Data Repair of Inconsistent DL-Programs

Thomas Eiter    Michael Fink    Daria Stepanova

Knowledge-Based Systems Group,
Institute of Information Systems,
Vienna University of Technology
http://www.kr.tuwien.ac.at/

IJCAI 2013 –August 6, 2013
Motivation

- **DL-program**: ontology + rules (loose coupling combination approach);
- DL-atoms serve as query interfaces to ontology;
- Possibility to add information from the rule part to ontology prior to querying it allows for bidirectional information flow.

However, information exchange between rules and ontology can have unforeseen effects and cause **inconsistency** of the DL-program (absence of answer sets).
Motivation

• DL-program: ontology + rules (loose coupling combination approach);
• DL-atoms serve as query interfaces to ontology;
• Possibility to add information from the rule part to ontology prior to querying it allows for bidirectional information flow.

However, information exchange between rules and ontology can have unforeseen effects and cause inconsistency of the DL-program (absence of answer sets).

In this work: Repair data part of the ontology ($DL$-Lite$_A$), i.e. change ontology ABox s.t. the resulting DL-program is consistent.
Overview

Motivation

DL-programs

Repair answer sets

Computation

Conclusion
**DL-Lite\(_A\)**

- Lightweight Description Logic for accessing large data sources.
- Concepts and roles model sets of objects and their relationships.

\[
C \rightarrow A \mid \exists R \quad R \rightarrow P \mid P^-
\]

- A *DL-Lite\(_A\)* ontology \(\mathcal{O} = \langle \mathcal{T}, \mathcal{A} \rangle\) consists of:
  - **TBox** \(\mathcal{T}\) specifying constraints at the conceptual level.
    \[
    C_1 \sqsubseteq C_2, \quad C_1 \sqsubseteq \neg C_2, \\
    R_1 \sqsubseteq R_2, \quad R_1 \sqsubseteq \neg R_2, \quad (\text{funct } R).
    \]
  - **ABox** \(\mathcal{A}\) specifying the facts that hold in the domain.
    \[
    A(b) \quad P(a, b)
    \]
**DL-Lite_\mathcal{A}**

- Lightweight Description Logic for accessing large data sources.
- Concepts and roles model sets of objects and their relationships.

\[ C \rightarrow A \ | \ \exists R \quad R \rightarrow P | P^- \]

- A *DL-Lite_\mathcal{A}* ontology \( \mathcal{O} = \langle \mathcal{T}, \mathcal{A} \rangle \) consists of:
  - **TBox** \( \mathcal{T} \) specifying constraints at the conceptual level.
    \[ C_1 \sqsubseteq C_2, \quad C_1 \sqsubseteq \neg C_2, \]
    \[ R_1 \sqsubseteq R_2, \quad R_1 \sqsubseteq \neg R_2, \quad (\text{funct } R). \]
  - **ABox** \( \mathcal{A} \) specifying the facts that hold in the domain.
    \[ A(b) \quad P(a,b) \]

**Example**

\( \mathcal{T} = \left\{ \begin{array}{l}
\text{Child} \sqsubseteq \exists \text{hasParent} \\
\text{Female} \sqsubseteq \neg \text{Male}
\end{array} \right\} \quad \mathcal{A} = \left\{ \begin{array}{l}
\text{hasParent}(\text{john}, \text{pat}) \\
\text{Male}(\text{john})
\end{array} \right\} \)
**DL-Lite**

- Lightweight Description Logic for accessing large data sources.
- Concepts and roles model sets of objects and their relationships.

\[ C \rightarrow A \mid \exists R \quad R \rightarrow P | P^- \]

- A **DL-Lite** ontology \( \mathcal{O} = \langle \mathcal{T}, A \rangle \) consists of:
  - **TBox** \( \mathcal{T} \) specifying constraints at the conceptual level.
    \[ C_1 \sqsubseteq C_2, \quad C_1 \sqsubseteq \neg C_2, \]
    \[ R_1 \sqsubseteq R_2, \quad R_1 \sqsubseteq \neg R_2, \quad (\text{funct } R). \]
  - **ABox** \( A \) specifying the facts that hold in the domain.
    \[ A(b) \quad P(a, b) \]

**Example**

\[ \mathcal{T} = \left\{ \begin{array}{l} \text{Child} \sqsubseteq \exists \text{hasParent} \\ \text{Female} \sqsubseteq \neg \text{Male} \end{array} \right\} \]

\[ A = \left\{ \begin{array}{l} \text{hasParent}(john, pat) \\ \text{Male}(john) \end{array} \right\} \]

Conjunctive query answering in **DL-Lite** is tractable [Calvanese et al., 2007].
Example: DL-program

\[ \Pi = \langle O, P \rangle \text{ is a DL-program.} \]

\[ O = \left\{ \begin{array}{l}
(1) \text{Child} \sqsubseteq \exists \text{hasParent} \\
(2) \text{Adopted} \sqsubseteq \text{Child} \\
(3) \text{Female} \sqsubseteq \neg \text{Male} \\
(4) \text{Male}(\text{pat}) \\
(5) \text{Male}(\text{john}) \\
(6) \text{hasParent}(\text{john}, \text{pat})
\end{array} \right\} \]
Example: DL-program

\[ \Pi = \langle O, P \rangle \] is a DL-program.

\[ O = \begin{cases} 
(1) \text{Child} \sqsubseteq \exists \text{hasParent} \\
(2) \text{Adopted} \sqsubseteq \text{Child} \\
(3) \text{Female} \sqsubseteq \neg \text{Male} \\
(4) \text{Male}(\text{pat}) \\
(5) \text{Male}(\text{john}) \\
(6) \text{hasParent}(\text{john}, \text{pat}) 
\end{cases} \]

\[ P = \begin{cases} 
(7) \text{ischildof}(\text{john}, \text{alex}); \\
(8) \text{boy}(\text{john}); \\
(9) \text{hasfather}(\text{john}, \text{pat}) \leftarrow \text{DL}[\text{Male} \cup \text{boy}; \text{Male}](\text{pat}), \\
\text{DL}[; \text{hasParent}](\text{john}, \text{pat}) 
\end{cases} \]

- interpretation: \( I = \{ \text{ischildof}(\text{john}, \text{alex}), \text{boy}(\text{john}), \text{hasfather}(\text{john}, \text{pat}) \} \);
- satisfaction relation: \( I \models^O \text{boy}(\text{john}); I \models^O \text{DL}[; \text{hasParent}](\text{john}, \text{pat}) \);
- semantics is given in terms of answer sets, which are \( x \)-founded models;
- \( flp \) and \( weak \) semantics are relevant in this work;
- \( I \) is both \( weak \)- and \( flp \)-founded model.
Example: Inconsistent DL-program

\[ \Pi = \langle \mathcal{O}, P \rangle \]

\[ \mathcal{O} = \{ \]
\[ (1) \text{Child} \sqsubseteq \exists \text{hasParent} \]
\[ (2) \text{Adopted} \sqsubseteq \text{Child} \]
\[ (3) \text{Female} \sqsubseteq \neg \text{Male} \]
\[ \} \]

\[ P = \{ \]
\[ (7) \text{ischildof}(\text{john}, \text{alex}); \]
\[ (8) \text{boy}(\text{john}); \]
\[ (9) \text{hasfather}(\text{john}, \text{pat}) \leftarrow \text{DL}[; \text{Male} \cup \text{boy}; \text{Male}](\text{pat}), \]
\[ \text{DL}[; \text{hasParent}](\text{john}, \text{pat}); \]
\[ (10) \bot \leftarrow \text{not DL}[; \text{Adopted}](\text{john}), \text{pat} \neq \text{alex}, \]
\[ \text{hasfather}(\text{john}, \text{pat}), \text{ischildof}(\text{john}, \text{alex}), \]
\[ \text{not DL}[\text{Child} \cup \text{boy}; \neg \text{Male}](\text{alex}) \]
\[ \} \]
Example: Inconsistent DL-program

\[ \Pi = \langle \mathcal{O}, P \rangle \]

\[ \mathcal{O} = \begin{cases} 
(1) \text{Child} \sqsubseteq \exists \text{hasParent} & (4) \text{Male}(\text{pat}) \\
(2) \text{Adopted} \sqsubseteq \text{Child} & (5) \text{Male}(\text{john}) \\
(3) \text{Female} \sqsubseteq \neg \text{Male} & (6) \text{hasParent}(\text{john}, \text{pat}) 
\end{cases} \]

\[ P = \begin{cases} 
(7) \text{ischildof}(\text{john}, \text{alex}); & (8) \text{boy}(\text{john}); \\
(9) \text{hasfather}(\text{john}, \text{pat}) \leftarrow \text{DL}[; \text{Male} \uplus \text{boy}; \text{Male}](\text{pat}), \text{DL}[; \text{hasParent}](\text{john}, \text{pat}); \\
(10) \bot \leftarrow \neg \text{DL}[; \text{Adopted}](\text{john}), \text{pat} \neq \text{alex}, \text{hasfather}(\text{john}, \text{pat}), \text{ischildof}(\text{john}, \text{alex}), \neg \text{DL}[\text{Child} \uplus \text{boy}; \neg \text{Male}](\text{alex}) 
\end{cases} \]
Example: Inconsistent DL-program

\[ \Pi = \langle \mathcal{O}, P \rangle \]

\[ \mathcal{O} = \begin{cases} 
(1) & \text{Child} \sqsubseteq \exists \text{hasParent} \\
(2) & \text{Adopted} \sqsubseteq \text{Child} \\
(3) & \text{Female} \sqsubseteq \neg \text{Male} 
\end{cases} \]

\[ P = \begin{cases} 
(7) & \text{ischildof}(\text{john}, \text{alex}) ; \\
(8) & \text{boy}(\text{john}) ; \\
(9) & \text{hasfather}(\text{john}, \text{pat}) \leftarrow DL[; \text{Male} \oplus \text{boy} ; \text{Male}](\text{pat}), \\
& DL[; \text{hasParent}](\text{john}, \text{pat}) ; \\
(10) & \perp \leftarrow \text{not DL}[; \text{Adopted}](\text{john}), \text{pat} \neq \text{alex}, \\
& \text{hasfather}(\text{john}, \text{pat}), \text{ischildof}(\text{john}, \text{alex}), \\
& \text{not DL}[\text{Child} \oplus \text{boy} ; \neg \text{Male}](\text{alex}) 
\end{cases} \]
Example: Inconsistent DL-program

\[ \Pi = \langle O, P \rangle \text{ is inconsistent!} \]

\[ O = \begin{cases} 
(1) \text{Child} \sqsubseteq \exists \text{hasParent} & (4) \text{Male}(\text{pat}) \\
(2) \text{Adopted} \sqsubseteq \text{Child} & (5) \text{Male}(\text{john}) \\
(3) \text{Female} \sqsubseteq \neg \text{Male} & (6) \text{hasParent}(\text{john}, \text{pat}) 
\end{cases} \]

\[ P = \begin{cases} 
(7) \text{ischildof}(\text{john}, \text{alex}); & (8) \text{boy}(\text{john}); \\
(9) \text{hasfather}(\text{john}, \text{pat}) \leftarrow \text{DL}[; \text{Male} \uplus \text{boy}; \text{Male}](\text{pat}), \\
& \text{DL}[; \text{hasParent}](\text{john}, \text{pat}); \\
(10) \bot \leftarrow \text{not DL}[; \text{Adopted}](\text{john}), \text{pat} \neq \text{alex}, \\
& \text{hasfather}(\text{john}, \text{pat}), \text{ischildof}(\text{john}, \text{alex}), \\
& \text{not DL}[\text{Child} \uplus \text{boy}; \neg \text{Male}](\text{alex}) 
\end{cases} \]

No answer sets.
Example: Inconsistent DL-program

$\Pi = \langle O, P \rangle$ is consistent!

$O = \left\{ \begin{array}{ll}
(1) & \text{Child} \sqsubseteq \exists \text{hasParent} \\
(2) & \text{Adopted} \sqsubseteq \text{Child} \\
(3) & \text{Female} \sqsubseteq \neg \text{Male} \\
\end{array} \right\}$

$P = \left\{ \begin{array}{ll}
(7) & \text{ischildof}(\text{john}, \text{alex}); \\
(8) & \text{boy}(\text{john}); \\
(9) & \text{hasfather}(\text{john}, \text{pat}) \leftarrow \text{DL}[; \text{Male} \uplus \text{boy}; \text{Male}](\text{pat}), \\
& \text{DL}[; \text{hasParent}](\text{john}, \text{pat}); \\
(10) & \bot \leftarrow \text{not DL}[; \text{Adopted}](\text{john}), \text{pat} \neq \text{alex}, \\
& \text{hasfather}(\text{john}, \text{pat}), \text{ischildof}(\text{john}, \text{alex}), \\
& \text{not DL}[\text{Child} \uplus \text{boy}; \neg \text{Male}](\text{alex}) \\
\end{array} \right\}$

$I_1 = \{ \text{ischildof}(\text{john}, \text{alex}), \text{boy}(\text{john}) \}$
Example: Inconsistent DL-program

\[ \Pi = \langle \mathcal{O}, P \rangle \] is consistent!

\[ \mathcal{O} = \begin{cases} 
(1) & \text{Child} \sqsubseteq \exists \text{hasParent} \\
(2) & \text{Adopted} \sqsubseteq \text{Child} \\
(3) & \text{Female} \sqsubseteq \neg \text{Male} \\
\end{cases} \]

\[ P = \begin{cases} 
(7) & \text{ischildof}(john, alex); \quad (8) & \text{boy}(john); \\
(9) & \text{hasfather}(john, pat) \leftarrow \text{DL}[; \text{Male} \cup \text{boy}; \text{Male}](pat), \\
& \quad \text{DL}[; \text{hasParent}](john, pat); \\
(10) & \bot \leftarrow \text{not DL}[; \text{Adopted}](john), \ pat \neq alex, \\
& \quad \text{hasfather}(john, pat), \text{ischildof}(john, alex), \\
& \quad \text{not DL}[\text{Child} \cup \text{boy}; \neg \text{Male}](alex) \\
\end{cases} \]

\[ l_1 = \{ \text{ischildof}(john, alex), \text{boy}(john) \} \]
Repair Answer Sets

Definition
Let $\Pi = \langle \mathcal{O}, \mathcal{P} \rangle$, $\mathcal{O} = \langle \mathcal{T}, \mathcal{A} \rangle$ be a DL-program,

- an ABox $\mathcal{A}'$ is an $x$-repair of $\Pi$ if
  - $\mathcal{O}' = \langle \mathcal{T}, \mathcal{A}' \rangle$ is consistent;
  - $\Pi' = \langle \mathcal{O}', \mathcal{P} \rangle$ has some $x$-answer set.

$\text{rep}_x(\Pi)$ is the set of all $x$-repairs of $\Pi$.

- $I$ is an $x$-repair answer set of $\Pi$, if $I \in \text{AS}_x(\Pi')$, where $\Pi' = \langle \mathcal{O}', \mathcal{P} \rangle$, $\mathcal{O}' = \langle \mathcal{T}, \mathcal{A}' \rangle$, and $\mathcal{A}' \in \text{rep}_x(\Pi)$.

$\text{RAS}_x(\Pi)$ is the set of all $x$-repair AS of $\Pi$.

$\text{rep}^l_x(\Pi)$ is the set of all $\mathcal{A}'$ under which $I$ is an $x$-repair answer set of $\Pi$. 
**Repair Answer Sets**

**Definition**

Let $\Pi = \langle \mathcal{O}, P \rangle$, $\mathcal{O} = \langle T, \mathcal{A} \rangle$ be a DL-program,

- an ABox $\mathcal{A}'$ is an $x$-repair of $\Pi$ if
  - $\mathcal{O}' = \langle T, \mathcal{A}' \rangle$ is consistent;
  - $\Pi' = \langle \mathcal{O}', P \rangle$ has some $x$-answer set.

$\text{rep}_x(\Pi)$ is the set of all $x$-repairs of $\Pi$.

- $I$ is an $x$-repair answer set of $\Pi$, if $I \in \text{AS}_x(\Pi')$, where $\Pi' = \langle \mathcal{O}', P \rangle$, $\mathcal{O}' = \langle T, \mathcal{A}' \rangle$, and $\mathcal{A}' \in \text{rep}_x(\Pi)$.

$\text{RAS}_x(\Pi)$ is the set of all $x$-repair AS of $\Pi$.

$\text{rep}_x^I(\Pi)$ is the set of all $\mathcal{A}'$ under which $I$ is an $x$-repair answer set of $\Pi$.

**Example**

$I_1 = \{ \text{ischildof}(\text{john}, \text{alex}), \text{boy}(\text{john}) \}$ is an $flp$-repair answer set with repair $\mathcal{A}_1' = \{ \text{Male}(\text{pat}), \text{Male}(\text{john}) \}$; $\mathcal{A}_1' \in \text{rep}_{flp}^I(\Pi)$. 
**Complexity of Repair Answer Sets**

**Theorem**
*Deciding $\text{AS}_x(\Pi) \neq \emptyset$ and deciding $\text{RAS}_x(\Pi) \neq \emptyset$ have in all cases the same complexity.*

<table>
<thead>
<tr>
<th>$\Pi$</th>
<th>$\text{RAS}_{\text{FLP}}(\Pi) \neq \emptyset$</th>
<th>$\text{RAS}_{\text{weak}}(\Pi) \neq \emptyset$</th>
</tr>
</thead>
<tbody>
<tr>
<td>normal</td>
<td>$\Sigma_2^P$-complete</td>
<td>$NP$-complete</td>
</tr>
<tr>
<td>disjunctive</td>
<td>$\Sigma_2^P$-complete</td>
<td>$\Sigma_2^P$-complete</td>
</tr>
</tbody>
</table>

**Membership:**
- guess repair $A'$ together with $I$ and proceed with the check as usual;
- deciding $I \models_\mathcal{O} a$ is feasible in polynomial time if $\mathcal{O}$ is in $DL$-$\text{Lite}_A$;

**Hardness:** for normal FLP AS hardness proof of ordinary disjunctive LP can be adapted, for other cases hardness is inherited from ordinary ASP.
**DL-program Evaluation**

**Algorithm 1: AnsSet:** Compute $AS_x(Π)$

| Input: A DL-program $Π$, $x ∈ \{weak, flp\}$ |
| Output: $AS_x(Π)$ |
| for $\hat{I} ∈ AS(Π)$ do |
| if $CMP(\hat{I}, Π) ∧ xFND(\hat{I}, Π)$ then |
| output $\hat{I}|_Π$ |
| end |
| end |

- $\hat{Π}$ is $Π$ with all DL-atoms $a$ substituted by ordinary atoms $e_a$ plus additional guess rules for values of $e_a$;
- $CMP(\hat{I}, Π)$ is a compatibility check, i.e. check whether the values of DL-atoms coincide with the values of their replacement atoms in $\hat{I}$;
- $xFND(\hat{I}, Π)$ is $x$-foundedness check;
- $\hat{I}|_Π$ is a restriction of $\hat{I}$ to original language of $Π$. 
### DL-program Evaluation

**Algorithm 1: \( AnsSet: \) Compute \( AS_x(\Pi) \)**

**Input:** A DL-program \( \Pi, x \in \{\text{weak, flp}\} \)

**Output:** \( AS_x(\Pi) \)

1. for \( \hat{I} \in AS(\hat{\Pi}) \) do
   2a. if \( CMP(\hat{I}, \Pi) \land xFND(\hat{I}, \Pi) \) then
      2b. output \( \hat{I}|_{\Pi} \)
   end

end

**Reasons for inconsistency:**

1. \( \hat{\Pi} \) does not have any answer sets;
2. for all \( \hat{I} \in AS(\Pi) \):
   a. compatibility check failed or
   b. \( x \)-foundedness check failed.
Ontology Repair Problem

To address the compatibility check issue we introduce:

Definition

A ontology repair problem (ORP) is a triple $\mathcal{P} = \langle \mathcal{O}, D_1, D_2 \rangle$, where $\mathcal{O} = \langle T, A \rangle$ is an ontology and $D_i = \{ \langle U^i_j, Q^i_j \rangle | 1 \leq j \leq m_i \}, \ i = 1, 2$ are sets of pairs where $U^i_j$ is any ABox and each $Q^i_j$ is a DL-query.

A repair (solution) for $\mathcal{P}$ is any ABox $A'$ s.t.

- $\mathcal{O}' = \langle T, A' \rangle$ is consistent;
- $\tau(\langle T, A' \cup U^1_k \rangle) \models Q^1_j$ holds for $1 \leq j \leq m_1$;
- $\tau(\langle T, A' \cup U^2_k \rangle) \not\models Q^2_j$ holds for $1 \leq j \leq m_2$. 
Ontology Repair Problem

To address the compatibility check issue we introduce:

**Definition**

A **ontology repair problem (ORP)** is a triple \( \mathcal{P} = \langle \mathcal{O}, D_1, D_2 \rangle \), where \( \mathcal{O} = \langle \mathcal{T}, \mathcal{A} \rangle \) is an ontology and \( D_i = \{\langle U^i_j, Q^i_j \rangle \mid 1 \leq j \leq m_i \} \), \( i = 1, 2 \) are sets of pairs where \( U^i_j \) is any ABox and each \( Q^i_j \) is a DL-query.

**Example**

\[
\Pi = \langle \mathcal{O}, P \rangle, \text{ where } P = \left\{ p(c); r(c); q(c) \leftarrow DL[C \cup r; D](c); \quad \quad \downarrow \leftarrow DL[D \cup p, E \cup r; \neg C](c) \right\}. \\
\]

- \( \hat{I} = \{p(c), r(c), q(c), e_{a_1}\} \): \( a_1 \) is guessed true, \( a_2 \) is guessed false;
- \( \mathcal{P} = \langle \mathcal{O}, D_1, D_2 \rangle \), where
  - \( D_1 = \{\langle \neg C(c) \rangle; D(c) \} \}; \\
  - \( D_2 = \{\langle D(c), \neg E(c) \rangle; \neg C(c) \} \}. \\

\( \mathcal{O} \) is NP-complete in general, even if \( \mathcal{O} = \emptyset \).
Ontology Repair Problem

A repair (solution) for $\mathcal{P}$ is any ABox $\mathcal{A}'$ s.t.

- $\mathcal{O}' = \langle \mathcal{T}, \mathcal{A}' \rangle$ is consistent;
- $\tau(\langle \mathcal{T}, \mathcal{A}' \cup U^1_k \rangle) \models Q^1_j$ holds for $1 \leq j \leq m_1$;
- $\tau(\langle \mathcal{T}, \mathcal{A}' \cup U^2_k \rangle) \not\models Q^2_j$ holds for $1 \leq j \leq m_2$.

Example

Let $\mathcal{O} = \langle E \sqsubseteq D, A \sqsubseteq D, \neg C(c) \rangle$;

- $\mathcal{P} = \langle \mathcal{O}, D_1, D_2 \rangle$, where
  - $D_1 = \{ \langle \{ \neg C(c) \}; D(c) \rangle \}$;
  - $D_2 = \{ \langle \{ D(c), \neg E(c) \}; \neg C(c) \rangle \}$. 

ORP is $\text{NP}$-complete in general, even if $\mathcal{O} = \emptyset$. $\mathcal{A}' = \{ \mathcal{A}(c) \}$ is a solution for $\mathcal{P}$. 

Ontology Repair Problem

A repair (solution) for $\mathcal{P}$ is any ABox $\mathcal{A}'$ s.t.

- $\mathcal{O}' = \langle \mathcal{T}, \mathcal{A}' \rangle$ is consistent;
- $\tau(\langle \mathcal{T}, \mathcal{A}' \cup U_1^1 \rangle) \models Q_j^1$ holds for $1 \leq j \leq m_1$;
- $\tau(\langle \mathcal{T}, \mathcal{A}' \cup U_1^2 \rangle) \not\models Q_j^2$ holds for $1 \leq j \leq m_2$.

Example

Let $\mathcal{O} = \langle E \sqsubseteq D, A \sqsubseteq D, \neg C(c) \rangle$;

- $\mathcal{P} = \langle \mathcal{O}, D_1, D_2 \rangle$, where $\mathcal{A}' = \{A(c)\}$ is a solution for $\mathcal{P}$.
  - $D_1 = \{\langle \{\neg C(c)\}; D(c) \rangle \}$;
  - $D_2 = \{\langle \{D(c), \neg E(c)\}; \neg C(c) \rangle \}$.
Ontology Repair Problem

A repair (solution) for $\mathcal{P}$ is any ABox $\mathcal{A}'$ s.t.

- $\mathcal{O}' = \langle \mathcal{T}, \mathcal{A}' \rangle$ is consistent;
- $\tau(\langle \mathcal{T}, \mathcal{A}' \cup \mathcal{U}_k^1 \rangle) \models Q_j^1$ holds for $1 \leq j \leq m_1$;
- $\tau(\langle \mathcal{T}, \mathcal{A}' \cup \mathcal{U}_k^2 \rangle) \not\models Q_j^2$ holds for $1 \leq j \leq m_2$.

Example

Let $\mathcal{O} = \langle E \sqsubseteq D, A \sqsubseteq D, \neg C(c) \rangle$;

- $\mathcal{P} = \langle \mathcal{O}, D_1, D_2 \rangle$, where $\mathcal{A}' = \{A(c)\}$ is a solution for $\mathcal{P}$.
  - $D_1 = \{\langle \{\neg C(c)\}; D(c) \rangle\}$;
  - $D_2 = \{\langle \{D(c), \neg E(c)\}; \neg C(c) \rangle\}$.

ORP is $NP$-complete in general, even if $\mathcal{O} = \emptyset$. 
Consider a set $\mathcal{AB}$ of all possible ABoxes. Function $\sigma : 2^{\mathcal{AB}} \times \mathcal{AB} \rightarrow 2^{\mathcal{AB}}$ is a selection function. $\sigma(S, A) \subseteq S$ is a set of preferred ABoxes.

A selection $\sigma : 2^{\mathcal{AB}} \times \mathcal{AB} \rightarrow 2^{\mathcal{AB}}$ is independent if $\sigma(S, A) = \sigma(S', A) \cup \sigma(S \setminus S', A)$, whenever $S' \subseteq S$.

Example

- deletion repair is independent;
- set-minimal change repair is not independent;
- cardinality minimal change repair is not independent.
**Tractable Cases of ORP**

C1. **bounded \(\delta^{\pm}\)-change:** \(\sigma_{\delta^{\pm},k}(S, A) = \{A' \mid |A' \Delta A| \leq k\}\), for some \(k\);

C2. **deletion repair:** \(\sigma_{\text{del}}(S, A) = \{A' \mid A' \subseteq A\}\);  

C3. **deletion \(\delta^{+}\):** first apply \(\sigma_{\text{del}}\) and get \(\mu(O)\) s.t. for all \(1 \leq j \leq m_2\)  

\[ \tau(\langle T, A' \cup U_j^2 \rangle) \not= Q_j^2 \text{, then further compute } \sigma_{\delta^+}(S, \mu(O)) \];

C4. **addition under bounded opposite polarity:**  

\[ \sigma_{bop}(S, A) = \{A' \supseteq \mu(O)\mid |A'^{+}\setminus A| \leq k \text{ or } |A'^{-}\setminus A| \leq k\} \]

C1 - C4 are independent.

**Applicability of results for independent selections:**

- deciding whether repair \(A'\) is selected by \(\sigma\)  
  does not require looking at other repairs;

- without major complexity increase \(\sigma\)'s can be combined with  
  - DB-style factorization and localization techniques;  
  - local search.
Repair Answer Set Computation

Algorithm 2: $\text{RepAns}$: Compute $rep_{(\sigma, x)}^{\hat{I}|\Pi}(\Pi)$

\begin{align*}
\text{Input: } & \Pi = \langle \mathcal{O}, P \rangle, \mathcal{O} = \langle T, A \rangle, \hat{I} \in AS(\Pi), \sigma, x \in \{\text{weak, flp}\} \\
\text{Output: } & rep_{(\sigma, x)}^{\hat{I}|\Pi}(\Pi) \\
\text{for } & \mathcal{A}' \in \text{ORP}(\hat{I}, \Pi, \sigma) \text{ do} \\
& \quad \text{if } CMP(\hat{I}, \langle T, \mathcal{A}', P \rangle) \land x\text{FND}(\hat{I}, \langle T, \mathcal{A}', P \rangle) \text{ then} \\
& \quad \quad \text{output } \mathcal{A}' \\
& \text{end} \\
\text{end}
\end{align*}

- $\text{ORP}(\hat{I}, \Pi, \sigma)$ computes $\sigma$ repairs for $\hat{I}, \Pi$;
- $\text{CMP}(\hat{I}, \langle T, \mathcal{A}', P \rangle)$ checks whether $\hat{I}$ is compatible w.r.t. $\Pi'$;
- $x\text{FND}(\hat{I}, \langle T, \mathcal{A}', P \rangle)$ checks whether $\hat{I}$ is $x$-founded w.r.t. $\Pi'$.

$\text{RepAnsSet}$ outputs $\hat{I}$ if the result of $\text{RepAns}$ is nonempty.
Repair Answer Set Computation

**Algorithm 2: RepAns: Compute** $rep_{(\sigma,x)}^{\hat{I}|\Pi}(\Pi)$

**Input:** $\Pi = \langle \mathcal{O}, P \rangle$, $\mathcal{O} = \langle \mathcal{T}, \mathcal{A} \rangle$, $\hat{I} \in AS(\hat{\Pi})$, $\sigma$, $x \in \{weak, flip\}$

**Output:** $rep_{(\sigma,x)}^{\hat{I}|\Pi}(\Pi)$

```plaintext
for $\mathcal{A}' \in ORP(\hat{I}, \Pi, \sigma)$ do
  if $CMP(\hat{I}, \langle T, A', P \rangle)$ $\land$ $xFND(\hat{I}, \langle T, A', P \rangle)$ then
    output $\mathcal{A}'$
  end
end
```

- $ORP(\hat{I}, \Pi, \sigma)$ computes $\sigma$ repairs for $\hat{I}, \Pi$;
- $CMP(\hat{I}, \langle T, A', P \rangle)$ checks whether $\hat{I}$ is compatible w.r.t. $\Pi'$;
- $xFND(\hat{I}, \langle T, A', P \rangle)$ checks whether $\hat{I}$ is $x$-founded w.r.t. $\Pi'$.

$RepAnsSet$ outputs $\hat{I}$ if the result of $RepAns$ is nonempty.

$RepAns$ and $RepAnsSet$ are **sound** and **complete** for independent $\sigma$. 
Related Work

- Repairing ontologies
  - consistent query answering over \textit{DL-Lite} ontologies based on repair technique [Lembo \textit{et al.}, 2010], [Bienvenu, 2012];
  - QA to \textit{DL-Lite}_A ontologies that miss expected tuples (abductive explanations corresponding to repairs) [Calvanese \textit{et al.}, 2012].

- Repairing nonmonotonic logic programs
  - extended abduction for deleting minimal sets of rules (in reality addition is also possible) [Sakama and Inoue, 2003].

- Repairing inconsistent combination of rules and ontologies
  - paraconsistent semantics, based on the HT logic [Fink, 2012];
  - inconsistency tolerance in DL-programs [Pührer \textit{et al.}, 2010].
Conclusion and Future Work

Conclusions:
- consideration of repair answer sets (RAS);
- same complexity as ordinary AS (for $\mathcal{O}$ in $DL-Lite_A$);
- RAS computation by extending the existing evaluation algorithm;
- involvement of a generalized ontology repair problem (ORP);
- tractable cases for independent selections.

Future work:
- extending the work to other DLs ($\mathcal{EL}^{++}$, RL);
- DL-programs with richer queries (unions of conjunctive queries);
- further $\sigma$-selections;
- optimization and implementation.
Meghyn Bienvenu.
On the complexity of consistent query answering in the presence of simple ontologies.

Diego Calvanese, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati.
Tractable reasoning and efficient query answering in description logics: The DL-Lite family.
Diego Calvanese, Magdalena Ortiz, Mantas Simkus, and Giorgio Stefanoni.
The complexity of explaining negative query answers in DL-Lite.

Michael Fink.
Paraconsistent hybrid theories.
Domenico Lembo, Maurizio Lenzerini, Riccardo Rosati, Marco Ruzzi, and Domenico Fabio Savo.

Jörg Pührer, Stijn Heymans, and Thomas Eiter.

Chiaki Sakama and Katsumi Inoue.
DL-program: syntax

Signature: $\Sigma = \langle C, I, P, C, R \rangle$, where
- $\Sigma_0 = \langle I, C, R \rangle$ is a DL signature;
- $C \supseteq I$ is a set of constant symbols;
- $P$ is a finite set of predicate symbols of arity $\geq 0$, s.t. $P \cap \{C \cup R\} = \emptyset$.

DL-atom is of the form $DL[S_1 op_1 p_1, \ldots, S_m op_m p_m; Q](t)$, $m \geq 0$, where
- $S_i \in C \cup R$;
- $op_i \in \{\cup, \cup, \cap\}$;
- $p_i \in P$ (unary or binary);
- $Q(t)$ is a DL-query:
  - $C(t_1), \neg C(t_1), t = t_1$, where $C \in C$;
  - $R(t_1, t_2), \neg R(t_1, t_2), t = t_1, t_2$, where $R \in R$.
  - $C \sqsubseteq D, C \not\sqsubseteq D, t = \epsilon$, where $C, D \in C \cup \{\top, \bot\}$;

DL-program: $\Pi = \langle O, P \rangle$, $O$ is a DL ontology, $P$ is a set of DL-rules:

$$a_1 \lor \ldots \lor a_n \leftarrow b_1, \ldots, b_k, \text{not } b_{k+1}, \ldots, \text{not } b_m,$$

$m \geq k \geq 0$, $a_i$ is a classical literal; $b_i$ is a classical literal or a DL-atom.
DL-program: semantics

Consider grounding $grd(\Pi) = \langle O, grd(P) \rangle$ of $\Pi = \langle O, P \rangle$ over $C$ and $P$.

Interpretation $I$ is a consistent set of ground literals over $C$ and $P$.

- for ground literal $\ell$: $I \models^O \ell$ iff $\ell \in I$;
- for ground DL-atom $a = DL[S_1 op_1 p_1, \ldots, S_m op_m p_m; Q](c)$:

  $I \models^O a$

  iff $\tau(\langle T, A \cup \lambda^I(a) \rangle) \models Q(c)$, where $\tau(O)$ is a modular translation of $O$ to FOL, $\lambda^I(a) = \bigcup_{i=1}^{m} A_i(I)$ is a DL-update of $O$ under $I$ by $a$:

  - $A_i(I) = \{ S_i(t) \mid p_i(t) \in I \}$, for $op_i = \cup$;
  - $A_i(I) = \{ \neg S_i(t) \mid p_i(t) \in I \}$, for $op_i = \cup$;
  - $A_i(I) = \{ \neg S_i(t) \mid p_i(t) \notin I \}$, for $\land$.

FLP-reduct $\rho_{flp}P^I$ of $P$ is a set of ground DL-rules $r$ s.t. $I \models b^+(r)$, $I \not\models b^-(r)$.

Weak-reduct $\rho_{weak}P^I$ of $P$: removes all DL-atoms $b_i$, $1 \leq i \leq k$ and all not $b_j$, $k < j \leq m$ from the rules of $\rho_{flp}P^I$.

$I$ is an $x$-answer set of $P$ iff $I$ is a minimal model of its $x$-reduct.