Explaining and Repairing Description Logic Ontologies

Franz Baader
Theoretical Computer Science
TU Dresden, Germany

Source: Protégé Wiki
Explaining and Repairing Description Logic Ontologies

Franz Baader
Theoretical Computer Science
TU Dresden, Germany
General goal

“develop formalisms for providing high-level descriptions of the world that can be effectively used to build intelligent applications”

[Brachman & Nardi, 2003]

- formalism: well-defined syntax and formal, unambiguous semantics
- high-level description: only relevant aspects represented, others left out
- intelligent applications: must be able to reason about the knowledge, and infer implicit knowledge from the explicitly represented knowledge
- effectively used: need for practical reasoning tools and efficient implementations
Description Logics

- Family of logic-based knowledge representation languages tailored towards representing terminological knowledge

- Many DLs are decidable fragments of first-order logic

- Close relationship to propositional modal logics

- Design goal: good compromise between expressiveness and complexity

- Decidability and complexity results for a great variety of DLs and various inference problems, but also implementation of practical systems
  
  - very expressive DLs of high worst-case complexity, but with highly optimized “practical” reasoning procedures
  
  - inexpressive DLs with tractable inference problems, which are expressive enough for certain applications

- Applications: natural language processing, configuration, databases, modelling in engineering domains, ontologies (Web ontology language OWL, biomedical ontologies)
Description Logics from a general point of view

Concepts
- **Constructors** for building complex concept descriptions out of atomic concepts (unary predicates) and roles (binary predicates).
- Interpretation $\mathcal{I}$ assigns sets $C^\mathcal{I}$ to concept descriptions $C$ according to the semantics of the constructors.

TBoxes
- Finite set of **general concept inclusions** (GCIs) of the form $C \subseteq D$ where $C, D$ are concept descriptions.
- The interpretation $\mathcal{I}$ is a model of a TBox $\mathcal{T}$ if $C^\mathcal{I} \subseteq D^\mathcal{I}$ holds for all GCIs $C \subseteq D$ in $\mathcal{T}$.

ABoxes
- Finite set of **assertions** of the form $C(a)$ and $r(a, b)$ where $C$ is a concept description, $r$ a role, and $a, b$ individual names.
- The interpretation $\mathcal{I}$ is a model of an ABox $\mathcal{A}$ if $a^\mathcal{I} \in C^\mathcal{I}$ and $(a^\mathcal{I}, b^\mathcal{I}) \in r^\mathcal{I}$ holds for all assertions $C(a)$ and $r(a, b)$ in $\mathcal{A}$. 
Constructors of the DL $\mathcal{ALC}$

- top concept $\top$, negation $\neg C$
- conjunction $C \sqcap D$, disjunction $C \sqcup D$, existential restriction $\exists r.C$
- value restriction $\forall r.C$

**TBox**

- General concept inclusion (GCI)
  - $\text{Good\_course} \sqsubseteq \text{Course} \sqcap \ldots$
  - $\exists \text{has\_teacher}. \top \sqsubseteq \text{Course}$
  - $\exists \text{has\_student}. \text{Smart} \sqsubseteq \forall \text{has\_teacher}. \text{Happy}$

**ABox**

- Properties of individuals
  - $\text{Good\_Course}(\text{Course123})$
  - $\text{has\_teacher}(\text{Course123}, \text{Franz})$
  - $\text{has\_topic}(\text{Course123}, \text{DL})$
Constructors of the DL $\mathcal{ALC}$

- top concept $\top$, negation $\neg C$
- conjunction $C \cap D$, disjunction $C \cup D$
- existential restriction $\exists r.C$
- value restriction $\forall r.C$

An advanced course that $\text{Course} \cap \text{Advanced} \cap$
has a smart or studious student, $\exists \text{has\_student}.(\text{Smart} \cup \text{Studious}) \cap$
no easy topic, $\forall \text{has\_topic}.\neg \text{Easy} \cap$
and a teacher $\exists \text{has\_teacher}.\top$

The semantics of the constructors is defined through identities:

- $(C \cap D)^I = C^I \cap D^I$,
- $(\exists r.C)^I = \{d \mid \exists e.(d, e) \in r^I \land e \in C^I\}$,
- $(\forall r.C)^I = \{d \mid \forall e.(d, e) \in r^I \rightarrow e \in C^I\}$,
- $\ldots$

Dresden
Reasoning makes implicitly represented knowledge explicit, provided as service by the DL system, e.g.:

Subsumption: Is \( C \) a subconcept of \( D \)?

\[ \mathcal{T} \models C \sqsubseteq D \iff C^\mathcal{I} \subseteq D^\mathcal{I} \text{ for all models } \mathcal{I} \text{ of the TBox } \mathcal{T}. \]

Satisfiability: Is the concept \( C \) non-contradictory?

\( C \) is satisfiable w.r.t. \( \mathcal{T} \) iff \( C^\mathcal{I} \neq \emptyset \) for some model \( \mathcal{I} \) of \( \mathcal{T} \).

Consistency: Is the ABox \( \mathcal{A} \) non-contradictory?

\( \mathcal{A} \) is consistent w.r.t. \( \mathcal{T} \) iff it has a model that is also a model of \( \mathcal{T} \).

Instantiation: Is \( e \) an instance of \( C \)?

\[ (\mathcal{A}, \mathcal{T}) \models C(e) \iff e^\mathcal{I} \in C^\mathcal{I} \text{ for all models } \mathcal{I} \text{ of } \mathcal{T} \text{ and } \mathcal{A}. \]
### Complexity of subsumption reasoning \( \mathcal{T} \models C \subseteq D \)

depends on the expressivity of the DL and the TBox formalism

<table>
<thead>
<tr>
<th></th>
<th>no TBox</th>
<th>acyclic TBox</th>
<th>cyclic TBox</th>
<th>general TBox</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>( \mathcal{FL}_0 )</strong></td>
<td>( \text{P} )</td>
<td>( \text{coNP} )</td>
<td>( \text{PSpace} )</td>
<td>( \text{ExpTime} )</td>
</tr>
<tr>
<td><strong>( \mathcal{ALC} )</strong></td>
<td>( \text{PSpace} )</td>
<td>( \text{PSpace} )</td>
<td>( \text{ExpTime} )</td>
<td>( \text{ExpTime} )</td>
</tr>
<tr>
<td><strong>( \mathcal{EL} )</strong></td>
<td>( \text{P} )</td>
<td>( \text{P} )</td>
<td>( \text{P} )</td>
<td>( \text{P} )</td>
</tr>
</tbody>
</table>

\( \mathcal{FL}_0 \): \( C \sqcap D, \; \forall r.C, \; \top \)

\( \mathcal{EL} \): \( C \sqcap D, \; \exists r.C, \; \top \)

© Franz Baader
Reasoning example in Protégé
Error management and explanation

- Large ontologies often contain errors, which are usually detected when unintended consequences are deduced.
- Even some of the intended consequences may appear to be unintuitive to users.

Understanding the reasons for unintuitive or unintended consequences can be difficult:

- W.r.t. a previous version of the medical ontology SNOMED CT, the concept *Amputation-of-finger* was classified as a subconcept of *Amputation-of-hand*.
- Finding and understanding the reason for this in a large ontology with ∼350 000 GCIs is not easy.
Error management and explanation comes in different flavours

- Pinpointing: identify the source of the consequence

  *Minimal subsets of the ontology from which a given consequence follows.*

- Explanation: provide a convincing argument for the consequence

  *Show a proof of the consequence in an appropriate calculus.*

- Repair: provide suggestions for error resolution

  *Maximal subsets of the ontology from which the consequence does not follow.*

  *Optimal repairs preserve a maximal set of consequences while removing the unwanted ones.*
Axiom Pinpointing

determine the source of the consequence

Justification: minimal subset of the ontology that has the consequence

Given: ontology $O$ and GCI or concept assertion $\alpha$ such that $O \models \alpha$

Justification: minimal subset $J \subseteq O$ such that $J \models \alpha$

\[
\begin{align*}
\mathcal{T} & : 
\begin{align*}
a_1 & : \ A \subseteq \exists r.A \\
a_2 & : \ A \subseteq Y \\
a_3 & : \ \exists r. Y \subseteq B \\
a_4 & : \ Y \subseteq B
\end{align*}
\end{align*}
\]

\[\mathcal{T} \models A \subseteq B\]

Justifications: \(\{a_2, a_4\}\), \(\{a_1, a_2, a_3\}\)
Axiom Pinpointing

scientific challenges

- How can we compute justifications?
- How many justifications does a consequence have
  (in the worst case or in practice)?
- How hard is it to compute one or to enumerate all justifications?

Pinpointing in $\mathcal{EL}$: [B., Peñaloza, Suntisrivaraporn; 2007]
[B., Suntisrivaraporn; 2008] [Peñaloza, Sertkaya; 2017]

- Both black box and glass box approaches for computing justifications.
- A given consequence may have exponentially many justifications
  in the cardinality of $\mathcal{O}$.
- In our experiments with SNOMED CT, most of the subsumption
  consequences (78%) had justifications of size at most 10.
- A single justification can be computed in polynomial time.
- Unless P=NP, there is no output polynomial algorithm for enumerating
  all justifications.
Justification example in Protégé
Proofs
to explain DL entailment

- Given a justification \( J \) for a consequence, the user still needs to understand how the consequence can be derived using the axioms in \( J \).
- A proof provides us with a step by step derivation using easy to understand proof rules.

Proof rules for \( \mathcal{EL} \)

<table>
<thead>
<tr>
<th>CR1</th>
<th>( A \sqsubseteq A )</th>
</tr>
</thead>
<tbody>
<tr>
<td>CR2</td>
<td>( A \sqsubseteq \top )</td>
</tr>
<tr>
<td>CR3</td>
<td>( A \sqsubseteq B ) if ( A \sqsubseteq B \in \mathcal{T} )</td>
</tr>
<tr>
<td>CR4</td>
<td>( A_1 \sqsubseteq A_2 ) ( A_2 \sqsubseteq A_3 ) ( A_1 \sqsubseteq A_3 )</td>
</tr>
<tr>
<td>CR5</td>
<td>( A \sqsubseteq A_1 ) ( A \sqsubseteq A_2 ) ( A_1 \sqcap A_2 \sqsubseteq B ) ( A \sqsubseteq B )</td>
</tr>
<tr>
<td>CR6</td>
<td>( A \sqsubseteq \exists r.A_1 ) ( A_1 \sqsubseteq B_1 ) ( \exists r.B_1 \sqsubseteq B ) ( A \sqsubseteq B )</td>
</tr>
</tbody>
</table>

[B., Horrocks, Lutz, Sattler; 2017]
Proofs

example of proof in the $\mathcal{EL}$ calculus

\[
\begin{array}{c}
\text{CR1} \quad A \sqsubseteq A \\
\text{CR2} \quad A \sqsubseteq \top \\
\text{CR3} \quad A \sqsubseteq B \quad \text{if } A \sqsubseteq B \in \mathcal{T} \\
\text{CR4} \quad A_1 \sqsubseteq A_2, A_2 \sqsubseteq A_3 \Rightarrow A_1 \sqsubseteq A_3 \\
\text{CR5} \quad A \sqsubseteq A_1, A \sqsubseteq A_2, A_1 \sqcap A_2 \sqsubseteq B \Rightarrow A \sqsubseteq B \\
\text{CR6} \quad A \sqsubseteq \exists r.A, A_1 \sqsubseteq B_1, \exists r.B_1 \sqsubseteq B \Rightarrow A \sqsubseteq B
\end{array}
\]

$\mathcal{T}$

\[
\begin{array}{c}
a_1 : A \sqsubseteq \exists r.A \\
a_2 : A \sqsubseteq Y \\
a_3 : \exists r.Y \sqsubseteq B
\end{array}
\]
Proofs to explain DL entailment

Scientific challenges

- What are “good proofs” for explanation purpose, depending on the experience of the user? *User studies* [Alrabbaa et al.; 2022]

- Once a measure of the quality of proofs is fixed, how hard is it to compute optimal proofs? *Complexity results and algorithms* [Alrabbaa et al.; 2020] [Alrabbaa et al.; 2021]

- How can one display proofs in an easily legible and adaptable way? *Interactive visualisation tool Evonne* [Méndez et al.; 2023]
Proofs

complexity of computing good ones
[Alrabbaa et al.; 2020] [Alrabbaa et al.; 2021]

Formal framework

\[ (\mathcal{T}, \eta) \]

Deriver \[ \mathcal{D} \]

(Reasoner, Calculus)

Derivation Structure \[ \mathcal{D}(\mathcal{T}, \eta) \]

Proof \[ \mathcal{P} \]

- **Problem:** find a proof in the derivation structure with a value of the measure below a given threshold \( t \)
- **Complexity results** for different types of derivers, measures, and encoding of the number \( t \)
### Proofs

Complexity of computing good ones

[Alrabbaa et al.; 2020]  [Alrabbaa et al.; 2021]

<table>
<thead>
<tr>
<th></th>
<th>polynomial unary</th>
<th>polynomial binary</th>
<th>exponential unary</th>
<th>exponential binary</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size</td>
<td>NP</td>
<td>NP</td>
<td>NP</td>
<td>NExpTime</td>
</tr>
<tr>
<td>Monotone recursive (\Phi)-measures</td>
<td>(\leq P)</td>
<td>(\leq P)</td>
<td>(\leq \text{ExpTime})</td>
<td>(\leq \text{ExpTime})</td>
</tr>
<tr>
<td>Tree size</td>
<td>P</td>
<td>P</td>
<td>NP</td>
<td>PSpace</td>
</tr>
<tr>
<td>Depth</td>
<td>P</td>
<td>P</td>
<td>PSpace</td>
<td>ExpTime</td>
</tr>
<tr>
<td>Logarithmic depth</td>
<td>P</td>
<td>P</td>
<td>ExpTime</td>
<td>ExpTime</td>
</tr>
</tbody>
</table>

- **Problem:** find a proof in the derivation structure with a value of the measure below a given threshold \(t\)

- **Complexity results** for different types of derivers, measures, and encoding of the number \(t\)
Proofs in Protégé
using our proof plugin Evée

and in our interactive visualisation tool
Evonne
Classical Repair

remove the source of the consequence

subset of the ontology that does not have the unintended consequence

Given: ontology $O$ and GCI or concept assertion $\alpha$ such that $O \models \alpha$

Classical repair: subset $R \subseteq O$ such that $R \not\models \alpha$

Optimal classical repair: maximal subset $R \subseteq O$ such that $R \not\models \alpha$

Algorithm for computing an optimal classical repair: [Reiter; 1987]

1. Compute all justifications $J_1, \ldots, J_k$ of $\alpha$.
2. Compute a minimal hitting set $H$ of $J_1, \ldots, J_k$. 
   \[ H \cap J_i \neq \emptyset \]
   for $i = 1, \ldots, k$.
3. Output $R := O \setminus H$.

For every choice of the hitting set, this algorithm produces an optimal classical repair, and all optimal classical repairs can be generated this way.
Classical Repair

remove the source of the consequence

\[ a_1 : \ A \subseteq \exists r.A \]
\[ a_2 : \ A \subseteq Y \]
\[ a_3 : \ \exists r.Y \subseteq B \]
\[ a_4 : \ Y \subseteq B \]

\[ \models \alpha = A \subseteq B \]

Justifications:
\{a_2, a_4\}, \ \{a_1, a_2, a_3\}

Minimal hitting sets:
\{a_2\}, \ \{a_1, a_4\}, \ \{a_3, a_4\}

Optimal classical repairs:
\{a_1, a_3, a_4\}, \ \{a_2, a_3\}, \ \{a_1, a_2\}

Diagnoses

Algorithm for computing an optimal classical repair: [Reiter; 1987]

1. Compute all justifications \( J_1, \ldots, J_k \) of \( \alpha \).
2. Compute a minimal hitting set \( H \) of \( J_1, \ldots, J_k \).
   \[ H \cap J_i \neq \emptyset \]
   for \( i = 1, \ldots, k \).
3. Output \( R := \mathcal{O} \setminus H \).

For every choice of the hitting set, this algorithm produces an optimal classical repair, and all optimal classical repairs can be generated this way.
Repairs

in our interactive visualisation tool

Evonne
Classical Repair may remove too many consequences

\[
\mathcal{T} = \{\exists \text{owns}. (GermanCar \cap Diesel) \subseteq \exists \text{gets}. \text{Compensation}\}
\]

\[
\mathcal{A} = \{\exists \text{owns}. (GermanCar \cap Diesel)(Robert)\}
\]

\[
\alpha = \exists \text{gets}. \text{Compensation}(Robert)
\]

Classical repair: remove \(\exists \text{owns}. (GermanCar \cap Diesel)(Robert)\)

More gentle: replace the assertion with \(\exists \text{owns}. (GermanCar)(Robert)\)

Even more gentle: replace the assertion with

\(\exists \text{owns}. \text{GermanCar}(Robert)\) and \(\exists \text{owns}. \text{Diesel}(Robert)\)

Main idea to get better repairs:
consider inclusion for consequences instead of inclusion for axioms

\[
\text{Con}(\mathcal{O}) := \{\alpha \mid \mathcal{O} \models \alpha\}
\]
Optimal Repair

- The ontology $\mathcal{R}$ is a repair of $\mathcal{O}$ w.r.t. $\alpha$ if
  \[ \text{Con}(\mathcal{R}) \subseteq \text{Con}(\mathcal{O}) \setminus \{\alpha\}. \]
- This repair is optimal if there is no repair $\mathcal{R}'$ of $\mathcal{O}$ w.r.t. $\alpha$ with
  \[ \text{Con}(\mathcal{R}) \subset \text{Con}(\mathcal{R}'). \]

Example: replacing $\exists\text{owns.}(\text{GermanCar} \sqcap \text{Diesel})(\text{Robert})$ with $\exists\text{owns.}\text{GermanCar}(\text{Robert})$ and $\exists\text{owns.}\text{Diesel}(\text{Robert})$ yields an optimal repair.
Optimal Repair

need not exist even for ABoxes without TBox

Example

Consider \( \mathcal{O} := \{ V(n), \ell(n, n) \} \)
and \( \alpha := V(n) \).

For all \( k \geq 0 \), the assertion \( \exists \ell. (V \land (\exists \ell.)^k. \top)(n) \) belongst to \( \text{Con}(\mathcal{O}) \).

Adding finitely many of them to \( \mathcal{O} \setminus \{ \alpha \} \) yields a repair, but every finite repair entails only finitely many of them.

Using quantified ABoxes with anonymous individuals solves this problem:

\( \exists \{ x \}. \mathcal{R} \) for \( \mathcal{R} := \{ \ell(n, x), \ell(x, n), V(x), \ell(x, x) \} \)
is an optimal repair.
Optimal Repair

- Determine cases for which optimal repairs always exist and cover all repairs.
- How many optimal repairs are there and how large can they become?
- How hard is it to compute one or all optimal repairs?

Case of quantified ABoxes w.r.t. static $\mathcal{EL}$ TBoxes:

- If we consider only concept assertions as consequences, then existence and coverage are satisfied. There may be exponentially many optimal repairs of up to exponential size, which can be computed in exponential time.

- If we consider conjunctive queries as consequences, then we must additionally assume that the TBox is cycle-restricted and the computation algorithm requires an NP-oracle.

[B., Kriegel, Nuradiansyah, Peñaloza; 2020]
[B., Koopmann, Kriegel, Nuradiansyah; 2021]
Summary

- Pinpointing: identify the source of the consequence

  *Minimal subsets of the ontology from which a given consequence follows.*

- Explanation: provide a convincing argument for the consequence

  *Show a proof of the consequence in an appropriate calculus.*

- Repair: provide suggestions for error resolution

  *Maximal subsets of the ontology from which the consequence does not follow.*

  *Optimal repairs preserve a maximal set of consequences while removing the unwanted ones.*
References

Pinpointing

- Franz Baader, Rafael Pealoza, Boontawee Suntisrivaraporn: Pinpointing in the Description Logic $\mathcal{EL}^+$. KI 2007, LNCS 4667: 52-67.


References

Proofs


Repair


- Franz Baader, Francesco Kriegl, Adrian Nuradiansyah, Rafael Pealoza: Computing Compliant Anonymisations of Quantified ABoxes w.r.t. $EL$ Policies. ISWC (1) 2020, LNCS 12506: 3-20.

- Franz Baader, Patrick Koopmann, Francesco Kriegl, Adrian Nuradiansyah: Computing Optimal Repairs of Quantified ABoxes w.r.t. Static $EL$ TBoxes. CADE 2021, LNCS 12699: 309-326
Systems  download links

Protégé  ontology editor

https://protege.stanford.edu/

Evee  library and Protégé plugin for justifications and proofs

https://github.com/de-tu-dresden-inf-lat/evee

Evonne  visualisation tool supporting explanation and repair

https://imld.de/en/research/research-projects/evonne/