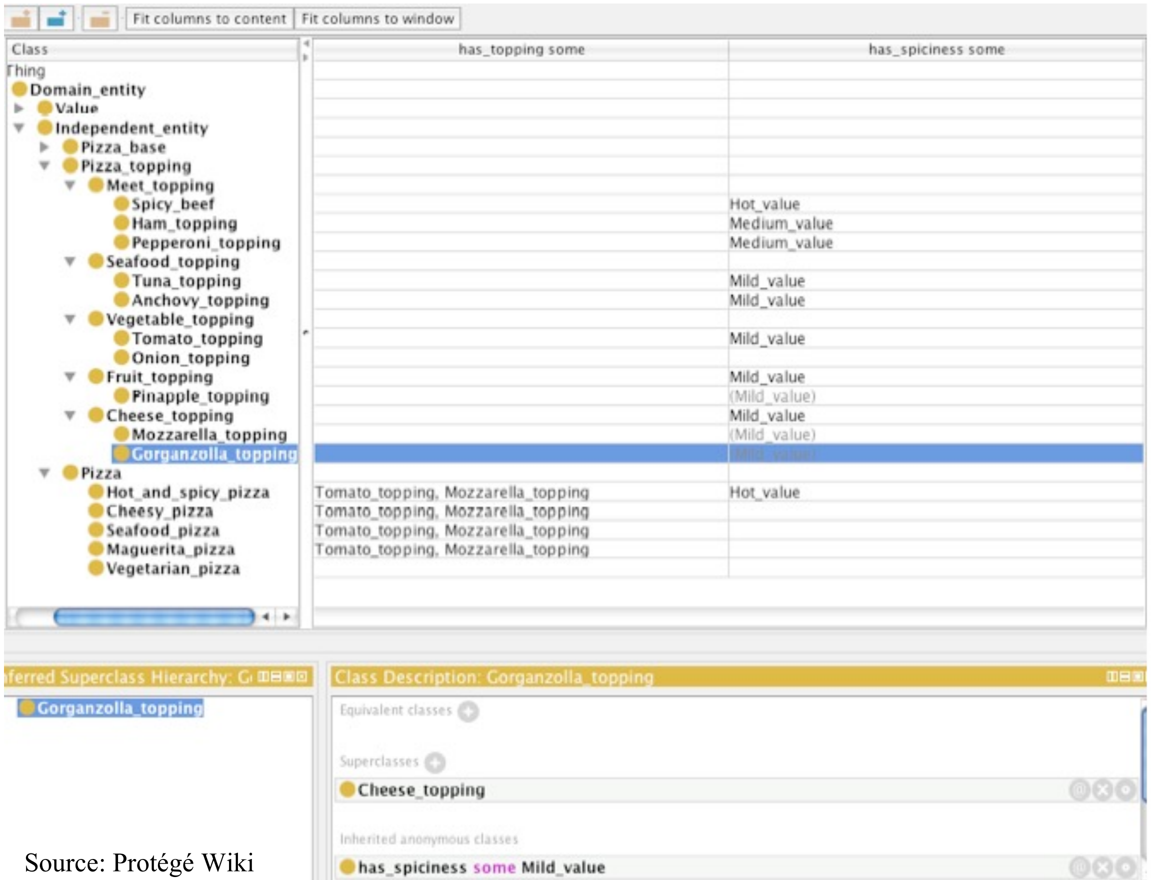
Explaining and Repairing Description Logic Ontologies

Franz Baader

Theoretical Computer Science

TU Dresden, Germany



Source: Protégé Wiki

TECHNISCHE UNIVERSITÄT DRESDEN

DFG Deutsche Forschungsgemeinschaft

Explaining and Repairing Description Logic Ontologies

Franz Baader
Theoretical Computer Science
TU Dresden, Germany

TECHNISCHE UNIVERSITÄT DRESDEN

DFG Deutsche Forschungsgemeinschaft

Knowledge Representation

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 — *FaCT, Racer, Pellet, HermiT, . . . Konclude, MORe*
  - inexpressive DLs with tractable inference problems, which are expressive enough for certain applications — *CEL, Snorocket, ELK, QuOnto, Mastro, ontop*
- Applications: natural language processing, configuration, databases, modelling in engineering domains, ontologies (Web ontology language OWL, biomedical ontologies)

Dresden

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.

Ontology

TBoxes

- Finite set of general concept inclusions (GCIs) of the form $C \sqsubseteq D$ 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 $C \sqsubseteq D$ 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 $\forall r.C$

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

TBox

General concept inclusion (GCI)

$Good\_course \sqsubseteq Course \sqcap \ldots$

$\exists has\_teacher.\top \sqsubseteq Course$

$\exists has\_student.Smart \sqsubseteq \forall has\_teacher.Happy$

ABox

Properties of `individuals`

$Good\_Course(\texttt{Course123})$

$has\_teacher(\texttt{Course123}, \texttt{Franz})$

$has\_topic(\texttt{Course123}, \texttt{DL})$

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 $\forall r.C$

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

The semantics of the constructors is defined through identities:

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

Reasoning

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

polynomial reductions

Subsumption: Is C a subconcept of D?

$\mathcal{T} \models C \sqsubseteq D$ iff $C^\mathcal{I} \subseteq D^\mathcal{I}$ for all models $\mathcal{I}$ 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 $e^\mathcal{I} \in C^\mathcal{I}$ for all models $\mathcal{I}$ of $\mathcal{T}$ and $\mathcal{A}$.

in presence of negation

Complexity

of subsumption reasoning $\mathcal{T} \models C \sqsubseteq D$

depends on the expressivity of the DL and the TBox formalism

	no TBox	acyclic TBox	cyclic TBox	general TBox
$\mathcal{FL}_0$	P	coNP	PSpace	ExpTime
$\mathcal{ALC}$	PSpace	PSpace	ExpTime	ExpTime
$\mathcal{EL}$	P	P	P	P

$\mathcal{FL}_0$: $C \sqcap D$, $\forall r.C$, $\top$

$\mathcal{EL}$: $C \sqcap D$, $\exists r.C$, $\top$

Dresden

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.

Dresden

Axiom Pinpointing

determine the source of the consequence

Justification: minimal subset of the ontology that has the consequence

Given: ontology $\mathcal{O}$ and GCI or concept assertion α such that $\mathcal{O} \models \alpha$

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

$\mathcal{T}$

$$
\begin{aligned}
a_1 &: \quad A \sqsubseteq \exists r.A \\
a_2 &: \quad A \sqsubseteq Y \\
a_3 &: \quad \exists r.Y \sqsubseteq B \\
a_4 &: \quad Y \sqsubseteq B
\end{aligned}
$$

$\mathcal{T} \models A \sqsubseteq B$

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

Dresden

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 $\mathcal{EL}$

CR1 $\dfrac{}{A \sqsubseteq A}$ CR2 $\dfrac{}{A \sqsubseteq \top}$ CR3 $\dfrac{}{A \sqsubseteq B}$ if $A \sqsubseteq B \in \mathcal{T}$

CR4 $\dfrac{A_1 \sqsubseteq A_2 \quad A_2 \sqsubseteq A_3}{A_1 \sqsubseteq A_3}$ CR5 $\dfrac{A \sqsubseteq A_1 \quad A \sqsubseteq A_2 \quad A_1 \sqcap A_2 \sqsubseteq B}{A \sqsubseteq B}$

CR6 $\dfrac{A \sqsubseteq \exists r.A_1 \quad A_1 \sqsubseteq B_1 \quad \exists r.B_1 \sqsubseteq B}{A \sqsubseteq B}$

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

Dresden

Proofs

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

CR1 $\dfrac{}{A \sqsubseteq A}$ CR2 $\dfrac{}{A \sqsubseteq \top}$ CR3 $\dfrac{}{A \sqsubseteq B}$ if $A \sqsubseteq B \in \mathcal{T}$

CR4 $\dfrac{A_1 \sqsubseteq A_2 \quad A_2 \sqsubseteq A_3}{A_1 \sqsubseteq A_3}$ CR5 $\dfrac{A \sqsubseteq A_1 \quad A \sqsubseteq A_2 \quad A_1 \sqcap A_2 \sqsubseteq B}{A \sqsubseteq B}$

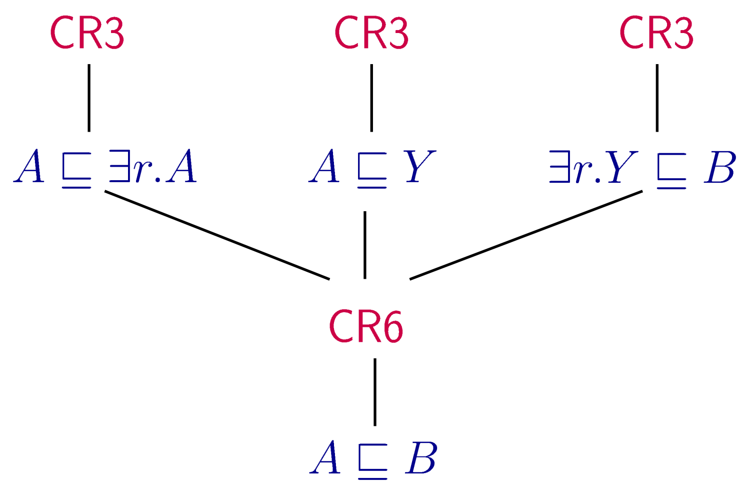
CR6 $\dfrac{A \sqsubseteq \exists r.A_1 \quad A_1 \sqsubseteq B_1 \quad \exists r.B_1 \sqsubseteq B}{A \sqsubseteq B}$

$\mathcal{T}$

a_1 : $A \sqsubseteq \exists r.A$

a_2 : $A \sqsubseteq Y$

a_3 : $\exists r.Y \sqsubseteq B$



Dresden

Proofs to explain DL entailment

Scientific challenges

- What are "good proofs" for explanation purpose, depending on 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

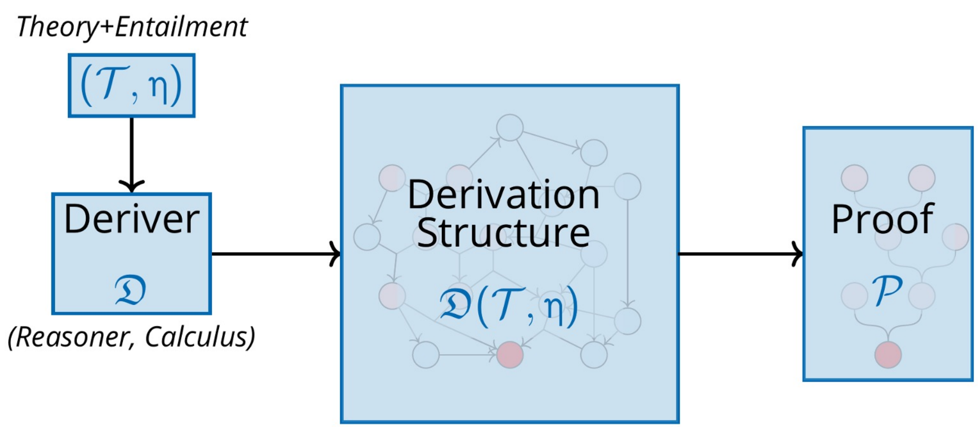
  [Méndez et al.; 2023]

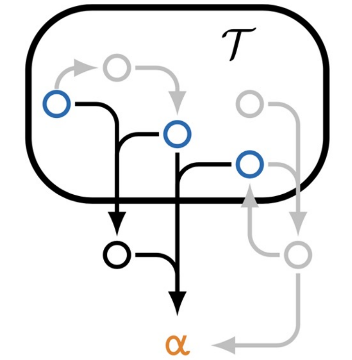
Proofs

complexity of computing good ones

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

Formal framework





- 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

Dresden

Proofs

complexity of computing good ones

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

	polynomial unary	polynomial binary	exponential unary	exponential binary
Size	NP	NP	NP	NExpTime
Monotone recursive Φ-measures	$\leq$ P	$\leq$ P	$\leq$ ExpTime	$\leq$ ExpTime
Tree size	P	P	NP	PSpace
Depth	P	P	PSpace	ExpTime
Logarithmic depth	P	P	ExpTime	ExpTime

- 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

Dresden

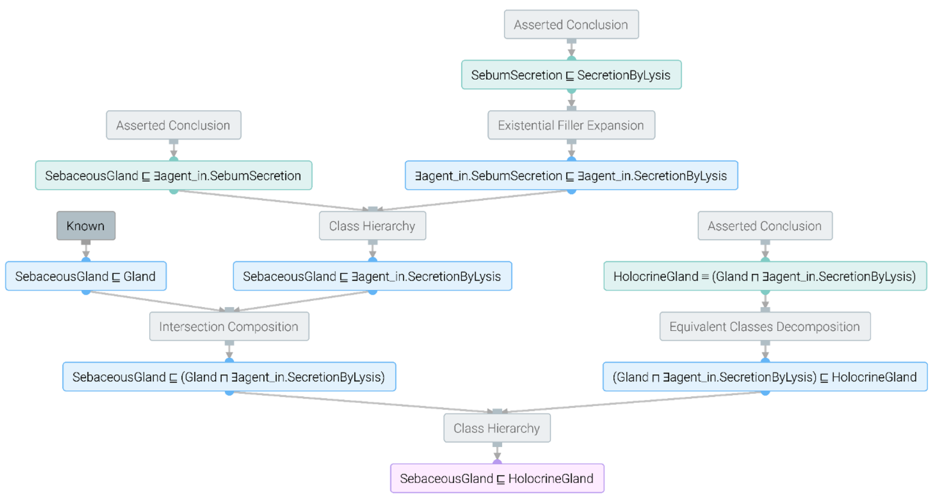
Proofs in Protégé

using our proof plugin Evee

protégé

and in our interactive visualisation tool

Evonne



Dresden

Classical Repair

remove the source of the consequence

subset of the ontology that does not have the unintended consequence

Given: ontology $\mathcal{O}$ and GCI or concept assertion α such that $\mathcal{O} \models \alpha$

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

Optimal classical repair: maximal subset $R \subseteq \mathcal{O}$ 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 α.
2. Compute a minimal hitting set H of $J_1, \ldots, J_k$.
3. Output $R := \mathcal{O} \setminus H$.

$H \cap J_i \neq \emptyset$ for $i = 1, \ldots, k$.

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

Dresden

Classical Repair

remove the source of the consequence

$$\left.\begin{array}{ll} a_1: & A \sqsubseteq \exists r.A \\ a_2: & A \sqsubseteq Y \\ a_3: & \exists r.Y \sqsubseteq B \\ a_4: & Y \sqsubseteq B \end{array}\right\} \models \alpha = A \sqsubseteq B$$

Justifications:

$\{a_2, a_4\}, \ \{a_1, a_2, a_3\}$

Minimal hitting sets:

$\{a_2\}, \ \{a_1, a_4\}, \ \{a_3, a_4\}$

Diagnoses

Optimal classical repairs:

$\{a_1, a_3, a_4\}, \ \{a_2, a_3\}, \ \{a_1, a_2\}$

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

1. Compute all justifications $J_1, \ldots, J_k$ of α.
2. Compute a minimal hitting set H of $J_1, \ldots, J_k$.
3. Output $R := \mathcal{O} \setminus H$.

$H \cap J_i \neq \emptyset$ for $i = 1, \ldots, k$.

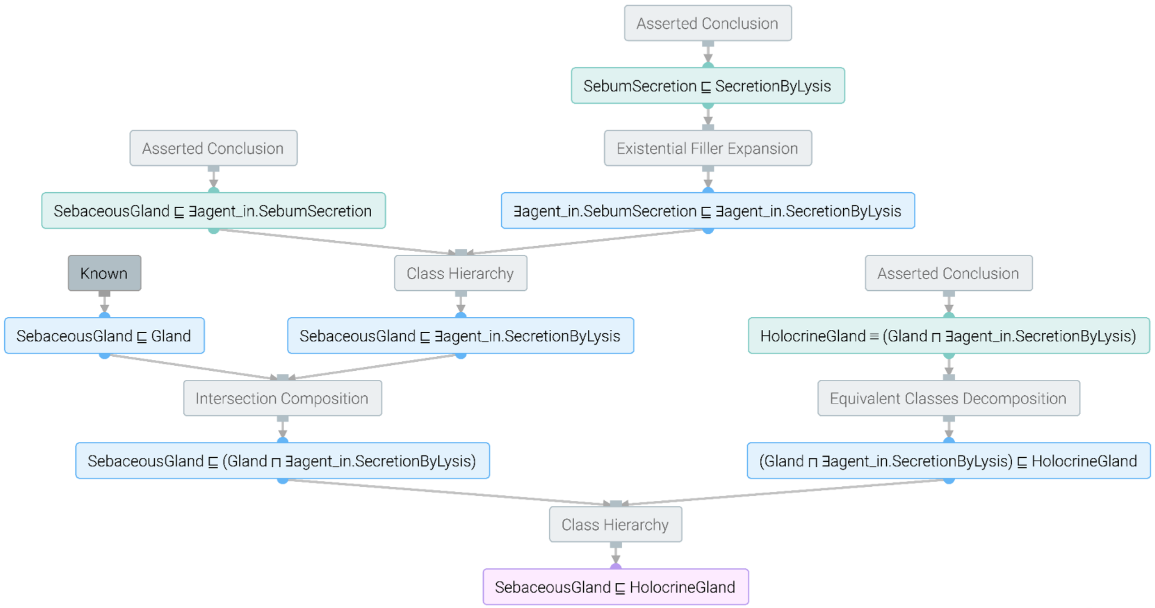
For every choice of the hitting set, this algorithm produces an optimal classical repair, and all optimal classical repairs can be generated this way.

Dresden

Repairs

in our interactive visualisation tool

Evonne



Dresden

Classical Repair

may remove too many consequences

$$\mathcal{T} = \{\exists owns.(GermanCar \sqcap Diesel) \sqsubseteq \exists gets.Compensation\}$$
$$\mathcal{A} = \{\exists owns.(GermanCar \sqcap Diesel)(\texttt{Robert})\}$$
$$\alpha = \exists gets.Compensation(\texttt{Robert})$$

Classical repair: remove $\exists owns.(GermanCar \sqcap Diesel)(\texttt{Robert})$

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

Even more gentle: replace the assertion with

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

Main idea to get better repairs:

consider inclusion for consequences instead of inclusion for axioms

$$Con(\mathcal{O}) := \{\alpha \mid \mathcal{O} \models \alpha\}$$

Dresden

Optimal Repair

- The ontology $\mathcal{R}$ is a repair of $\mathcal{O}$ w.r.t. α if

$$\mathit{Con}(\mathcal{R}) \subseteq \mathit{Con}(\mathcal{O}) \setminus \{\alpha\}.$$

- This repair is optimal if there is no repair $\mathcal{R}'$ of $\mathcal{O}$ w.r.t. α with

$$\mathit{Con}(\mathcal{R}) \subset \mathit{Con}(\mathcal{R}').$$

Example: replacing $\exists owns.(GermanCar \sqcap Diesel)(\texttt{Robert})$ with

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

yields an optimal repair.

Dresden

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 $\exists\ell.(V \sqcap (\exists\ell.)^k.\top)(n)$ belongst to $Con(\mathcal{O})$.

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

scientific challenges

- 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.
- Franz Baader, Boontawee Suntisrivaraporn: Debugging SNOMED CT Using Axiom Pinpointing in the Description Logic $\mathcal{EL}^+$. KR-MED 2008, CEUR-WS 410.
- Rafael Pealoza, Baris Sertkaya: Understanding the complexity of axiom pinpointing in lightweight description logics. Artif. Intell. 250: 80-104 (2017).

Dresden

References

Proofs

- Franz Baader, Ian Horrocks, Carsten Lutz, Ulrike Sattler: An Introduction to Description Logic. Cambridge University Press 2017, pp. 1-255.
- Christian Alrabbaa, Stefan Borgwardt, Anke Hirsch, Nina Knieriemen, Alisa Kovtunova, Anna Milena Rothermel, Frederik Wiehr: In the Head of the Beholder: Comparing Different Proof Representations. RuleML+RR 2022, LNCS 13752: 211-226.
- Christian Alrabbaa, Franz Baader, Stefan Borgwardt, Patrick Koopmann, Alisa Kovtunova: Finding Small Proofs for Description Logic Entailments: Theory and Practice. LPAR 2020, EPiC 73: 32-67.
- Christian Alrabbaa, Franz Baader, Stefan Borgwardt, Patrick Koopmann, Alisa Kovtunova: Finding Good Proofs for Description Logic Entailments using Recursive Quality Measures. CADE 2021, LNCS 12699: 291-308.
- Julian Mendez, Christian Alrabbaa, Patrick Koopmann, Ricardo Langner, Franz Baader, Raimund Dachselt: Evonne: A Visual Tool for Explaining Reasoning with OWL Ontologies and Supporting Interactive Debugging. Computer Graphics Forum (2023). To appear.

References

Repair

- Raymond Reiter: A Theory of Diagnosis from First Principles. Artif. Intell. 32(1): 57-95 (1987).
- Franz Baader, Francesco Kriegel, Adrian Nuradiansyah, Rafael Pealoza: Computing Compliant Anonymisations of Quantified ABoxes w.r.t. $\mathcal{EL}$ Policies. ISWC (1) 2020, LNCS 12506: 3-20.
- Franz Baader, Patrick Koopmann, Francesco Kriegel, Adrian Nuradiansyah: Computing Optimal Repairs of Quantified ABoxes w.r.t. Static $\mathcal{EL}$ TBoxes. CADE 2021, LNCS 12699: 309-326

Dresden

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/