

Assuming that $\mathcal{T} \cup \mathcal{A} \models q(a)$ holds, is there a proof of (tree) size $\leq n$ w.r.t. $\mathcal{D}_{cq}/\mathcal{D}_{sk}$?

First result:
- Proofs can be translated between $\mathcal{D}_{cq}$ and $\mathcal{D}_{sk}$ in polynomial time.

Data complexity ($\mathcal{T}$ and $q(a)$ are fixed):
- In $\text{AC}^0$ for $\text{DL-Lite}R$ (exploit query rewritability).

Combined complexity:
- NP-complete for $\text{DL-Lite}R$ (tree size can be bounded by a polynomial).
- In P for tree-shaped queries in $\text{DL-Lite}R$ w.r.t. tree size and $\mathcal{D}_{sk}$ (use placeholders for Skolem terms).
- NP-hard for tree-shaped queries w.r.t. size or $\mathcal{D}_{cq}$ and empty TBox (reductions from SAT).
The Complexity of Finding Small (Tree) Proofs

Assuming that $\mathcal{T} \cup \mathcal{A} \models q(a)$ holds, is there a proof of (tree) size $\leq n$ w.r.t. $\mathcal{D}_{cq}/\mathcal{D}_{sk}$?

First result:

- Proofs can be translated between $\mathcal{D}_{cq}$ and $\mathcal{D}_{sk}$ in polynomial time

Finding Good Proofs for Ontology-Mediated Query Answers
Center for Perspicuous Computing (CPEC) // © Stefan Borgwardt
XLoKR'22, Haifa, Israel, 31st July 2022
The Complexity of Finding Small (Tree) Proofs

Assuming that $\mathcal{T} \cup \mathcal{A} \models q(a)$ holds, is there a proof of (tree) size $\leq n$ w.r.t. $\mathcal{D}_{cq}/\mathcal{D}_{sk}$?

First result:
- Proofs can be translated between $\mathcal{D}_{cq}$ and $\mathcal{D}_{sk}$ in polynomial time

Data complexity ($\mathcal{T}$ and $q(a)$ are fixed):
- In $\text{AC}^0$ for $\text{DL-Lite}_R$ (exploit query rewritability)
The Complexity of Finding Small (Tree) Proofs

Assuming that $\mathcal{T} \cup \mathcal{A} \models q(a)$ holds, is there a proof of (tree) size $\leq n$ w.r.t. $\mathcal{D}_{cq}/\mathcal{D}_{sk}$?

First result:

- Proofs can be translated between $\mathcal{D}_{cq}$ and $\mathcal{D}_{sk}$ in polynomial time

Data complexity ($\mathcal{T}$ and $q(a)$ are fixed):

- In $\text{AC}^0$ for $DL$-$\text{Lite}_R$ (exploit query rewritability)

Combined complexity:

- NP-complete for $DL$-$\text{Lite}_R$ (tree size can be bounded by a polynomial)
- In P for tree-shaped queries in $DL$-$\text{Lite}_R$ w.r.t. tree size and $\mathcal{D}_{sk}$ (use placeholders for Skolem terms)
- NP-hard for tree-shaped queries w.r.t. size or $\mathcal{D}_{cq}$ and empty TBox (reductions from SAT)
Summary

- Framework for describing proofs of query answers
- Two different kinds of inference rules
- Complexity results for $DL$-$Lite_R$

Ongoing/future work:
- Extend to other Horn DLs
- Combine with TBox reasoning
- Interactive presentation for user studies
- Find out which deriver is more useful in practice
Summary

- Framework for describing proofs of query answers
- Two different kinds of inference rules
- Complexity results for DL-Lite$_R$

Ongoing/future work:

- Extend to other Horn DLs
- Combine with TBox reasoning
- Interactive presentation for user studies
- Find out which deriver is more useful in practice
Summary

- Framework for describing proofs of query answers
- Two different kinds of inference rules
- Complexity results for $DL$-$Lite_R$

Ongoing/future work:

- Extend to other Horn DLs
- Combine with TBox reasoning
- Interactive presentation for user studies
- Find out which deriver is more useful in practice

Thank you!
References


