Repairing $\mathcal{EL}_\bot$ Ontologies Using Debugging, Weakening and Completing

Ying Li [a,b] and Patrick Lambrix [a,b,*]

[a] *Department of Computer and Information Science, Linköping University, Sweden*
E-mails: ying.li@liu.se, patrick.lambrix@liu.se
[b] *The Swedish e-Science Research Centre, Sweden*
E-mails: ying.li@liu.se, patrick.lambrix@liu.se

Abstract. The quality of ontologies in terms of their correctness and completeness is crucial for developing high-quality ontology-based applications. Traditional debugging techniques repair ontologies by removing unwanted axioms, but may thereby remove consequences that are correct in the domain of the ontology. In this paper we propose an interactive approach to mitigate this for $\mathcal{EL}_\bot$ ontologies by axiom weakening and completing. We present the first approach for repairing that takes into account debugging, removing, weakening and completing. We show different combination strategies, discuss the influence on the final ontologies and show experimental results. We show that previous work has only considered special cases, and that there is a trade-off, and how to deal with it, involving the amount of validation work for a domain expert and the quality of the ontology in terms of correctness and completeness. We also present new algorithms for weakening and completing. [1]

Keywords: Ontology engineering, Ontology repair, Ontology debugging

1. Introduction

Debugging ontologies aims to remove unwanted knowledge in the ontology. This can be knowledge that leads to logical problems such as inconsistency or incoherence (semantic defects) or statements that are not correct in the domain of the ontology (modeling defects) (e.g., [2]).

The workflow for dealing with unwanted axioms in ontologies or networks consists of several steps including the detection and localization of the defects and the repairing. There are many kinds of approaches to detect and localize defects in ontologies and many of these are complementary to each other, i.e., they detect different kinds of defects. For a short overview we refer to [3]. In this paper we assume we have detected defects and we now need to repair the ontologies. In the classical approaches (e.g., [2, 4–18]) the end result is a set of axioms to remove from the ontology that is obtained after detection and localization, and the repairing consists solely of removing the suggested axioms. However, first, these approaches are usually purely logic-based and therefore may remove correct axioms (e.g., [19]). Therefore, it is argued that a domain expert should validate the results of such systems. Furthermore,

*Corresponding author. E-mail: patrick.lambrix@liu.se.

[1] This paper is an extended version of [1]. In this paper we deal with ontologies represented in $\mathcal{EL}_\bot$ while $\mathcal{EL}$ was used in [1]. We have added full debugging (in contrast to removing in [1]), and the input can be wrong axioms, whereas in [1] the input needed to be wrong *asserted* axioms. Further, we added theory and implementation regarding the combination of debugging, weakening and completing as well as new experiments.

removing an axiom may remove more knowledge than necessary. Correct knowledge that is derivable with the help of the wrong axioms may not be derivable in the new ontology. In this paper we mitigate these effects of removing wrong axioms by, in addition to removing those axioms, also adding correct knowledge. Two approaches could be used. A first approach is to replace a wrong axiom with a weakened version of the axiom (e.g., [20–24]). Another approach is to complete[2] an ontology (e.g., [31–34]) which adds previously unknown correct axioms that allow to derive existing axioms, and that could be used on the results of weakening. These approaches have, however, not been studied together.

In this paper we focus on $\mathcal{EL}_\bot$ ontologies. $\mathcal{EL}_\bot$ is a fragment of the description logic $\mathcal{EL}^{++}$, which extends the description logic $\mathcal{EL}$ with bottom ($\bot$). $\mathcal{EL}$ and its extension $\mathcal{EL}^{++}$ are used by well-known ontologies such as SNOMED or Gene Ontology [35], for which the subsumption checking remains tractable.

Our main contribution (i) is about *choices* that are to be made when using and combining basic operations such as debugging, weakening and completing to repair ontologies, and about the fact that these choices influence the quality of the repaired ontologies in terms of correctness and completeness. We define a framework for repairing ontologies that allows us to investigate these choices. It is the first work that combines debugging and removing with weakening and completing. For this framework we give a formal definition of the repairing problem (Section 3), introduce basic operations (Section 4), and introduce different operations for combining debugging, removing, weakening and completing approaches, and their relationships (Section 5). Using the relationships between these operations, we show that different solutions to the repairing problem exist even using the same basic weakening and completing algorithms (an insight that no other work has discussed). We show that there is a trade-off involving completeness and correctness of the resulting ontologies, and the validation effort of a domain expert (another insight that no other work has discussed). Further, we show how basic algorithms can be combined according to a preference for the level of completeness and correctness. Earlier work on weakening and earlier work on completing can be represented using our operators and their particular weakening and completing algorithms. Using the framework we can show that different approaches for debugging made different choices. Earlier work on weakening used one particular combination strategy (although with different weakening algorithms by different authors). Similarly, work on completing used one particular combination strategy. Our work shows thus that there are actually different variants of the earlier work by combining their basic algorithms in different ways, with trade-offs involving completeness, correctness and validation work. By using our framework together with existing algorithms for debugging, weakening and completing, we essentially provide a blueprint for extending previous work and systems.

In addition to the formal framework there are also other contributions. (ii) We show the trade-offs for different combination strategies for 6 ontologies in experiments (Section 6). Further, in Section 4 (iii) we develop a new algorithm for weakening and a new algorithm for completing. For efficiency reasons, weakening algorithms restrict the search space, and we propose a new heuristic for this restriction. Our algorithm for completing is an extension of the approach in [31]. Finally, (iv) we provide two implemented systems, a Protégé plugin and a stand-alone system (Section 7).

We introduce preliminaries for our work in Section 2 and related work in Section 8. The paper concludes in Section 9.

2. Preliminaries

In this paper we assume that ontologies are represented using a description logic TBox. Description logics [36] are knowledge representation languages where concept descriptions are constructed inductively from a set N_C of atomic concepts and a set N_R of atomic roles and (possibly) a set N_I of individual names. Different description logics allow for different constructors for defining complex concepts and roles. An interpretation $\mathcal{I}$ consists of a non-empty

[2]The term 'completing' has been used with different meanings. In this paper we refer to completing as the dual task of weakening. The term has been used with other meanings. For instance, in [25, 26] a knowledge base is complete if, for a given set of "interesting" concepts, it contains the relevant knowledge about implications between these concepts, and if a knowledge base is not complete in this sense, completing means extending the knowledge to make it so. Related terms are, e.g., ontology extension [27], ontology learning [28], ontology enrichment [29], and ontology revision [30].

set $\Delta^{\mathcal{I}}$ and an interpretation function $\cdot^{\mathcal{I}}$ which assigns to each atomic concept $P \in N_C$ a subset $P^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, to each atomic role $r \in N_R$ a relation $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, and to each individual name[3] $i \in N_I$ an element $i^{\mathcal{I}} \in \Delta^{\mathcal{I}}$. The interpretation function is straightforwardly extended to complex concepts. A TBox is a finite set of axioms which in $\mathcal{EL}_\bot$ are *general concept inclusions* (GCIs). The syntax and semantics for $\mathcal{EL}_\bot$ are shown in Table 2.

Table 1

$\mathcal{EL}_\bot$ syntax and semantics. (Note that P and Q are arbitrary concepts. In the remainder we often use P and Q for atomic concepts.)

Name	Syntax	Semantics
top	$\top$	$\Delta^{\mathcal{I}}$
bottom	$\bot$	$\emptyset$
conjunction	$P \sqcap Q$	$P^{\mathcal{I}} \cap Q^{\mathcal{I}}$
existential restriction	$\exists r.P$	$\{x \in \Delta^{\mathcal{I}} \mid \exists y \in \Delta^{\mathcal{I}} : (x,y) \in r^{\mathcal{I}} \wedge y \in P^{\mathcal{I}}\}$
GCI	$P \sqsubseteq Q$	$P^{\mathcal{I}} \subseteq Q^{\mathcal{I}}$

An interpretation $\mathcal{I}$ is a *model* of a TBox $\mathcal{T}$ if for each GCI in $\mathcal{T}$, the semantic conditions are satisfied. We say that a TBox $\mathcal{T}$ is *inconsistent* if there is no model for $\mathcal{T}$. Further, a concept P in a TBox $\mathcal{T}$ is *unsatisfiable* if for all models $\mathcal{I}$ of $\mathcal{T}$: $P^{\mathcal{I}} = \emptyset$. We say that a TBox is *incoherent* if it contains an unsatisfiable concept.

One of the main reasoning tasks for description logics is subsumption checking[4] in which the problem is to decide for a TBox $\mathcal{T}$ and concepts P and Q whether $\mathcal{T} \models P \sqsubseteq Q$, i.e., whether $P^{\mathcal{I}} \subseteq Q^{\mathcal{I}}$ for every model $\mathcal{I}$ of TBox $\mathcal{T}$. In this paper we update the TBox during the repairing and we always use subsumption with respect to the current TBox.

3. Problem Formulation

We can now formally define the repairing problem that we want to solve (Definition 1). We are given an ontology represented by a TBox $\mathcal{T}$ and a set of wrong axioms W. Further, to guarantee a high level of quality of the ontology (i.e., the aim is that as few correct information as possible is removed and as few incorrect information as possible is added), domain expert validation is a necessity (e.g., [19]). Therefore, we assume an oracle (representing a domain expert) that, when given an axiom, can answer whether this axiom is correct or wrong in the domain of interest of the ontology. We have not required specific properties regarding the performance of the oracle. For instance, we did not require that an oracle always answers correctly or that the oracle gives consistent answers. As a first step we have chosen this way as it reflects reality. According to our long experience working with domain experts in ontology engineering, domain experts make mistakes. However, this does not necessarily mean that domain expert validation is not useful. In experiments in ontology alignment, it was shown that oracles making up to 30% mistakes were still beneficial (e.g., [37]). Further, requiring consistent answers seems to be a tough requirement for domain experts. This would require the ability to reason with long proof chains, while humans usually do well for chains of limited length. It is also not clear how to check that a particular domain expert would fulfil the required properties. Therefore, in this work we do not require such properties, but provide user support in our systems by providing warnings when incompatible validations are made and then allow the domain expert to revise the validations. We do acknowledge, however, that requiring such properties and thereby classifying types of domain experts (e.g., [3, 38]), may allow us to guarantee certain properties regarding correctness and completeness and allow us to reduce the search space of possible repairs.

A repair (A, D) for the ontology given the TBox $\mathcal{T}$, oracle Or, and a set of wrong axioms W, is a tuple containing two sets: a set A of axioms that are correct according to the oracle and should be added to the TBox, and a set D

[3] As we do not deal with individuals in this paper, we do not use individuals in the later sections.

[4] Note that unsatisfiability checking in $\mathcal{EL}_\bot$ can be be performed using subsumption checking. A concept P is unsatisfiable if $P \sqsubseteq \bot$. Further, we can express that two concepts P and Q are disjoint by requiring that $P \sqcap Q \sqsubseteq \bot$.

of asserted axioms that are not correct according to the oracle and should be removed from the TBox. We require that when the axioms in D are removed and the axioms in A are added, the wrong axioms in W cannot be derived anymore.

Definition 1. (Repair)[5] Let $\mathcal{T}$ be a TBox. Let Or be an oracle that given a TBox axiom returns true or false. Let W be a finite set of TBox axioms in $\mathcal{T}$ such that $\forall\, \psi \in W$: $Or(\psi)$ = false.

Then, a repair for Debug-Problem DP($\mathcal{T}$, Or, W) is a tuple (A, D) where A and D are finite sets of TBox axioms such that
(i) $\forall\, \psi \in A$: $Or(\psi)$ = true;
(ii) D is a finite set of *asserted* axioms in $\mathcal{T}$;
(iii) $\forall\, \psi \in D$: $Or(\psi)$ = false;
(iv) $\forall\, \psi \in W$: $(\mathcal{T} \cup A) \setminus D \not\models \psi$.

Our aim is to find repairs that remove as much wrong knowledge and add as much correct knowledge to our ontology as possible. Therefore, we introduce the preference relations *less incorrect* and *more complete* between ontologies (Definitions 2 and 3) that formalize these intuitions, respectively.

Definition 2. (less incorrect - ontologies) Let $\mathcal{O}_1$ and $\mathcal{O}_2$ be two ontologies represented by TBoxes $\mathcal{T}_1$ and $\mathcal{T}_2$ respectively.
Then, $\mathcal{O}_1$ is *less incorrect* than $\mathcal{O}_2$ ($\mathcal{O}_2$ is *more incorrect* than $\mathcal{O}_1$) iff $(\forall \psi : (\mathcal{T}_1 \models \psi \wedge Or(\psi) = false) \rightarrow \mathcal{T}_2 \models \psi)) \wedge (\exists \psi : Or(\psi) = false \ \wedge \mathcal{T}_1 \not\models \psi \wedge \mathcal{T}_2 \models \psi)$.
$\mathcal{O}_1$ and $\mathcal{O}_2$ are *equally incorrect* iff $\forall \psi : Or(\psi) = false \rightarrow (\mathcal{T}_1 \models \psi \leftrightarrow \mathcal{T}_2 \models \psi)$.

Definition 3. (more complete - ontologies) Let $\mathcal{O}_1$ and $\mathcal{O}_2$ be two ontologies represented by TBoxes $\mathcal{T}_1$ and $\mathcal{T}_2$ respectively.
Then, $\mathcal{O}_1$ is *more complete* than $\mathcal{O}_2$ (or $\mathcal{O}_2$ is *less complete* than $\mathcal{O}_1$) iff $(\forall \psi : (\mathcal{T}_2 \models \psi \wedge \ Or(\psi) = true) \rightarrow \mathcal{T}_1 \models \psi)) \wedge (\exists \psi : Or(\psi) = true \ \wedge \mathcal{T}_1 \models \psi \wedge \mathcal{T}_2 \not\models \psi)$.
$\mathcal{O}_1$ and $\mathcal{O}_2$ are *equally complete* iff $\forall \psi : Or(\psi) = true \rightarrow (\mathcal{T}_1 \models \psi \leftrightarrow \mathcal{T}_2 \models \psi)$.

4. Basic operations - debugging, removing, weakening and completing

Our main contribution is a framework that combines debugging, removing, weakening and completing. In this section we describe these basic operations. Further, different algorithms can be used for each of these different operations. We describe the algorithms for debugging, removing, weakening and completing that we use in our experiments in Section 6. Our results in Section 5 regarding the combinations of algorithms also hold when other algorithms are used.

4.1. Preliminaries for the algorithms

For our algorithms we assume that ontologies are represented by *normalized* $\mathcal{EL}_\bot$ TBoxes. A normalized $\mathcal{EL}_\bot$ TBox $\mathcal{T}$ contains only axioms of the forms $P \sqsubseteq Q$, $P \sqcap Q \sqsubseteq R$, $\exists r.P \sqsubseteq Q$ and $P \sqsubseteq \exists r.Q$ where P, Q, $R \in N_C$ and $r \in N_R$. Every $\mathcal{EL}_\bot$ TBox can in linear time be transformed into a normalized TBox that is a conservative extension, i.e., every model of the normalized TBox is also a model of the original TBox and every model of the original TBox can be extended to a model of the normalized TBox [35]. An algorithm for normalizing $\mathcal{EL}_\bot$ TBoxes is given in the appendix.

Further, we define the *simple complex concept set for a TBox* $\mathcal{T}$, which contains all atomic concepts in the ontology as well as the concepts that can be constructed by using one constructor ($\sqcap$ or $\exists$) and only atomic concepts and roles in the ontology (Definition 4). Note that $\top$ and $\bot$ are not in SCC($\mathcal{T}$).[6] Further, if the number of concepts in $N_C^{\mathcal{T}}$ is n and the number of roles in $N_R^{\mathcal{T}}$ is t, then the number of concepts in SCC($\mathcal{T}$) is $(n^2 + n)/2 + tn$.[7]

[5] This is a simplified version of the definition of repair in [3] where, in addition to a set W of wrong axioms to remove, also a set M of correct axioms to add is given as input.

[6] We discuss the consequences of this choice when we introduce the basic algorithms in sections 4.4 and 4.5.

[7] n atomic concepts, $(n^2 - n)/2$ concepts of the form $P \sqcap Q$, and tn concepts of the form $\exists r.P$.

Definition 4. For a normalized $\mathcal{EL}_\bot$ TBox $\mathcal{T}$ with $N_C^\mathcal{T}$ the set of atomic concepts occurring in $\mathcal{T}$ and $N_R^\mathcal{T}$ the set of atomic roles occurring in $\mathcal{T}$, we define the *simple complex concept set for* $\mathcal{T}$, denoted by SCC($\mathcal{T}$), as the set containing all the concepts of the forms P, $P \sqcap Q$, and $\exists r.P$ where $P, Q \in N_C^\mathcal{T}$ and $r \in N_R^\mathcal{T}$.

In our algorithms we use two basic operations which remove and add axioms to a TBox. The result of *Remove-axioms*($\mathcal{T}$,D) for a TBox $\mathcal{T}$ and a set of axioms D is the TBox $\mathcal{T} \setminus D$. If D contains only wrong axioms (such as W), then the ontology represented by Remove-axioms($\mathcal{T}$,D) is less (if at least one of the removed axioms cannot be derived anymore) or equally incorrect (if all removed axioms can still be derived), as well as less (if some correct axioms cannot be derived anymore by removing the wrong ones) or equally complete (if all correct axioms can still be derived), than the ontology represented by $\mathcal{T}$. The result of *Add-axioms*($\mathcal{T}$,A) for a TBox $\mathcal{T}$ and a set of axioms A is the TBox $\mathcal{T} \cup A$. If A contains only correct axioms then the ontology represented by Add-axioms($\mathcal{T}$,A) is more (if some added axioms were not derivable from the ontology) or equally complete (if all added axioms were derivable from the ontology), as well as more (if some wrong axioms can now be derived by adding the new ones) or equally incorrect (if no new wrong axioms can now be derived by adding the new ones), than the ontology represented by $\mathcal{T}$.

We also need to compute sub-concepts and super-concepts of concepts. However, to reduce the infinite search space of possible axioms to add during weakening and completing, we limit the use of nesting operators while computing sub- and super-concepts.[8] This we do by only considering sub- and super-concepts in the SCC of a TBox (Definition 5). As subsumption checking in $\mathcal{EL}_\bot$ is tractable[9], finding these sub- and super-concepts is tractable.

Definition 5. (super- and sub-concepts in SCC)
$sup(P, \mathcal{T}) = \{\ sp \in \text{SCC}(\mathcal{T}) \mid \mathcal{T} \models P \sqsubseteq sp\ \}$
$sub(P, \mathcal{T}) = \{\ sb \in \text{SCC}(\mathcal{T}) \mid \mathcal{T} \models sb \sqsubseteq P\ \}$

Finally, as we work on normalized $\mathcal{EL}_\bot$ TBoxes, we need to make sure that when adding axioms, these are of one of the forms $P \sqsubseteq Q$, $P \sqcap Q \sqsubseteq R$, $\exists r.P \sqsubseteq Q$ and $P \sqsubseteq \exists r.Q$ where $P, Q, R \in N_C^\mathcal{T}$ and $r \in N_R^\mathcal{T}$. We note that new atomic concepts, not originally in the ontology, may be introduced.

4.2. Debugging

Operation. Given a set of wrong axioms W, *debugging* aims to find a set of wrong asserted axioms D that when the axioms in D are removed from the ontologies, the axioms in W cannot be derived anymore. As an example, in Figure 1 derived wrong axiom A $\sqsubseteq$ C needs to be removed. This can be done by removing asserted axiom A $\sqsubseteq$ B or asserted axiom B $\sqsubseteq$ C. Many approaches have been proposed (e.g., [2, 4–18], overview in [3])[10]. A basic approach is based on the computation of justifications for the wrong axioms and then computing a Hitting set over the set of justifications. A justification for axiom ψ in $\mathcal{T}$ is a set of axioms $\mathcal{T}' \subseteq \mathcal{T}$ such that $\mathcal{T}' \models \psi$ and $\forall \mathcal{T}'' \subsetneq \mathcal{T}' : \mathcal{T}'' \not\models \psi$. A Hitting set for a collection of sets $\mathcal{S}$ is a set $H \subseteq \cup_{S \in \mathcal{S}} S$ such that $\forall S \in \mathcal{S} : H \cap S \neq \emptyset$.

Algorithm in the experiments. Algorithm 1 shows a debugging algorithm for $\mathcal{EL}_\bot$ TBoxes. The function *GenerateJustifications* is used for computing the justifications of the wrong axiom $\alpha \sqsubseteq \beta$. It is based on the black-box algorithm in [9], which uses the reasoner as an entailment checking oracle when computing justifications. Further, the function *Validate_axiom* asks the domain expert to validate the asserted axioms from the generated justifications.

4.3. Removing

Operation. In this paper we call *Removing* the operation of deleting *asserted* axioms from the TBox.

Algorithm in the experiments. Removing axioms in D from the TBox $\mathcal{T}$ is performed by applying *Remove-axioms*($\mathcal{T}$,D) as defined in Section 4.1.

[8] Weaker limitations are possible, but the weaker the restriction, the larger the solution search space and the higher the probability of a less usable practical system.

[9] Subsumption checking in $\mathcal{EL}$++ is tractable [35].

[10] There are other approaches that take ABoxes into account.

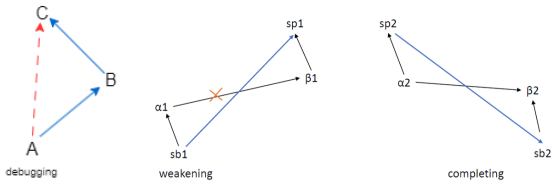


Figure 1. Examples. Debugging: Derived wrong axiom A $\sqsubseteq$ C needs to be removed. This can be done by removing asserted axiom A $\sqsubseteq$ B or asserted axiom B $\sqsubseteq$ C. Weakening: unwanted axiom $\alpha 1 \sqsubseteq \beta 1$ is replaced by correct axiom $sb1 \sqsubseteq sp1$. Completion: wanted axiom $\alpha 2 \sqsubseteq \beta 2$ is replaced by correct axiom $sp2 \sqsubseteq sb2$; $\alpha 2 \sqsubseteq \beta 2$ is still derivable and additional correct axiom $sp2 \sqsubseteq sb2$ is in the repaired ontology.

Algorithm 1 Generate the justifications of the wrong axioms and validate all the asserted axioms from the generated justifications

Input: TBox $\mathcal{T}$, Oracle Or, set of unwanted axioms W
Output: set of asserted wrong axioms D

$D \leftarrow \emptyset$
for each $\alpha \sqsubseteq \beta \in W$ **do**
  Jus$(\alpha \sqsubseteq \beta) \leftarrow GenerateJustifications(\alpha \sqsubseteq \beta)$
  for each $J \in$ Jus$(\alpha \sqsubseteq \beta)$ **do**
    for each $axiom \in J$ **do**
      if $Validate\_axiom(axiom, Or) = false$ **then**
        $D \leftarrow D \cup \{axiom\}$
      end if
    end for
  end for
end for
return D

4.4. Weakening

Operation. Given an axiom, *weakening* aims to find other axioms that are weaker than the given axiom, i.e., the given axiom logically implies the other axioms in the TBox. For an axiom $\alpha \sqsubseteq \beta$, this is often done by replacing α by a more specific concept or replacing β by a more general concept. For the repairing this means that a wrong axiom $\alpha \sqsubseteq \beta$ can be replaced by a correct weaker axiom, thereby mitigating the effect of removing the wrong axiom (Figure 1).

Algorithm in the experiments. Algorithm 2 presents a tractable weakening algorithm for normalized $\mathcal{EL}_\bot$ TBoxes. For a given axiom $\alpha \sqsubseteq \beta$, it finds correct axioms $sb \sqsubseteq sp$ such that sb is a sub-concept in SCC($\mathcal{T}$) of α and sp is a super-concept in SCC($\mathcal{T}$) of β.[11] Further, there should not be another correct axiom under these conditions that would add more correct knowledge to the ontology than $sb \sqsubseteq sp$. As we work with normalized $\mathcal{EL}_\bot$ TBoxes, the new axioms are normalized. The existence of such weaker axioms is not guaranteed.

4.5. Completing

Operation. *Completing* aims to find correct axioms that are not derivable from the ontology yet and that would make a given axiom derivable. It was introduced to aid domain experts when adding axioms to the ontology to find additional knowledge to add. While weakening is usually performed on unwanted axioms, completing is usually

[11] Note that if $\top$ would be allowed in SCC($\mathcal{T}$) then, as it is a super-concept of any other concept, and thus a super-concept of β, we would generate a number of trivial candidates $sb \sqsubseteq \top$ for weakening. Further, if $\bot$ would be allowed in SCC($\mathcal{T}$) then, as it is a sub-concept of any other concept, it is a sub-concept of α and thus there would always be a (trivial) weakened axiom $\bot \sqsubseteq \beta$.

Algorithm 2 Weakened axiom set

Input: TBox $\mathcal{T}$, Oracle Or, unwanted axiom $\alpha \sqsubseteq \beta$
Output: Weakened axiom set of $\alpha \sqsubseteq \beta$

1: $wt_{\alpha\sqsubseteq\beta} \leftarrow \{sb \sqsubseteq sp \mid sb \in sub(\alpha, \mathcal{T}) \wedge sp \in sup(\beta, \mathcal{T}) \wedge \text{Or}(sb \sqsubseteq sp) = \text{True} \wedge \neg\exists\ sb' \in sub(\alpha, \mathcal{T}),\ sp' \in sup(\beta, \mathcal{T})\text{: } (\text{Or}(sb' \sqsubseteq sp') = \text{True} \wedge ((sb \sqsubseteq sb' \wedge sp' \sqsubset sp) \vee (sb \sqsubset sb' \wedge sp' \sqsubseteq sp)))\}$
2: $w_{\alpha\sqsubseteq\beta} \leftarrow \emptyset$
3: **for each** $sb \sqsubseteq sp \in wt_{\alpha\sqsubseteq\beta}$ **do**
4: $\quad w_{\alpha\sqsubseteq\beta} \leftarrow w_{\alpha\sqsubseteq\beta} \cup \text{Normalize}(sb \sqsubseteq sp)$
5: **end for**
6: **return** $w_{\alpha\sqsubseteq\beta}$

performed on wanted axioms (and in this paper on the axioms resulting from the weakening). For an axiom $\alpha \sqsubseteq \beta$, this can be done by replacing α by a more general concept or replacing β by a more specific concept. The axiom $\alpha \sqsubseteq \beta$ can be replaced by a correct stronger axiom, thereby adding additional knowledge to the ontology (Figure 1).

Algorithm in the experiments. Algorithm 3 presents a tractable completion algorithm for normalized $\mathcal{EL}_\bot$ TBoxes. For a given axiom $\alpha \sqsubseteq \beta$, it finds correct axioms $sp \sqsubseteq sb$ such that sp is a super-concept in SCC($\mathcal{T}$) of α and sb is a sub-concept in SCC($\mathcal{T}$) of β (Figure 1).[12] This means that if $sp \sqsubseteq sb$ is added to $\mathcal{T}$, then $\alpha \sqsubseteq \beta$ would be derivable. Further, there should not be another correct axiom under these conditions that would add more correct knowledge to the ontology than $sp \sqsubseteq sb$. Similarly as for weakening, the new axioms are normalized. The completed axiom set is guaranteed to be not empty for a correct axiom $\alpha \sqsubseteq \beta$. It contains $\alpha \sqsubseteq \beta$ or other axioms that lead to the derivation of $\alpha \sqsubseteq \beta$.

Algorithm 3 Completed axiom set

Input: TBox $\mathcal{T}$, Oracle Or, a wanted axiom $\alpha \sqsubseteq \beta$
Output: Completed axiom set of $\alpha \sqsubseteq \beta$

1: $ct_{\alpha\sqsubseteq\beta} \leftarrow \{sp \sqsubseteq sb \mid sp \in sup(\alpha, \mathcal{T}) \wedge sb \in sub(\beta, \mathcal{T}) \wedge \text{Or}(sp \sqsubseteq sb) = \text{True} \wedge \neg\exists\ sp' \in sup(\alpha),\ sb' \in sub(\beta)\text{: } (\text{Or}(sp' \sqsubseteq sb') = \text{True} \wedge (sp \sqsubseteq sp' \wedge sb' \sqsubset sb) \vee (sp \sqsubset sp' \wedge sb' \sqsubseteq sb))\}$
2: $c_{\alpha\sqsubseteq\beta} \leftarrow \emptyset$
3: **for each** $sb \sqsubseteq sp \in ct_{\alpha\sqsubseteq\beta}$ **do**
4: $\quad c_{\alpha\sqsubseteq\beta} \leftarrow c_{\alpha\sqsubseteq\beta} \cup \text{Normalize}(sb \sqsubseteq sp)$
5: **end for**
6: **return** $c_{\alpha\sqsubseteq\beta}$

When the completion is added, we can reduce the amounts of concepts in the completed axiom sets by only showing combinations that would not introduce equivalence between concepts in the ontology. This means that in the implemented version of the completing algorithm, sp should belong to $sup(\alpha, \mathcal{T}) \setminus sup(\beta, \mathcal{T})$ ($sup(\alpha, \mathcal{T})$ in the original algorithm) and sb to $sub(\beta, \mathcal{T}) \setminus sub(\alpha, \mathcal{T})$ ($sub(\beta, \mathcal{T})$ in the original algorithm). These new sets of super- and sub-concepts are called *source* and *target*.

Note that weakening and completing are dual operations where the former finds weaker axioms and the latter stronger axioms. This is reflected in the mirroring of the sub- and super-concepts of α and β in Algorithms 2 and 3.

[12] Note that if $\top$ would be allowed in SCC($\mathcal{T}$) then, as it is a super-concept of any other concept and thus a super-concept of α, we would generate candidate axioms of the form $\top \sqsubseteq sb$, which are unlikely desired as they would make sb equivalent to $\top$. Further, if $\bot$ would be allowed in SCC($\mathcal{T}$) then, as it is a sub-concept of any other concept and thus a sub-concept of β, $\alpha \sqsubseteq \bot$ is a candidate for completing. If the oracle would validate this as correct, then it would mean we would introduce in the ontology that α is an incoherent concept.

5. Combination strategies

In the previous section we defined quality-improving operations for repairing ontologies. Debugging with removing leads to less incorrect (but possibly less complete[13]) ontologies, while weakening and completing lead to more complete (but possibly more incorrect[14]) ontologies. In this section we show different ways to combine these basic operations. Choices need to be made regarding, e.g., processing axioms all at once or one at the time, when to validate generated solutions, and when to update the ontology. We provide building blocks for combination algorithms that show these choices, and show the influence of using different choices on the completeness and correctness of the final ontology.

5.1. Building blocks

To show the trade-off between the choices for different combination strategies regarding completeness, correctness and validation effort, we define operators in Table 2 that can be used as building blocks in the design of algorithms. For each of the basic operations (debugging, removing, weakening and completing) we show choices.

Table 2
Debugging, removing, weakening and completing - operations.

Operations	Description
S-one	Compute the justifications for one wrong axiom at the time.
S-all	Compute the justifications for all wrong axioms at once.
D-one-v	Generate one Hitting set from the justifications, then validate the asserted axioms in the generated Hitting set.
D-v-one	Validate one valid Hitting set in the justifications.
D-all-v/D-v-all	Validate all asserted axioms from the justifications.
R-all	Remove all the wrong axioms at once.
R-one	Remove the wrong axioms one at a time.
R-none	Remove nothing.
W-all	Weaken all wrong axioms at once.
W-one	Weaken the wrong axioms one at a time
C-all	Complete all weakened axioms at once.
C-one	Complete the weakened axioms one at a time.
AB-one	Add one wrong axiom back.
AB-all	Add all wrong axioms back.
AB-none	Add nothing back.
U-now	Update the changes immediately.
U-end_one	Update the changes after the iteration of each wrong axiom.
U-end_all	Update the changes after iterations of all wrong axioms.

During the **debugging** phase, the justifications of the wrong axioms in W are computed. There are different choices to be made regarding generating the justifications of the wrong axioms all at once or one at a time, as well as regarding validating all the axioms in the justifications or just a valid Hitting set, and when to validate. Therefore, we introduce different combination operators. *S-one* and *S-all* represent the choice to calculate the justifications for one wrong axiom at the time or for all at once. The operations with a name starting with *D* concern choices regarding using the justifications to generate asserted wrong axioms to remove from the ontology, where the *one/all* choice concerns whether to validate one Hitting set or all axioms in the justifications. The validation of these axioms can

[13] if some correct axioms cannot be derived anymore after removing the wrong axioms.
[14] if wrong axioms in the ontology are used to derive new wrong axioms together with the newly added correct axioms.

be done during (such that we always have a Hitting set with wrong asserted axioms) or after (which may lead to not repairing) the generation of the Hitting set, represented by the order of *v* (for validation) and *one* (while for the *all* the order does not matter)[15].

After the debugging phase, given a set of asserted wrong axioms, there are different ways to repair the ontology using the removing, weakening and completing operations. These different ways take into account the different choices that can be made in terms of, e.g., the order in which the operations are performed, the order in which the axioms are processed, whether one axiom is dealt with at a time or all at once, and when the TBox is updated. In all cases we need to **remove** the wrong asserted axioms at the end such that the final ontology does not allow deriving the initially given wrong axioms. However, it is possible to remove them immediately after the debugging phase (*R-all*), one at the time (*R-one*), or only at the end (*R-none*). When not removing the wrong asserted axioms immediately, they can still be used during the weakening and completing to produce possibly more candidate weakened or completed axioms, thereby influencing the completeness of the final ontology. Similarly, after having removed axioms, they may or may not be added back during the computations (*AB-all*, *AB-one*, *AB-none*). Added back axioms (which will be removed again at the end of the process) can be used during the weakening and completing, thereby influencing the completeness of the final ontology.

For each of the asserted wrong axioms to remove, we want to find **weakened** axioms. There is a choice to weaken all axioms at once (*W-all*) or one at a time (*W-one*). Once a weakened axiom set is computed, we may choose to add it to the ontology as soon as one is computed (*U-now*)[16], or when all weakened axioms for all wrong axioms are computed (*U-end_all*).[17] When choosing *U-now*, the weakened axioms for one wrong axiom may influence the computation of the weakened axioms for other wrong axioms.

Finally, we can perform **completing** on the weakened axioms. There is a choice to complete all weakened axioms at once (*C-all*) or one at a time (*C-one*). Similarly as for weakening, once completed axiom sets are computed, we may choose to add these to the ontology as soon as one is computed for one particular weakened axiom (*U-now*), when all completed axioms for the weakened axioms of a particular wrong axiom are computed (*U-end_one*), or when all completed axioms for all weakened axioms are computed (*U-end_all*). When choosing *U-now* or *U-end_one*, the completed axioms for one wrong axiom may influence the computation of the completed axioms for other wrong axioms.

As we have already hinted, the different choices influence the completeness, correctness and validation work by a domain expert. We can represent the choices and their influence in Hasse diagrams (Figure 2)[18]. In general, operations higher up in the diagrams use more (but also more possibly wrong) information during the computations and lead to more (or equally) complete ontologies, more (or equally) incorrect ontologies as well as more validation work for the domain expert.[19] For instance, Figure 2b shows that weakening one axiom at a time and immediately updating the TBox (*W-one*, *U-now*) leads to a more complete ontology (and more validation effort) than the other choices. Figure 2c, shows that ontologies repaired by algorithms using one axiom at a time completing and immediate updates (*C-one*, *U-now*) are more complete than ontologies repaired using one axiom at a time completing and updating the ontology after each weakened axiom set for a wrong axiom (*C-one*, *U-end_one*). These ontologies are in turn more complete than for the other choices. In Section 5.2 we prove these statements.

---

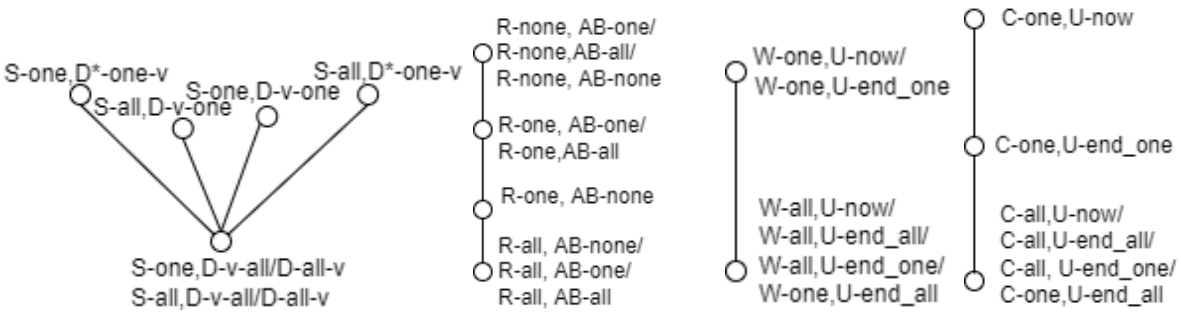
[15] D-v-all: the system outputs the axioms from the justifications one by one and asks the domain expert to validate. D-all-v: the system outputs a list with all axioms from the justifications and then asks the domain expert to validate all axioms in the list. These two options will lead to the same result, i.e., all axioms in the justifications will be validated and the output is the list of all wrong asserted axioms in the justifications.

[16] *U-end_one* represents that a weakened axiom set for a particular wrong axiom is computed and for weakening this is the same operation as *U-now*.

[17] In our combination algorithms, the choice in updating is represented by the use of $\mathcal{T}$ (the original TBox) or $\mathcal{T}_r$ (the current TBox) as TBox.

[18] In the Hasse diagram (Figure 2(a)), D* means the final output is a valid Hitting set.

[19] We note that adding more correct knowledge to an ontology makes the ontology more (or equally) complete, but also more (or equally) incorrect. The latter is because the newly added correct knowledge may, together with existing incorrect knowledge in the ontology, allow the derivation of new incorrect knowledge. However, the root cause is the existing incorrect knowledge and once that knowledge would be removed, also the newly derived incorrect knowledge would be removed. Therefore, this should not be a reason for not adding correct knowledge to the ontology.



(a) Debugging (b) Removing (c) Weakening (d) Completing

Figure 2. Hasse diagrams. (a) selecting and debugging; (b) removing and adding back wrong axioms; (c) weakening and updating; (d) completing and updating.

5.2. Derivation of the Hasse diagrams

We first state general observations that help us in the explanation of the derivation of the Hasse diagrams.

For a given TBox $\mathcal{T}$, let Der($\mathcal{T}$) denote the set of derivable axioms from $\mathcal{T}$. Then, for TBoxes $\mathcal{T}_1$ and $\mathcal{T}_2$, if $\mathcal{T}_1 \sqsubseteq \mathcal{T}_2$, then we know that Der($\mathcal{T}_1$) $\sqsubseteq$ Der($\mathcal{T}_2$). This means that if an axiom is derivable from TBox $\mathcal{T}_1$, it is also derivable from TBox $\mathcal{T}_2$, but not necessarily the other way around. This also means that the ontology $\mathcal{O}_1$ represented by TBox $\mathcal{T}_1$ is less or equally incorrect than the ontology $\mathcal{O}_2$ represented by TBox $\mathcal{T}_2$. Further, $\mathcal{O}_2$ is more or equally complete than $\mathcal{O}_1$.

Given a TBox $\mathcal{T}$ and two sets of wrong asserted axioms D_1 and D_2 such that $D_1 \sqsubseteq \mathcal{T}$, $D_2 \sqsubseteq \mathcal{T}$, and $D_1 \sqsubseteq D_2$, then $\mathcal{T} \setminus D_2 \sqsubseteq \mathcal{T} \setminus D_1$. Given the reasoning above, the ontology where the wrong axioms in D_2 have been removed (represented by $\mathcal{T} \setminus D_2$) is less or equally incorrect than the ontology where the wrong axioms in D_1 have been removed (represented by $\mathcal{T} \setminus D_1$). The latter is more or equally complete than the former.

We also know that, if $\mathcal{T}_1 \sqsubseteq \mathcal{T}_2$, as the sub- and super-concepts of a concept are computed using subsumption axioms, the set of sub-concepts for a concept in $\mathcal{T}_1$ is a subset of the set of sub-concepts for that concept in $\mathcal{T}_2$, and the set of super-concepts for a concept in $\mathcal{T}_1$ is a subset of the set of super-concepts for that concept in $\mathcal{T}_2$. When computing weakened axiom sets and completed axiom sets, the algorithms compute sets of sub-concepts and sets of super-concepts to generate candidate axioms for these weakened and completed axiom sets. Therefore, if $\mathcal{T}_1 \sqsubseteq \mathcal{T}_2$, the sets of candidate axioms for the weakened and completed axiom sets computed for $\mathcal{T}_1$ are subsets of those computed for $\mathcal{T}_2$. This means more validation work for $\mathcal{T}_2$, but also possibly a more complete final ontology.

The Hasse diagrams are based on these observations.

5.2.1. Debugging

When choosing the operations which contain D-all-v or D-v-all, all wrong asserted axioms in the justifications of the given wrong axioms are retained after validation ($W_{S-all,D-all-v/D-v-all} = W_{S-one,D-all-v/D-v-all}$).[20] For the other choices, not all axioms in the justifications are validated and used and thus the set of asserted wrong axioms to remove for each of those choices is a subset of $W_{S-all,D-all-v/D-v-all}$. Note that the Hitting sets computed by the different choices may be different and thus they are siblings in the Hasse diagram.

[20] We consider here the debugging phase separate from the weakening and completing. In the case we would interleave the operations, it is not clear how to compare the incorrectness of the original ontology and the repaired ontology. For instance, if we use S-one, then when removing the axioms in the justifications of a selected wrong axiom from the original ontology (version 1) we obtain a less incorrect ontology (version 2). Then during the weakening and completing steps, new axioms are added making the new version of the ontology (version 3) more complete, but possibly also more incorrect than version 2 (as new wrong axioms may be derivable using wrong axioms still in the ontology in combination with the added axioms). These new axioms may then influence the justifications for the next selected wrong axiom to process, finding more wrong asserted axioms to remove and thus find a less incorrect ontology (version 4) than version 3. Thus, when combining debugging with weakening and completing for S-one, for each original wrong axiom we would make the ontology less incorrect in one way and then more incorrect in another way, but also producing opportunities for additional removal of asserted wrong axioms that may not appear when using S-all.

5.2.2. Removing

In general, when removing all axioms at once, the TBox is a subset of the TBox with one axiom removed, which in turn is a subset of the TBox where no axioms are removed. When adding no axioms back, the TBox is a subset of the TBox with one axiom added back, which in turn is a subset of the TBox where all axioms are added back. If no wrong axioms are removed, then nothing needs to be added back and thus AB-one, AB-all and AB-none have the same result ($\mathcal{T}_{R-none,AB-all} = \mathcal{T}_{R-none,AB-one} = \mathcal{T}_{R-none,AB-none}$). The TBox for these strategies is larger during computation (of weakened or completed axiom sets) than the TBoxes where one or all wrong axioms are removed. If one wrong axiom at the time is removed, the adding back all (AB-all) or one (AB-one) give the same result ($\mathcal{T}_{R-one,AB-all} = \mathcal{T}_{R-one,AB-one}$) as both strategies add the same one axiom back. The TBox for these strategies is larger than when no wrong axiom is added back ($\mathcal{T}_{R-one,AB-none} \sqsubseteq \mathcal{T}_{R-one,AB-all} = \mathcal{T}_{R-one,AB-one}$). When all wrong axioms are removed at once, then they will be added back at the end or not.[21] However, this does not influence the TBox during the computation. Therefore, the add back strategy does not matter and the TBox during computation is smaller than when wrong axioms were removed one at a time ($\mathcal{T}_{R-all,AB-all} = \mathcal{T}_{R-all,AB-one} = \mathcal{T}_{R-all,AB-none} \sqsubseteq \mathcal{T}_{R-one,AB-none}$).

5.2.3. Weakening

First, we note that updating immediately or updating after each wrong axiom is the same operation for weakening, as a complete weakened axiom set for a wrong axiom is computed. Thus, the TBox for ($\mathcal{T}_{W-one,U-now}$) is the same as for ($\mathcal{T}_{W-one,U-end\_one}$), and the TBox for ($\mathcal{T}_{W-all,U-now}$) is the same as for ($\mathcal{T}_{W-all,U-end\_one}$). Further, when weakening one axiom at a time and updating the TBox (i.e., adding the axioms of the weakened axiom set for a wrong axiom) immediately, results in a larger TBox for the next computations of weakened axiom sets for wrong axioms, than if we would not update immediately ($\mathcal{T}_{W-one,U-end\_all} \sqsubseteq \mathcal{T}_{W-one,U-now}$). When not immediately updating, the TBox for generating the weakened axiom sets, stays the same for all wrong axioms and thus gives the same result as weakening all wrong axioms at once. Thus, $\mathcal{T}_{W-all,U-now} = \mathcal{T}_{W-all,U-end\_all} = \mathcal{T}_{W-one,U-end\_all}$.

5.2.4. Completing

When completing one axiom at a time and updating the TBox (i.e., adding the axioms of the completed axiom set for a weakened axiom) immediately, results in a larger TBox for the next computations of completed axiom sets for weakened axioms than not updating immediately ($\mathcal{T}_{C-one,U-end\_one} \sqsubseteq \mathcal{T}_{C-one,U-now}$, $\mathcal{T}_{C-one,U-end\_all} \sqsubseteq \mathcal{T}_{C-one,U-now}$,). When not updating immediately, there is the choice between updating after all weakened axioms for a particular wrong axiom have been processed or waiting until all weakened axioms for all wrong axioms are processed. The TBox for the former case is larger than the one for the latter case ($\mathcal{T}_{C-one,U-end\_all} \sqsubseteq \mathcal{T}_{C-one,U-end\_one}$). Waiting to update the TBox until all weakened axioms for all wrong axioms are processed, means the TBox stays the same during the computation of the completed axioms sets and thus gives the same result as completing all weakened axioms at once ($\mathcal{T}_{C-one,U-end\_all} = \mathcal{T}_{C-all,U-end\_all} = \mathcal{T}_{C-all,U-end\_one} = \mathcal{T}_{C-all,U-now}$).

5.3. Using the building blocks to compare combination algorithms

The combination algorithms can be defined by which of the building blocks are used and in which order. In this section we exemplify how the building blocks can be used to compare different combination algorithms in terms of correctness and completeness of the repaired ontology. In general, if the sequence of building block operators for one algorithm can be transformed into the sequence of operators of a second algorithm, by replacing some operators of the first algorithm using operators higher up in the Hasse diagrams in Figure 2, then the ontologies repaired using the second algorithm are more or equally complete and incorrect than the ontologies repaired using the first algorithm.

We use 4 representative algorithms (while more algorithms are discussed in our experiments in Section 6) to exemplify this. We note that all proposed algorithms are tractable and find repairs as defined in Definition 1.

As an example of the influence of the choices in the debugging part, Algorithm C14 and Algorithm C15 perform the same weakening and completing operations (and thus use the same building blocks for these operations), but

[21] After completing they should be removed, but after weakening they could be added back for the completion step.

Algorithm C14 uses *(S-all,D-v-all)* and Algorithm C15 uses *(S-one,D-v-one)*. Given that the algorithms are the same except for the debugging part, we can use the Hasse diagram in Figure 2(a), and conclude that the ontology repaired by Algorithm C14 is less incorrect (and possibly less complete) than the ontology repaired by Algorithm C15.

As an example using weakening and completing, Algorithm C9 uses weaken one at a time, remove all wrong, complete one at a time, then add completed axiom sets at the end, while Algorithm C10 uses weaken one at a time, remove all wrong, add completed axiom sets one at a time. We can then compare the algorithms using the Hasse diagrams. The sequence of building blocks in Algorithm C9 can be rewritten into the sequence of Algorithm C10 by replacing the completion operator *(C-one,U-end_all)* with the completion operator *(C-one,U-now)*, which is higher up in the Hasse diagram in Figure 2(d). Thus, repairing an ontology using Algorithm C10 leads to a more or equally complete (and possibly more incorrect) ontology than repairing using Algorithm C9.

Algorithm C14 Generate the justifications of each wrong axiom and validate all the asserted axioms in the justifications, weaken all wrong asserted axioms, remove all wrong asserted axioms, complete/add completed axiom set one at a time

---

Input: TBox $\mathcal{T}$, Oracle Or, set of unwanted axioms W
Output: A repaired TBox

$D \leftarrow \emptyset$
for each $\alpha \sqsubseteq \beta \in W$ **do**
  for each $J \in$ Justifications$(\alpha \sqsubseteq \beta)$ **do**
    for each $axiom \in J$ **do**
      if $Validate\_axiom(axiom, Or) = false$ **then**
        $D \leftarrow D \cup \{axiom\}$
      end if
    end for
  end for
end for
for each $\alpha \sqsubseteq \beta \in D$ **do**
  $\mathcal{T}_r \leftarrow$ Remove-axioms($\mathcal{T}$, $\{\alpha \sqsubseteq \beta\}$)
  $w_{\alpha \sqsubseteq \beta} \leftarrow$ weakened-axiom-set($\alpha \sqsubseteq \beta$, $\mathcal{T}_r$, Or)
end for
$\mathcal{T}_r \leftarrow$ Remove-axioms($\mathcal{T}_r$, D)
for each $\alpha \sqsubseteq \beta \in D$ **do**
  for each $sb \sqsubseteq sp \in w_{\alpha \sqsubseteq \beta}$ **do**
    $c_{sb \sqsubseteq sp} \leftarrow$ completed-axiom-set($sb \sqsubseteq sp$,$\mathcal{T}_r$, Or)
    $\mathcal{T}_r \leftarrow$ Add-axioms($\mathcal{T}_r$,$c_{sb \sqsubseteq sp}$)
  end for
end for
return $\mathcal{T}_r$

Algorithm C15 Generate the justifications of each wrong axiom and validate one Hitting set from the generated justifications, weaken all wrong asserted axioms, remove all wrong asserted axioms, complete/add completed axiom set one at a time

Input: TBox $\mathcal{T}$, Oracle Or, set of unwanted axioms W
Output: A repaired TBox

$D \leftarrow \emptyset$
for each $\alpha \sqsubseteq \beta \in W$ **do**
  for each $J \in$ Justifications$(\alpha \sqsubseteq \beta)$ **do**
    for each $axiom \in J$ **do**
      if $Validate\_axiom(axiom, Or) = false$ **then**
        $D \leftarrow D \cup \{axiom\}$
        break
      end if
    end for
  end for
end for
for each $\alpha \sqsubseteq \beta \in D$ **do**
  $\mathcal{T}_r \leftarrow$ Remove-axioms($\mathcal{T}$, $\{\alpha \sqsubseteq \beta\}$)
  $w_{\alpha \sqsubseteq \beta} \leftarrow$ weakened-axiom-set($\alpha \sqsubseteq \beta$, $\mathcal{T}_r$, Or)
end for
$\mathcal{T}_r \leftarrow$ Remove-axioms($\mathcal{T}_r$,D)
for each $\alpha \sqsubseteq \beta \in D$ **do**
  for each $sb \sqsubseteq sp \in w_{\alpha \sqsubseteq \beta}$ **do**
    $c_{sb \sqsubseteq sp} \leftarrow$ completed-axiom-set($sb \sqsubseteq sp$,$\mathcal{T}_r$, Or)
    $\mathcal{T}_r \leftarrow$ Add-axioms($\mathcal{T}_r$,$c_{sb \sqsubseteq sp}$)
  end for
end for
return $\mathcal{T}_r$

Algorithm C9 Weaken all wrong, remove all wrong, complete all weakened axioms and add completed axiom sets at end

Input: TBox $\mathcal{T}$, Oracle Or, set of unwanted axioms W
Output: A repaired TBox

for each $\alpha \sqsubseteq \beta \in W$ **do**
  $\mathcal{T}_r \leftarrow$ Remove-axioms($\mathcal{T}$, $\{\alpha \sqsubseteq \beta\}$)
  $w_{\alpha \sqsubseteq \beta} \leftarrow$ weakened-axiom-set($\alpha \sqsubseteq \beta$, $\mathcal{T}_r$, Or)
end for
$\mathcal{T}_r \leftarrow$ Remove-axioms($\mathcal{T}_r$,W)
for each $\alpha \sqsubseteq \beta \in W$ **do**
  $c_{\alpha \sqsubseteq \beta} \leftarrow \emptyset$
  for each $sb \sqsubseteq sp \in w_{\alpha \sqsubseteq \beta}$ **do**
    $c_{sb \sqsubseteq sp} \leftarrow$ completed-axiom-set($sb \sqsubseteq sp$,$\mathcal{T}_r$, Or)
    $c_{\alpha \sqsubseteq \beta} \leftarrow c_{\alpha \sqsubseteq \beta} \cup c_{sb \sqsubseteq sp}$
  end for
end for
$\mathcal{T}_r \leftarrow$ Add-axioms($\mathcal{T}_r$,$\bigcup_{\alpha \sqsubseteq \beta} c_{\alpha \sqsubseteq \beta}$)
return $\mathcal{T}_r$

Algorithm C10 Weaken all wrong, remove all wrong, complete/add completed axiom sets one at a time

Input: TBox $\mathcal{T}$, Oracle Or, set of unwanted axioms W
Output: A repaired TBox

for each $\alpha \sqsubseteq \beta \in W$ **do**
  $\mathcal{T}_r \leftarrow$ Remove-axioms($\mathcal{T}$, $\{\alpha \sqsubseteq \beta\}$)
  $w_{\alpha \sqsubseteq \beta} \leftarrow$ weakened-axiom-set($\alpha \sqsubseteq \beta$, $\mathcal{T}_r$, Or)
end for
$\mathcal{T}_r \leftarrow$ Remove-axioms($\mathcal{T}$, W)
for each $\alpha \sqsubseteq \beta \in W$ **do**
  for each $sb \sqsubseteq sp \in w_{\alpha \sqsubseteq \beta}$ **do**
    $c_{sb \sqsubseteq sp} \leftarrow$ completed-axiom-set($sb \sqsubseteq sp$,$\mathcal{T}_r$, Or)
    $\mathcal{T}_r \leftarrow$ Add-axioms($\mathcal{T}_r$,$c_{sb \sqsubseteq sp}$)
  end for
end for
return $\mathcal{T}_r$

Algorithm C11 Weaken all wrong, remove/complete/add the wrong/completed axiom sets one at a time, remove all wrong at end

Input: TBox $\mathcal{T}$, Oracle Or, set of unwanted axioms W
Output: A repaired TBox

for each $\alpha \sqsubseteq \beta \in W$ **do**
  $\mathcal{T}_r \leftarrow$ Remove-axioms($\mathcal{T}$, $\{\alpha \sqsubseteq \beta\}$)
  $w_{\alpha \sqsubseteq \beta} \leftarrow$ weakened-axiom-set($\alpha \sqsubseteq \beta$, $\mathcal{T}_r$, Or)
end for
for each $\alpha \sqsubseteq \beta \in W$ **do**
  $\mathcal{T}_r \leftarrow$ Remove-axioms($\mathcal{T}$, $\{\alpha \sqsubseteq \beta\}$)
  for each $sb \sqsubseteq sp \in w_{\alpha \sqsubseteq \beta}$ **do**
    $c_{sb \sqsubseteq sp} \leftarrow$ completed-axiom-set($sb \sqsubseteq sp$, $\mathcal{T}_r$, Or)
    $\mathcal{T} \leftarrow$ Add-axioms($\mathcal{T}$,$c_{sb \sqsubseteq sp}$)
  end for
end for
return Remove-axioms($\mathcal{T}$,W)

Algorithm C12 Remove all wrong, weaken all wrong, complete all weakened axioms and add completed axiom sets at end

Input: TBox $\mathcal{T}$, Oracle Or, set of unwanted axioms W
Output: A repaired TBox

$\mathcal{T}_r \leftarrow$ Remove-axioms($\mathcal{T}$, W)
for each $\alpha \sqsubseteq \beta \in W$ **do**
  $w_{\alpha \sqsubseteq \beta} \leftarrow$ weakened-axiom-set($\alpha \sqsubseteq \beta$, $\mathcal{T}_r$, Or)
end for
for each $\alpha \sqsubseteq \beta \in W$ **do**
  $c_{\alpha \sqsubseteq \beta} \leftarrow \emptyset$
  for each $sb \sqsubseteq sp \in w_{\alpha \sqsubseteq \beta}$ **do**
    $c_{sb \sqsubseteq sp} \leftarrow$ completed-axiom-set($sb \sqsubseteq sp$, $\mathcal{T}_r$, Or)
    $c_{\alpha \sqsubseteq \beta} \leftarrow c_{\alpha \sqsubseteq \beta} \cup c_{sb \sqsubseteq sp}$
  end for
end for
return Add-axioms($\mathcal{T}_r$,$\bigcup_{\alpha \sqsubseteq \beta} c_{\alpha \sqsubseteq \beta}$)

Algorithm C13 Remove all wrong, weaken/complete/add completed axiom sets one at a time

Input: TBox $\mathcal{T}$, Oracle Or, set of unwanted axioms W
Output: A repaired TBox

$\mathcal{T}_r \leftarrow$ Remove-axioms($\mathcal{T}$, W)
for each $\alpha \sqsubseteq \beta \in W$ **do**
    $w_{\alpha \sqsubseteq \beta} \leftarrow$ weakened-axiom-set($\alpha \sqsubseteq \beta$, $\mathcal{T}_r$, Or)
    for each $sb \sqsubseteq sp \in w_{\alpha \sqsubseteq \beta}$ **do**
        $c_{sb \sqsubseteq sp} \leftarrow$ completed-axiom-set($sb \sqsubseteq sp$, $\mathcal{T}_r$, Or)
        $\mathcal{T}_r \leftarrow$ Add-axioms($\mathcal{T}_r$,$c_{sb \sqsubseteq sp}$)
    end for
end for
return $\mathcal{T}_r$

6. Experiments

In addition to the theoretical conclusions about the influence of the different choices in the combination strategies in Section 5, we study the influence by performing practical experiments on several ontologies: Mini-GALEN (used as the running example, Figure 3 and Figure 4), PACO, NCI, OFSMR, EKAW and Pizza ontology. An overview of the numbers of concepts, roles and axioms in these ontologies is given in Table 3. We have used the parts of these ontologies that are expressible in $\mathcal{EL}_\bot$ in the sense that we removed the parts of axioms that used constructors not in $\mathcal{EL}_\bot$. We introduced new axioms in the ontologies by replacing existing axioms with axioms where the left-hand or right-hand side concepts of the existing axioms were changed. Further, we also flagged axioms as wrong in our full experiment set (e.g., in PACO). In the appendix we provide all the wrong axioms as well as the asserted wrong axioms used in the examples for each ontology after debugging.

N_C = {GPr (GranulomaProcess), NPr (NonNormalProcess),
PPh (PathologicalPhenomenon), F(Fracture), E (Endocarditis),
IPr (InflammationProcess), PPr (PathologicalProcess),
C (Carditis), CVD (CardioVascularDisease)};
N_R = { hAPr (hasAssociatedProcess) }
$\mathcal{T}$ = { CVD $\sqsubseteq$ PPh, F $\sqsubseteq$ PPh, $\exists$hAPr.PPr $\sqsubseteq$ PPh, E $\sqsubseteq$ C,
E $\sqsubseteq$ $\exists$hAPr.IPr, GPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ IPr, IPr $\sqsubseteq$ GPr, PPr $\sqsubseteq$ GPr, E $\sqsubseteq$ PPr };
W = { E $\sqsubseteq$ PPr, PPr $\sqsubseteq$ IPr, IPr $\sqsubseteq$ GPr, PPr $\sqsubseteq$ GPr }
Or returns *true* for:
GPr $\sqsubseteq$ IPr, GPr $\sqsubseteq$ PPr, GPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ PPr, IPr $\sqsubseteq$ NPr,
PPr $\sqsubseteq$ NPr, CVD $\sqsubseteq$ PPh, F $\sqsubseteq$ PPh, E $\sqsubseteq$ PPh, E $\sqsubseteq$ C,
E $\sqsubseteq$ CVD, C $\sqsubseteq$ PPh, C $\sqsubseteq$ CVD, $\exists$hAPr.PPr $\sqsubseteq$ PPh,
$\exists$hAPr.IPr $\sqsubseteq$ PPh, E $\sqsubseteq$ $\exists$hAPr.IPr, E $\sqsubseteq$ $\exists$hAPr.PPh.
Note that for an oracle that does not make mistakes,
if *Or*(P $\sqsubseteq$ Q) = true, then also *Or*($\exists$r.P $\sqsubseteq$ $\exists$r.Q)=true and
Or(P $\sqcap$ O $\sqsubseteq$ Q)=true.
For other axioms P $\sqsubseteq$ Q with P, Q $\in$ N_C, *Or*(P $\sqsubseteq$ Q) = false.

Figure 3. Mini-GALEN. (Visualized in Figure4.)

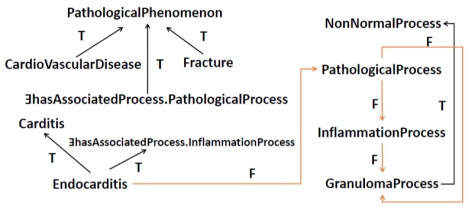


Figure 4. Visualization of the Mini-GALEN ontology in Figure 3. The axioms in the TBox are represented with black arrows except for the wrong axioms which are represented in red. The oracle's knowledge about the axioms in the ontology is marked with T (true) or F (false) at the arrows. Wrong asserted axioms: ①PPr⊑IPr ②IPr⊑GPr ③E⊑PPr ④PPr⊑GPr (can also be derived from ① and ②).

We use the OWL Explanation API[22] to generate the justifications of the wrong axioms in the debugging phase. OWL Explanation is a software library that is part of the OWL Explanation Workbench[23] [39], which is for working with justification-based explanations of entailments in OWL ontologies. Further, for subsumption checking in the

[22]https://mvnrepository.com/artifact/net.sourceforge.owlapi/owlexplanation
[23]https://github.com/matthewhorridge/owlexplanation

Table 3
Ontologies

	Mini-GALEN	Pizza	EKAW	OFSMR	PACO	NCI
Concepts	9	74	100	159	224	3304
Roles	1	33	8	2	23	1
Axioms	20	341	801	1517	1153	30364

algorithms we used HermiT[24]. Here, we give results for Mini-GALEN (Figure 3) which are representative for all experiments (see Tables 4 - 7). Results for the other ontologies are given in the appendix.

6.1. Influence of the strategies on correctness and completeness

As an example of the influence of using different strategies in the debugging step, Table 4 shows how the choices of selecting wrong axioms, when to compute Hitting sets and when to validate axioms, influence which asserted axioms are retained for weakening and completing. As discussed earlier, algorithms C14 and C15 perform the same weakening and completing operations, but Algorithm C14 uses *(S-all,D-v-all)* (with as result for the debugging step the D-v-all column of Table 4) and Algorithm C15 uses *(S-one,D-v-one)* (with as result for the debugging step one of the D-v-one columns of Table 4). Thus, as it should be according to the Hasse diagram for debugging, the ontology after the debugging step of Algorithm C14 is a less (or equally) incorrect ontology than the ontology after the debugging step of Algorithm C15. Indeed, as shown in Table 4, each asserted wrong axiom to be removed for a *D-v-one* choice is also to be removed for *D-v-all*, but for *D-v-all* there is always an additional asserted wrong axiom that is removed.

The influence of the different choices of the removing operation is exemplified by Algorithms C10 and C11. They use the same weakening and completing operations, but Algorithm C10 uses *(R-all,AB-none)* while Algorithm C11 uses the higher-level *(R-one,AB-one)*. According to the Hasse diagram (Figure 2b), the final ontology repaired by Algorithm C11 is expected to be more (or equally) complete than the one repaired by Algorithm C10. Indeed, in Table 6, the completed axiom set generated by Algorithm C11 is a super-set of the one generated by Algorithm C10.

The influence of different choices regarding weakening operation can be exemplified by comparing Algorithm C12 and C13. These two algorithms both remove all the wrong asserted axioms before the weakening step. Algorithm C12 uses the operator at *(W-all,U-end_all)* in Figure 2c. Algorithm C13 uses the higher-level operator *(W-one,U-now)*. Therefore, the ontology after weakening using Algorithm C13 is more or equally complete than the ontology after weakening using Algorithm C12. As it shown in Table 5, when using Algorithm C13, the domain expert needs to validate 6 candidate weakened axioms for Algorithm C12 and 12 for Algorithm C13. The resulting weakened axioms are the same for both algorithms. However, axiom PPr$\sqsubseteq$ NPr appears twice for Algorithm C13, and depending on the combinations used for the completing step, this may influence the final ontology.

The influence regarding the different options for completion operator can be exemplified by comparing Algorithms C9 and C10. The sequence of the building block operators in Algorithm C9 can be rewritten into the sequence of Algorithm C10 by replacing the completion operator *(C-one,U-end_all)* to the higher-level *(C-one,U-now)*. Thus, repairing an ontology using Algorithm C10, should, according to the Hasse diagram in Figure 2d, lead to a more (or equally) complete ontology than repairing using Algorithm C9. In Table 6, after the completing step, IPr $\sqsubseteq$ PPr generated by C10 is a stronger axiom than IPr $\sqsubseteq$ NPr generated by C9.

In Table 7 we show the results for different algorithms when, during the completion step, removing from the sub and sup sets, the concepts that would introduce equivalence relations in the target ontology. By comparing the results in Tables 6 and 7, we note that using sub/sup sets leads to a more (equally) complete ontology than using source/target sets. However, it requires more validation work from the domain expert and possibly undesired equivalences between concepts may be introduced.

[24]http://www.hermit-reasoner.com/

Table 4

Debugging for Mini-GALEN, For D-v-one there are different solutions based on the chosen Hitting set. (Same results for S-one and S-all.)

	D-v-all	D-v-one (1)	D-v-one (2)	D-v-one (3)
Wrong Axioms	PPr $\sqsubseteq$ GPr, E $\sqsubseteq$ GPr	PPr $\sqsubseteq$ GPr, E $\sqsubseteq$ GPr	PPr $\sqsubseteq$ GPr, E $\sqsubseteq$ GPr	PPr $\sqsubseteq$ GPr, E $\sqsubseteq$ GPr
Asserted Wrong Axioms	IPr $\sqsubseteq$ GPr, PPr $\sqsubseteq$ IPr, PPr $\sqsubseteq$ GPr, E $\sqsubseteq$ PPr	IPr $\sqsubseteq$ GPr, PPr $\sqsubseteq$ IPr, PPr $\sqsubseteq$ GPr	PPr $\sqsubseteq$ IPr, PPr $\sqsubseteq$ GPr, E $\sqsubseteq$ PPr	IPr $\sqsubseteq$ GPr, PPr $\sqsubseteq$ GPr, E $\sqsubseteq$ PPr

Table 5

Weakening for Mini-GALEN using Algorithms C10-C15. Four wrong asserted axioms give 4 sup/sub-sets respectively per algorithm. The wrong asserted axioms are IPr $\sqsubseteq$ GPr, PPr $\sqsubseteq$ IPr, E $\sqsubseteq$PPr, PPr $\sqsubseteq$ GPr. Note: when a weakened axiom was generated several times, we list it that many times (PPr $\sqsubseteq$ NPr is listed twice for C10, C11, and C13).

	C10	C11	C12	C13
Wrong asserted axioms	PPr $\sqsubseteq$ GPr, IPr $\sqsubseteq$ GPr, PPr $\sqsubseteq$ IPr, E $\sqsubseteq$ PPr	PPr $\sqsubseteq$ GPr, IPr $\sqsubseteq$ GPr, PPr $\sqsubseteq$ IPr, E $\sqsubseteq$ PPr	PPr $\sqsubseteq$ GPr, IPr $\sqsubseteq$ GPr, PPr $\sqsubseteq$ IPr, E $\sqsubseteq$ PPr	PPr $\sqsubseteq$ GPr, IPr $\sqsubseteq$ GPr, PPr $\sqsubseteq$ IPr, E $\sqsubseteq$ PPr
Sub(α,$\mathcal{T}$) Sup(β,$\mathcal{T}$)	2 3 2 1 2 2 3 4	2 3 2 1 2 2 3 4	1 1 1 1 2 2 1 1	1 1 2 1 2 2 3 2
Weakened	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr	IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr

Table 6

Completing the Mini-GALEN ontology using Algorithms C8-C13.

	C8	C9	C10	C11	C12	C13
Weakened	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr
Sup(α,$\mathcal{T}$) Sub(β,$\mathcal{T}$)	4 3 4 5 5 5	1 1 1 2 2 2	1 1 2 2 3 4	4 3 3 5 4 5	1 1 2 2	1 1 2 2 3 3
Completed	GPr $\sqsubseteq$ IPr, PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ PPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ PPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ PPr, GPr $\sqsubseteq$ IPr	IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr	IPr $\sqsubseteq$ PPr, PPr $\sqsubseteq$ NPr

Table 7

Completing the Mini-GALEN ontology using Algorithms C8-C13 by excluding concepts that would introduce equivalence relations in the ontology.

	C8	C9	C10	C11	C12	C13
Weakened	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr
Source Target	3 2 3 3 2 3	1 1 1 2 2 2	1 1 1 2 3 2	3 1 2 3 1 3	1 1 2 2	1 1 1 2 3 2
Completed	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr GPr $\sqsubseteq$ IPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ PPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr, GPr $\sqsubseteq$ IPr	IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr	IPr $\sqsubseteq$ PPr, PPr $\sqsubseteq$ NPr

6.2. Validation work

For an unwanted axiom $\alpha \sqsubseteq \beta$, the domain experts need to validate the asserted axioms in the justifications of $\alpha \sqsubseteq \beta$. Given a wrong asserted axiom $\alpha_a \sqsubseteq \beta_a$, during the weakening step, the maximum number of candidate weakened axioms that need to be validated is $|sub(\alpha_a, \mathcal{T})| * |sup_(\beta_a, \mathcal{T})|$. During the completion step, the maximum number of generated candidate completed axioms is $|sup(w_\alpha, \mathcal{T})| * |sub(w_\beta, \mathcal{T})|$ for a given weakened axiom $w_\alpha \sqsubseteq w_\beta$.

There is a trade-off between validation effort and the level of completeness. In general, a more complete ontology usually requires more validation work. The amount of the validation work may vary. In our experiments using the Mini-GALEN ontology, for instance, in the completion step, the amount of validation work required for Algorithm C13 was 3 times higher than the effort required by Algorithm C12, but repairing the Mini-Galen using Algorithm C13 leads to a more complete ontology than repairing using Algorithm C12.

7. Implemented systems

We implemented two systems. As Protégé is a well-known ontology development tool, we implemented a plugin for repairing based on Algorithm C9. Using this algorithm the user can repair all wrong axioms at once. However, by iteratively invoking this plugin the user can also repair the wrong axioms one at a time. Further, we extended the $\mathcal{EL}$ version of the RepOSE system [31, 40] with full debugging, weakening and completing. We allow the user to choose different combinations, thereby giving a choice in the trade-off between validation work and completeness.

These two implemented systems use an interactive way to repair the ontology. When the domain experts do the validation in different phases, the system supports this by showing lists of the asserted wrong/weakened/completed axioms or through a visualization whereby related axioms are shown together with the sub/sup sets of the left/right-hand concepts of the given axioms during the weakening and source(sup)/target(sub) sets of the left/right-hand concepts of the given axioms during the completing. The latter means that these axioms will be validated with their context in the domain of the ontology (showing the partial ontology through visualization).

In this paper, we use the Mini-GALEN ontology (Figs. 3 and 4) as the running example for the use of the systems.

7.1. Extension of RepOSE

We extended the $\mathcal{EL}$ version of the RepOSE system that focused on completing [31, 40]. The new system deals with $\mathcal{EL}_\bot$, handles full debugging, weakening and completing, and allows for different combinations of the basic operators. In this section, we use some examples with screenshots to show these different combinations.

To be able to use both the original ontology as well as partially repaired ontologies in different steps, the system allows loading an ontology before each step and save the (partially) repaired ontology after each step. This is done by using the buttons shown in Figure 5.

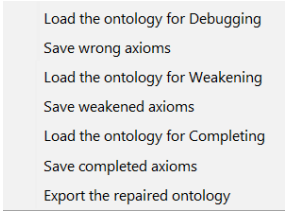
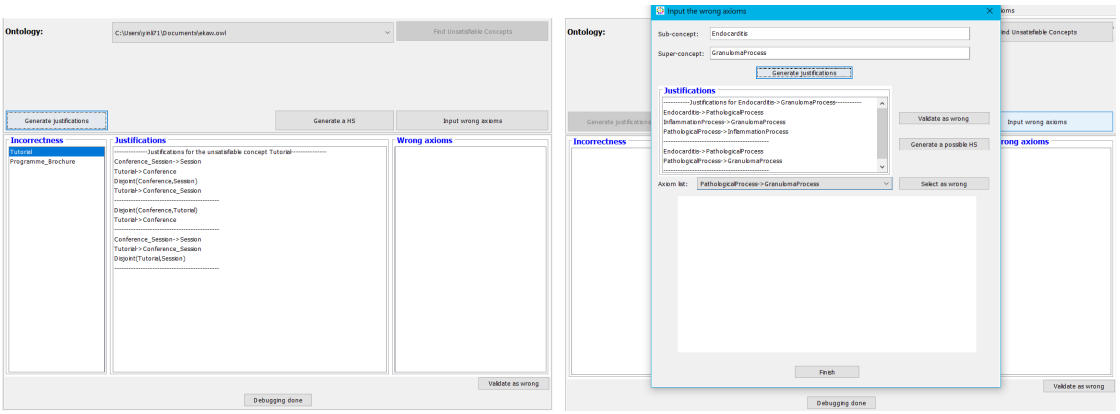


Figure 5. Loading, saving and exporting ontologies.

Example. Figure 6a shows the debugging panel of the extended RepOSE system. After loading the ontology, the user can click the *Find Unsatisfiable Concepts* button to check the incoherence of the ontology or click the *Input wrong axioms* button to specify the wrong axioms to remove from the ontology directly. If there are unsatisfiable concepts in this ontology, these concepts will be listed in the **Incorrectness** panel.

For these unsatisfiable concepts, by selecting each of them and clicking the *Generate justifications* button, the system computes the justifications and lists them in the **Justifications** panel. Then the user can validate the axioms in the justifications. Axioms deemed to be wrong can be added to the wrong axiom set by selecting them and clicking *Validate as wrong* button. In Figure 6a, the **Incorrectness** panel shows the unsatisfiable concepts in the EKAW ontology and the **Justifications** panel lists the asserted axioms in the justifications of axiom Tutorial $\sqsubseteq \bot$.

Figure 6b shows the pop-up window when clicking the *Input wrong axioms* button to specify the wrong axioms. The user can either choose the wrong asserted axiom from the axioms list directly or type the left/right hand-side concepts of a wrong axiom. For the latter case, after typing the wrong axiom, the user can also click the *Generate justifications* button to ask the system to compute the justifications of the input wrong axiom and then validate the asserted axioms from the justification list by selecting the wrong ones and clicking the *Validate as wrong* button to add them into the set of wrong axioms to be removed from the ontology at the end. After the justifications are computed, the user can also click the *Generate a possible HS* button to ask the system to compute a Hitting set based on the generated justifications. So, instead of validating all the wrong asserted axioms (D-v-all), the system



(a) Debugging panel.

(b) Justifications of the wrong axiom Endocarditis $\sqsubseteq$ GranulomaProcess.

Figure 6. Debugging

also provides another option that allows the user to only validate a Hitting set which refers to the operation D-v-one in Figure2a .

Example. Figure 6b shows the computed justifications of the wrong axiom Endocarditis $\sqsubseteq$ GranulomaProcess from the Mini-GALEN ontology. There are 3 wrong asserted axioms in the justifications: PathologicalProcess $\sqsubseteq$ InflammationProcess, InflammationProcess $\sqsubseteq$ GranulomaProcess and Endocarditis $\sqsubseteq$ PathologicalProcess. The user can select these wrong asserted axioms one by one and then click *Validate as wrong* button to add these axioms into the wrong axioms set.

When finishing the validation of the axioms by clicking the *Debugging done* button, the user can click the *Save wrong axioms* button from the file menu to save the validated wrong axioms. Then, the user can start the weakening step by clicking the *Load the ontology for Weakening* in the file menu to load the target ontology for weakening.

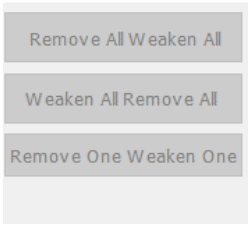
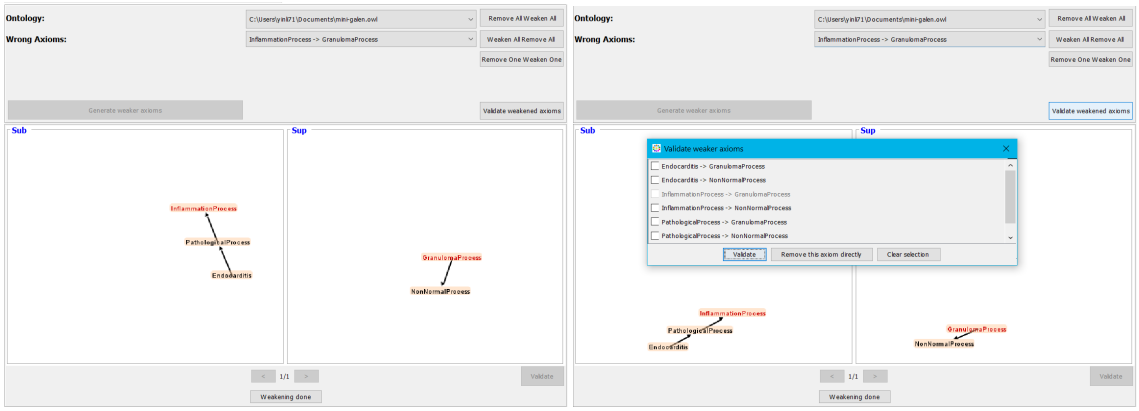


Figure 7. Different combinations of weakening and removing.

After loading the ontology for the weakening step, the user can choose different combinations of removing and weakening by clicking the different buttons in the weakening panel (Figure 7).

Weaken All Remove All refers to the combination *(R-none,AB-none/W-all,U-end_all)*. It does not remove wrong axioms during the computation, weakens all axioms at once and updates at the end. *Remove All Weaken All* refers to the combination *(R-all,AB-none/W-all,U-end_all)*. It removes all wrong axioms before the computation, weakens all axioms at once and updates at the end. *Remove One Weaken One* refers to the combination *(R-one,AB-one/W-one,U-end_all)*. It removes wrong axioms one at a time during the computation and puts them back before dealing with the next wrong axiom. It weakens axioms one at a time and updates at the end. For this step *Remove All Weaken All* produces the least validation work and least complete ontologies of the three strategies. *Remove One Weaken One* produces less validation work and less complete ontologies than *Weaken All Remove All*.

During the weakening step, after choosing the preferred combination strategy, the system generates the candidate weakened axioms for each axiom $\alpha \sqsubseteq \beta$ in the wrong asserted axioms set by clicking the *Generate weaker axioms* button. The system computes the set of sub-concepts of α (sub) and the set of super-concepts of β (sup), thereby



(a) Sub and sup of InflammationProcess ⊑ GranulomaProcess. (b) The candidate weakened axioms list of InflammationProcess ⊑ GranulomaProcess.

Figure 8. Weakening.

representing the possible choices for weakened axioms. These weakened axioms can be visualized in two ways: (i) as a list of axioms and (ii) by the sub and sup sets. In the former case weakened axioms are chosen by clicking the *Validate weakened axioms* button in the Weaken Wrong Axioms panel, selecting the axioms in the list and clicking the *Validate* button. In the latter case the user can choose weakened axioms by clicking on a concept in the sub set and a concept in the sup set and select the axiom as a weakened axiom by clicking the *Validate* button. When the validation of the candidate weakened axioms is finished, the weakening step can be ended by clicking the *Weakening done* button. Then, the user can click the *Save weakened axioms* in the file menu to save these weakened axioms.

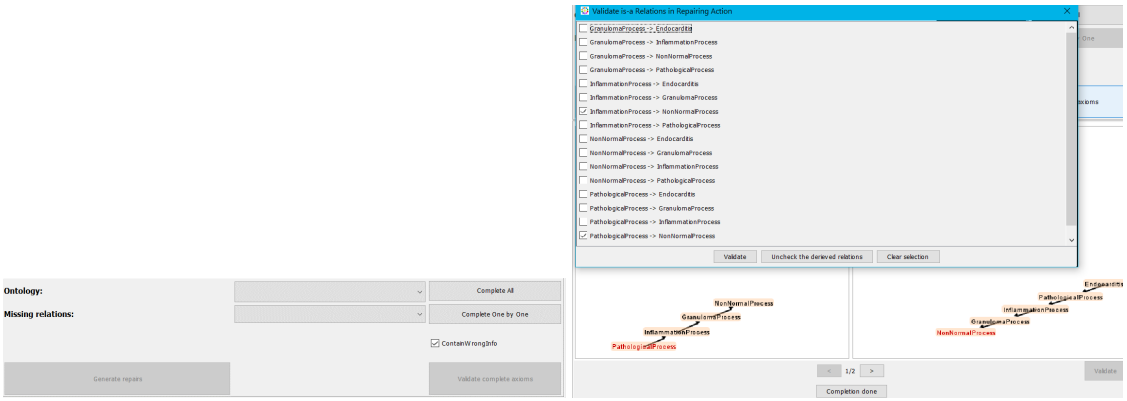
Example. Figure 8a shows the sub and sup of the wrong axiom InflammationProcess ⊑ GranulomaProcess. The weakened axiom InflammationProcess ⊑ NonNormalProcess is correct and can thus be selected by choosing the InflammationProcess from sub and NonNormalProcess from sup (visualization) or by choosing from the axiom list as in Figure 8b.

For the completion step, we also implemented two combination strategies that the user can choose by clicking the buttons *Complete All* or *Complete One by One* (Figure 9). *Complete All* refers to the combination *(C-all,U-end_all)*. It completes all weakened axioms at once and updates at the end. *Complete One by One* refers to the combination *(C-one,U-end_one)*. It completes the weakened axioms one at a time and updates the ontology after the weakened axiom set is handled for each wrong axiom. According to the Hasse diagrams in Figure2c, *Complete One by One* leads to more validation work and more complete ontologies than *Complete All*.

For the ontology used for completion, we also implemented the function that the user can remove the specified wrong asserted axioms from the target ontology by checking/unchecking the *ContainWrongInfo* checkbox as shown in Figure 9a.

During the completing step, after loading the target ontology and choosing the preferred completing combination strategy, the system generates the candidate completed axioms by clicking the *Generate repairs* button for each axiom in the validated weakened axioms set. The system computes two sets, source(sup) and target(sub) for each weakened axiom. These sets represent the possible choices for completing axioms. These axioms are visualized in two ways: (i) as a list of axioms and (ii) by the source(sup) and target(sub) sets. In the former case completed axioms are chosen by clicking the *Validate complete axioms* button, selecting the axioms in the list and clicking the *Validate* button. In the latter case the user can choose completed axioms by clicking on a concept in the source(sup) set and a concept in the target(sub) set and then validate the axiom by clicking the *Validate* button. When the validation is finished, the user can click *Completion done* to end the completing step.

Example. After the weakening step, we obtained the weakened axioms set {PathologicalProcess ⊑ NonNormalProcess, InflammationProcess ⊑ NonNormalProcess}. Figure 9b shows the sup and sub sets of PathologicalProcess ⊑ NonNormalProcess. The axiom InflammationProcess ⊑ PathologicalProcess is a correct axiom and was not derivable from the ontology yet. Adding this axiom (by using the visualization or by using the axiom list) makes the ontology more complete. Similar operations can be performed for the other axioms in the weakened axioms set.



(a) Completion panel. (b) Sup and Sub of PathlogicalProcess $\sqsubseteq$ NonNormalProcess.

Figure 9. Completing.

7.2. *Protégé plugin*

The Protégé plugin implements a simplified version of the RepOSE extension. As there exist several Protégé plugins for interactive debugging, e.g., OntoDebug[25], and the OWL Explanation Workbench[26], these tools can be used for computing explanations and providing justifications of a specific axiom or entailment during the debugging phase and then the domain expert can validate the axioms in the justifications and specify the unwanted axioms based on these explanations. We decided, therefore, that Protégé users can use these plugins and we implemented a Protégé plugin version of the RepOSE system that starts from the weakening step.

The loading and specifying of wrong axioms (Figure 10a), and the validations in the weakening (Figure 11a) and completing steps (Figure 11b) are similar to the RepOSE system. The combinations that are implemented are *(R-one,AB-one/W-one,U-end_all)* (i.e., same as *Remove One Weaken One* in RepOSE) for the weakening step, and *(C-all,U-end_all)* (i.e., same as *Complete all* in RepOSE) for the completing step. As adding completed axioms adds new knowledge to the ontology that was not earlier derivable, the system allows to find additional correct axioms by invoking the completion process again[27].

When all the desired axioms are added, clicking the *Finish* button closes this wizard and the new ontology is updated automatically. A summary panel is shown (Figure 10b), displaying the original wrong axioms, the computed weakened axioms and the completed axioms. The final ontology is created by removing the wrong axioms and adding the completed axioms. The weakened axioms will be derivable from the final ontology.

8. Related work

We discuss previous work on the *combinations* of debugging, removing, weakening and completing. As it is not the core topic of the paper, we do not review the techniques used for these basic steps, but only discuss what combination operators are used by the current approaches and for an overview of the techniques we refer to the recent overview article [3].

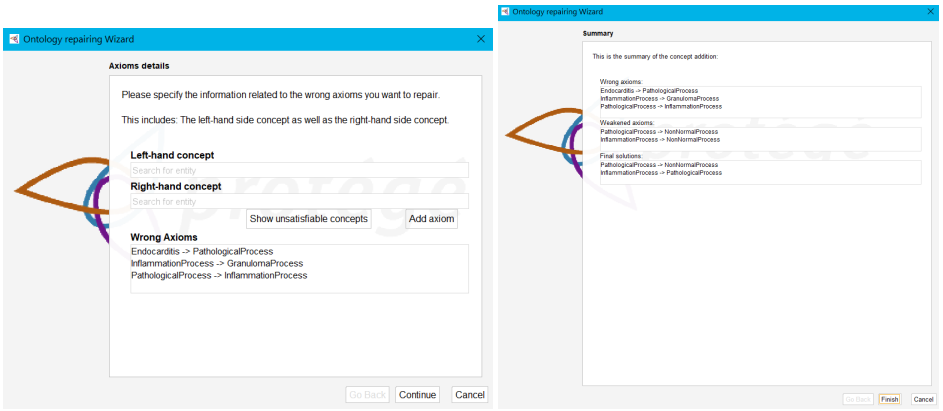
We are not aware of previous work that combines all of these basic operators.

Debugging systems often deal with semantic defects such as incoherence and inconsistency. For incoherence, for instance, they would compute the set of unsatisfiable concepts. The aim is then to remove the unsatisfiability. In our

[25] http://isbi.aau.at/ontodebug/

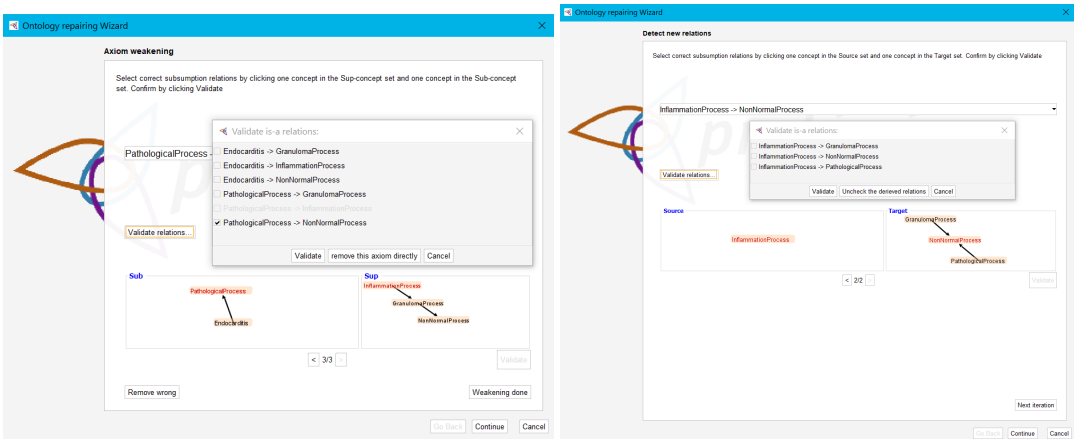
[26] http://owl.cs.manchester.ac.uk/research/explanation/

[27] The possibility of multiple iterations of the completing phase is an extension of the method in [1]. This can be done by clicking the *Next iteration* button.



(a) Input the whole wrong axioms set. (b) The summary info panel.

Figure 10.



(a) Sub and sup of PathologicalProcess $\sqsubseteq$InflammationProcess. (b) Source and target of InflammationProcess $\sqsubseteq$ NonNormalProcess.

Figure 11.

framework this would be a way to compute the set of wrong axioms, i.e., an unsatisfiable concept P contributes a wrong axiom $P \sqsubseteq \bot$. The input set W contains then these axioms.

Much of the work on debugging algorithms and systems mainly focuses on the computation of justifications for one or more unsatisfiable concepts or wrong axioms [2, 4–12, 14]. Several systems take a set of unsatisfiable concepts and try to find the root concepts, i.e., unsatisfiable concepts for which a contradiction in its definition does not depend on the unsatisfiability of another concept, e.g., [2, 10, 12]. The systems work with one unsatisfiable concept or wrong axiom at the time (S-one, e.g., [10, 11, 18]) or all at once (S-all, e.g., [6, 15]). Some of the systems have a user interface. The older systems usually visualize the justifications of wrong axioms, but it is not clear how the actual repairing is done. When validation of axioms is enabled, then often D-all is used, e.g., [10].

Weakening is always used in combination with **debugging or removing**. In these approaches, justifications for wrong axioms and a hitting set are computed. Then, weakened axioms are computed. In our approach we assume that when removing axioms from the ontology, the wrong axioms cannot be derived anymore. When this assumption is not made then, as pointed out in [22] (and ignored by older approaches) the weakening needs to be iterated to obtain a repair. We also note that none of the approaches explicitly state the use of a domain expert/oracle and they are purely logic-based. In practice, however, a domain expert/oracle is needed as otherwise axioms that are wrong in the domain of the ontology could be added. Regarding the weakening *algorithm*, in contrast to our approach, the other

approaches work on non-normalized TBoxes. This means that they may find better solutions for the weakening, but the search space for solutions also becomes infinite. In [23] algorithms for weakening for $\mathcal{EL}$ and $\mathcal{ALC}$ are given with tractable and exponential complexity, respectively. They are based on refinement operators that are applied on the concepts of GCIs. The approach is extended in [24] for $\mathcal{SROIQ}$ TBoxes with an algorithm with almost-sure termination. Also in [21] an approach based on refinement operators is presented for $\mathcal{ALC}$. The nesting of operators is restricted based on the size of a concept. In [22] the right-hand side of axioms is generalized, but the left-hand side is not specialized to obtain a well-founded weakening relation (i.e., there is no infinite chain of weakenings). Essentially, our use of $sup(P, \mathcal{T})$ and $sub(P, \mathcal{T})$ in the weakening is a similar approach. As we have restricted the $sup(P, \mathcal{T})$ and $sub(P, \mathcal{T})$ to contain only concepts in SCC($\mathcal{T}$), we only have a finite number of possible axioms. Regarding the *strategy to combine* removing or debugging with weakening, in all these other approaches usually one-at-a-time removing (R-one, AB-none) and weakening (W-one, U-now) is used. From our Hasse diagrams we can see that this means the most complete ontologies and most validation work for weakening, but neither the most nor the least complete ontologies for removing. We note that using our Hasse diagrams, new variants of these other approaches can be created with another trade-off involving correctness and completeness. Further, the issue of the influence of the order is not addressed.

Regarding **completing**, previous work with validation by a domain expert (e.g., [31] for the $\mathcal{EL}$ family, [32] for $\mathcal{ALC}$) allowed only axioms of the form $P \sqsubseteq Q$ where P and Q are atomic concepts in the completed axioms set while Algorithm 3 allows P and Q to be in SCC($\mathcal{T}$) (and then normalizes). That work used one particular combination strategy, i.e., (C-one, U-end-all). Other approaches, e.g., [33, 34], are non-interactive and deal with one axiom to be completed.

Debugging, removing and completing were combined in [41]. As input, in addition to a set of wrong axioms W, a set of missing axioms M should be given. Debugging is mainly used on W (with S-one, D-all) and completing on M (C-one, U-end-all). There is a restriction that axioms validated as correct in any step cannot be removed and axioms validated as wrong during any step cannot be added to the ontology. The steps can be interleaved. If the approach is used without a set of missing axioms, i.e. $M = \emptyset$, then the approach is a traditional debugging approach. Another approach that can be seen as combining these operations is the interactive test-driven debugging approach proposed in [13, 17] and implemented in the OntoDebug system. In this approach a user can input both wrong axioms and axioms that are considered to be correct. The system uses debugging algorithms to obtain diagnoses (which reflect different solutions for D as a part of a repair in Definition 1) and then generates queries about entailments, asks a domain expert to answer these queries (i.e., validation), and uses these answers to guide the identification of the target diagnosis.

9. Conclusion

In this paper we proposed an interactive approach using weakening and completing to mitigate the negative effects of removing wrong axioms in $\mathcal{EL}_\bot$ ontologies. We presented a framework (and the first approach) for combining debugging, removing with weakening and completing. We showed that there are different combination strategies and that there is a trade-off involving correctness and completeness. We also introduced a way to compare combination strategies and showed that earlier work covered one type of combination strategy. Further, we presented new algorithms for weakening and completion and using these, showed the influence of different combination strategies on the completeness for 6 ontologies in experiments.

For future work we will investigate the problem for more expressive description logics (e.g. $\mathcal{ALC}$). It is clear that, when applying this repairing framework to a more expressive language, we may need to look into other strategies for limiting and reducing the search space while still maintaining a practically feasible validation work for the domain expert during weakening and completing phase. Another direction for future work is to deal with ontology networks where the owners of the ontologies and alignments in the ontology network may have different policies regarding allowing others to change or propose changes to their ontologies and alignments. Also, computation time and validation resources may be a bigger issue for ontology networks than for single ontologies.

Acknowledgements: We thank Olaf Hartig for discussions leading to the Hasse diagrams. This work is financially supported by the Swedish e-Science Research Centre (SeRC), the Swedish Research Council (Vetenskapsrådet, dnr 2018-04147) and the Horizon Europe project Onto-DESIDE under Grant Agreement 101058682.

References

[1] Y. Li and P. Lambrix, Repairing $\mathcal{EL}$ Ontologies Using Weakening and Completing, in: *The Semantic Web - 20th International Conference, ESWC 2023, Hersonissos, Crete, Greece, May 28 - June 1, 2023, Proceedings*, C. Pesquita, E. Jimenez-Ruiz, J. McCusker, D. Faria, M. Dragoni, A. Dimou, R. Troncy and S. Hertling, eds, Lecture Notes in Computer Science, Vol. 13870, Springer, 2023, pp. 298–315. doi:10.1007/978-3-031-33455-9_18.

[2] A. Kalyanpur, B. Parsia, E. Sirin and J.A. Hendler, Debugging unsatisfiable classes in OWL ontologies, *Journal of Web Semantics* **3**(4) (2005), 268–293. doi:10.1016/J.WEBSEM.2005.09.005.

[3] P. Lambrix, Completing and Debugging Ontologies: State of the Art and Challenges in Repairing Ontologies, *ACM Journal of Data and Information Quality* **15**(4) (2023), 41:1–38. doi:10.1145/3597304.

[4] S. Schlobach and R. Cornet, Non-Standard Reasoning Services for the Debugging of Description Logic Terminologies, in: *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, G. Gottlob and T. Walsh, eds, 2003, pp. 355–360. https://www.ijcai.org/Proceedings/03/Papers/053.pdf.

[5] S. Schlobach, Debugging and Semantic Clarification by Pinpointing, in: *The Semantic Web: Research and Applications - Second European Semantic Web Conference, ESWC 2005, Heraklion, Crete, Greece, May 29-June 1, 2005. Proceedings*, A. Gomez-Perez and J. Euzenat, eds, Lecture Notes in Computer Science, Vol. 3532, Springer, 2005, pp. 226–240. doi:10.1007/11431053_16.

[6] S. Schlobach, Z. Huang, R. Cornet and F. van Harmelen, Debugging Incoherent Terminologies, *Journal of Automated Reasoning* **39**(3) (2007), 317–349. doi:10.1007/s10817-007-9076-z.

[7] T. Meyer, K. Lee, R. Booth and J. Pan, Finding maximally satisfiable terminologies for the description logic ALC, in: *AAAI'06 - Proceedings of the 21st national conference on Artificial intelligence*, A. Cohn, ed., AAAI Press, 2006, pp. 269–274. https://www.aaai.org/Papers/AAAI/2006/AAAI06-043.pdf.

[8] A. Kalyanpur, B. Parsia, E. Sirin and B. Cuenca Grau, Repairing Unsatisfiable Concepts in OWL Ontologies, in: *The Semantic Web: Research and Applications, 3rd European Semantic Web Conference, ESWC 2006, Budva, Montenegro, June 11-14, 2006, Proceedings*, Y. Sure and J. Domingue, eds, Lecture Notes in Computer Science, Vol. 4011, Springer, 2006, pp. 170–184. doi:10.1007/11762256_15.

[9] A. Kalyanpur, B. Parsia, M. Horridge and E. Sirin, Finding All Justifications of OWL DL Entailments, in: *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11-15, 2007*, K. Aberer, K. Choi, N.F. Noy, D. Allemang, K. Lee, L.J.B. Nixon, J. Golbeck, P. Mika, D. Maynard, R. Mizoguchi, G. Schreiber and P. Cudré-Mauroux, eds, Lecture Notes in Computer Science, Vol. 4825, Springer, 2007, pp. 267–280. doi:10.1007/978-3-540-76298-0_20.

[10] J. Lehmann and L. Bühmann, ORE - A Tool for Repairing and Enriching Knowledge Bases, in: *The Semantic Web - ISWC 2010 - 9th International Semantic Web Conference, ISWC 2010, Shanghai, China, November 7-11, 2010, Revised Selected Papers, Part II*, P.F. Patel-Schneider, Y. Pan, P. Hitzler, P. Mika, L. Zhang, J.Z. Pan, I. Horrocks and B. Glimm, eds, Lecture Notes in Computer Science, Vol. 6497, Springer, 2010, pp. 177–193. doi:10.1007/978-3-642-17749-1_12.

[11] Q. Ji, Z. Gao, Z. Huang and M. Zhu, An Efficient Approach to Debugging Ontologies Based on Patterns, in: *The Semantic Web - Joint International Semantic Technology Conference, JIST 2011, Hangzhou, China, December 4-7, 2011. Proceedings*, J.Z. Pan, H. Chen, H. Kim, J. Li, Z. Wu, I. Horrocks, R. Mizoguchi and Z. Wu, eds, Lecture Notes in Computer Science, Vol. 7185, Springer, 2011, pp. 425–433. doi:10.1007/978-3-642-29923-0_33.

[12] K. Moodley, T. Meyer and I.J. Varzinczak, Root Justifications for Ontology Repair, in: *Web Reasoning and Rule Systems - 5th International Conference, RR 2011, Galway, Ireland, August 29-30, 2011. Proceedings*, S. Rudolph and C. Gutiérrez, eds, Lecture Notes in Computer Science, Vol. 6902, Springer, 2011, pp. 275–280. doi:10.1007/978-3-642-23580-1_24.

[13] K.M. Shchekotykhin, G. Friedrich, P. Fleiss and P. Rodler, Interactive ontology debugging: Two query strategies for efficient fault localization, *Journal of Web Semantics* **12** (2012), 88–103. doi:10.1016/j.websem.2011.12.006.

[14] D. Fleischhacker, C. Meilicke, J. Völker and M. Niepert, Computing Incoherence Explanations for Learned Ontologies, in: *Web Reasoning and Rule Systems - 7th International Conference, RR 2013, Mannheim, Germany, July 27-29, 2013. Proceedings*, W. Faber and D. Lembo, eds, Lecture Notes in Computer Science, Vol. 7994, Springer, 2013, pp. 80–94. doi:10.1007/978-3-642-39666-3_7.

[15] M.F. Arif, C. Mencía, A. Ignatiev, N. Manthey, R. Peñaloza and J. Marques-Silva, BEACON: An Efficient SAT-Based Tool for Debugging $\mathcal{EL}^+$ Ontologies, in: *Theory and Applications of Satisfiability Testing - SAT 2016 - 19th International Conference, Bordeaux, France, July 5-8, 2016, Proceedings*, N. Creignou and D.L. Berre, eds, Lecture Notes in Computer Science, Vol. 9710, Springer, 2016, pp. 521–530. doi:10.1007/978-3-319-40970-2_32.

[16] P. Rodler and W. Schmid, On the Impact and Proper Use of Heuristics in Test-Driven Ontology Debugging, in: *Rules and Reasoning - Second International Joint Conference, RuleML+RR 2018, Luxembourg, September 18-21, 2018, Proceedings*, C. Benzmüller, F. Ricca, X. Parent and D. Roman, eds, Lecture Notes in Computer Science, Vol. 11092, Springer, 2018, pp. 164–184. doi:10.1007/978-3-319-99906-7_11.

[17] K. Schekotihin, P. Rodler and W. Schmid, OntoDebug: Interactive Ontology Debugging Plug-in for Protégé, in: *Foundations of Information and Knowledge Systems - 10th International Symposium, FoIKS 2018, Budapest, Hungary, May 14-18, 2018, Proceedings*, F. Ferrarotti and S. Woltran, eds, Lecture Notes in Computer Science, Vol. 10833, Springer, 2018, pp. 340–359. doi:10.1007/978-3-319-90050-6_19.

[18] J. Méndez, C. Alrabbaa, P. Koopmann, R. Langner, F. Baader and R. Dachselt, Evonne: A Visual Tool for Explaining Reasoning with OWL Ontologies and Supporting Interactive Debugging, *Computer Graphics forum* **42**(6) (2023), e14730:1–15. doi:10.1111/cgf.14730.

[19] C. Pesquita, D. Faria, E. Santos and F.M. Couto, To repair or not to repair: reconciling correctness and coherence in ontology reference alignments, in: *Proceedings of the 8th International Workshop on Ontology Matching*, P. Shvaiko, J. Euzenat, K. Srinivas, M. Mao and E. Jiménez-Ruiz, eds, CEUR Workshop Proceedings, Vol. 1111, 2013, pp. 13–24. http://ceur-ws.org/Vol-1111/om2013_Tpaper2.pdf.

[20] J.S.C. Lam, D.H. Sleeman, J.Z. Pan and W.W. Vasconcelos, A Fine-Grained Approach to Resolving Unsatisfiable Ontologies, *Journal on Data Semantics* **10** (2008), 62–95. doi:10.1007/978-3-540-77688-8_3.

[21] J. Du, G. Qi and X. Fu, A Practical Fine-grained Approach to Resolving Incoherent OWL 2 DL Terminologies, in: *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, CIKM 2014, Shanghai, China, November 3-7, 2014*, J. Li, X.S. Wang, M.N. Garofalakis, I. Soboroff, T. Suel and M. Wang, eds, ACM, 2014, pp. 919–928. doi:10.1145/2661829.2662046.

[22] F. Baader, F. Kriegel, A. Nuradiansyah and R. Peñaloza, Making Repairs in Description Logics More Gentle, in: *Principles of Knowledge Representation and Reasoning: Proceedings of the Sixteenth International Conference, KR 2018, Tempe, Arizona, 30 October - 2 November 2018.*, M. Thielscher, F. Toni and F. Wolter, eds, 2018, pp. 319–328. https://aaai.org/ocs/index.php/KR/KR18/paper/view/18056.

[23] N. Troquard, R. Confalonieri, P. Galliani, R. Peñaloza, D. Porello and O. Kutz, Repairing Ontologies via Axiom Weakening, in: *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, S.A. McIlraith and K.Q. Weinberger, eds, AAAI Press, 2018, pp. 1981–1988. https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17189.

[24] R. Confalonieri, P. Galliani, O. Kutz, D. Porello, G. Righetti and N. Troquard, Towards Even More Irresistible Axiom Weakening, in: *Proceedings of the 33rd International Workshop on Description Logics (DL 2020)*, S. Borgwardt and T. Meyer, eds, CEUR Workshop Proceedings, Vol. 2663, 2020. https://ceur-ws.org/Vol-2663/paper-8.pdf.

[25] F. Baader, B. Ganter, B. Sertkaya and U. Sattler, Completing Description Logic Knowledge Bases Using Formal Concept Analysis, in: *IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6-12, 2007*, M.M. Veloso, ed., 2007, pp. 230–235. http://ijcai.org/Proceedings/07/Papers/035.pdf.

[26] B. Sertkaya, OntoComP: A Protégé Plugin for Completing OWL Ontologies, in: *The Semantic Web: Research and Applications, 6th European Semantic Web Conference, ESWC 2009, Heraklion, Crete, Greece, May 31-June 4, 2009, Proceedings*, L. Aroyo, P. Traverso, F. Ciravegna, P. Cimiano, T. Heath, E. Hyvönen, R. Mizoguchi, E. Oren, M. Sabou and E. Simperl, eds, Lecture Notes in Computer Science, Vol. 5554, Springer, 2009, pp. 898–902. doi:10.1007/978-3-642-02121-3_78.

[27] H. Li, R. Armiento and P. Lambrix, A Method for Extending Ontologies with Application to the Materials Science Domain, *Data Science Journal* **18**(1) (2019), 50:1–21. doi:10.5334/dsj-2019-050.

[28] P. Buitelaar, P. Cimiano and B. Magnini, *Ontology Learning from Text: Methods, Evaluation and Applications*, IOS Press, 2005. ISBN 1-58603-523-1.

[29] S. Ferré and S. Rudolph, Advocatus Diaboli – Exploratory Enrichment of Ontologies with Negative Constraints, in: *EKAW: Knowledge Engineering and Knowledge Management*, Lecture Notes in Artificial Intelligence, Vol. 7603, Springer, 2012, pp. 42–56. doi:10.1007/978-3-642-33876-2_7.

[30] N. Nikitina, S. Rudolph and B. Glimm, Interactive ontology revision, *Journal of Web Semantics* **12-13** (2012), 118–130. doi:10.1016/j.websem.2011.12.002.

[31] F. Wei-Kleiner, Z. Dragisic and P. Lambrix, Abduction Framework for Repairing Incomplete $\mathcal{EL}$ Ontologies: Complexity Results and Algorithms, in: *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, July 27 -31, 2014, Québec City, Québec, Canada*, C.E. Brodley and P. Stone, eds, AAAI Press, 2014, pp. 1120–1127. doi:10.1609/AAAI.V28I1.8858.

[32] P. Lambrix, Z. Dragisic and V. Ivanova, Get My Pizza Right: Repairing Missing is-a Relations in $\mathcal{ALC}$ Ontologies, in: *Semantic Technology, Second Joint International Conference, JIST 2012, Nara, Japan, December 2-4, 2012. Proceedings*, H. Takeda, Y. Qu, R. Mizoguchi and Y. Kitamura, eds, Lecture Notes in Computer Science, Vol. 7774, Springer, 2012, pp. 17–32. doi:10.1007/978-3-642-37996-3_2.

[33] J. Du, H. Wan and H. Ma, Practical TBox Abduction Based on Justification Patterns, in: *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA.*, S.P. Singh and S. Markovitch, eds, 2017, pp. 1100–1106. doi:10.1609/aaai.v31i1.10683.

[34] F. Haifani, P. Koopmann, S. Tourret and C. Weidenbach, Connection-Minimal Abduction in $\mathcal{EL}$ via Translation to FOL, in: *Automated Reasoning, IJCAR 2022*, J. Blanchette, L. Kovács and D. Pattinson, eds, Lecture Notes in Computer Science, Vol. 13385, Springer, 2022, pp. 188–207. doi:10.1007/978-3-031-10769-6_12.

[35] F. Baader, S. Brandt and C. Lutz, Pushing the $\mathcal{EL}$ Envelope, in: *IJCAI-05, Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence, Edinburgh, Scotland, UK, July 30 - August 5, 2005*, L.P. Kaelbling and A. Saffiotti, eds, Professional Book Center, 2005, pp. 364–369. http://ijcai.org/Proceedings/05/Papers/0372.pdf.

[36] F. Baader, D. Calvanese, D.L. McGuinness, D. Nardi and P.F. Patel-Schneider (eds), *The Description Logic Handbook: Theory, Implementation, and Applications*, Cambridge University Press, 2003. ISBN 0-521-78176-0.

[37] Z. Dragisic, V. Ivanova, H. Li and P. Lambrix, Experiences from the anatomy track in the ontology alignment evaluation initiative, *Journal of Biomedical Semantics* **8**(1) (2017), 56:1–28. doi:10.1186/s13326-017-0166-5.

[38] P. Lambrix, F. Wei-Kleiner, Z. Dragisic and V. Ivanova, Repairing missing is-a structure in ontologies is an abductive reasoning problem, in: *Proceedings of the Second International Workshop on Debugging Ontologies and Ontology Mappings, Montpellier, France, May 27, 2013*, P. Lambrix, G. Qi, M. Horridge and B. Parsia, eds, CEUR Workshop Proceedings, Vol. 999, CEUR-WS.org, 2013, pp. 33–44. https://ceur-ws.org/Vol-999/paper3.pdf.

[39] M. Horridge, B. Parsia and U. Sattler, Laconic and Precise Justifications in OWL, in: *The Semantic Web - ISWC 2008, 7th International Semantic Web Conference, ISWC 2008, Karlsruhe, Germany, October 26-30, 2008. Proceedings*, A.P. Sheth, S. Staab, M. Dean, M. Paolucci, D. Maynard, T.W. Finin and K. Thirunarayan, eds, Lecture Notes in Computer Science, Vol. 5318, Springer, 2008, pp. 323–338. doi:10.1007/978-3-540-88564-1_21.

[40] P. Lambrix, F. Wei-Kleiner and Z. Dragisic, Completing the is-a structure in light-weight ontologies, *Journal of Biomedical Semantics* **6** (2015), 12:1–26. doi:10.1186/S13326-015-0002-8.

[41] P. Lambrix and V. Ivanova, A unified approach for debugging is-a structure and mappings in networked taxonomies, *Journal of Biomedical Semantics* **4** (2013), 10:1–19. doi:10.1186/2041-1480-4-10.

Appendix

In this appendix, we give more details about algorithms and experiments.

A. Normalization algorithm
B. Algorithms for combination strategies
C. Experiments - data
D. Experiments - results

A. Normalization algorithm

Algorithm 11 rewrites an axiom into one of the allowed forms.

Algorithm 11 Normalize($sb \sqsubseteq sp$)

Input: Axiom $sb \sqsubseteq sp$
Output: A set of axioms in normalized form

if $sp \in N_c$ **then**
    return { $sb \sqsubseteq sp$ }
else if sp is of the form $P \sqcap Q$ **then**
    return { $sb \sqsubseteq P$, $sb \sqsubseteq Q$ }
else if sp is of the form $\exists r.P$ **then**
    if $sb \in N_c$ **then**
        return { $sb \sqsubseteq sp$ }
    else if sb is of the form $\exists r.Q$ **then**
        Introduce new concept Z
        return { $\exists r.Q \sqsubseteq Z, Z \sqsubseteq \exists r.Q, Z \sqsubseteq sp$}
    else if sb is of the form $\exists s.Q$ **then**
        Introduce new concept Z
        return { $\exists s.Q \sqsubseteq Z, Z \sqsubseteq \exists s.Q, Z \sqsubseteq sp$}
    else if sb is of the form $Q \sqcap R$ **then**
        Introduce new concept Z
        return { $Q \sqcap R \sqsubseteq Z, Z \sqsubseteq Q, Z \sqsubseteq R, Z \sqsubseteq sp$ }
    end if
end if

B. Algorithms for combination strategies

In this part, we give more details about the different algorithms used in the experiments. We show all our algorithms for combining different removing, weakening and completing strategies including the ones that were presented in the paper earlier. A brief description of each algorithm is shown in Table 8.

Table 8
Algorithms.

Algorithm	Description
C1	Weaken one at a time, add weakened axiom sets and remove all wrong at end
C2	Remove/weaken/add weakened axiom sets one at a time
C3	Remove all wrong, weaken one at a time, add weakened axiom sets at end
C4	Remove all wrong, weaken/add weakened axiom sets one at a time
C5	Weaken all wrong, complete all weakened axioms, add completed axiom sets and remove all wrong at end
C6	Weaken/complete/add completed axiom sets one at a time, remove all wrong at end
C7	Remove/weaken/complete/add completed axiom sets one at a time
C8	Weaken/complete one at a time, add completed axiom sets and remove all wrong at end
C9	Weaken all wrong, remove all wrong, complete all weakened axioms and then add completed axiom sets at end
C10	Weaken all wrong, remove all wrong, complete/add completed axiom sets one at a time
C11	Weaken all wrong, remove/complete/add the wrong/completed axiom sets one at a time, remove all wrong at end
C12	Remove all wrong, weaken all wrong, complete all weakened axioms and add completed axiom sets at end
C13	Remove all wrong, weaken/complete/add completed axiom sets one at a time
C14	Generate the justifications of each wrong axiom and validate all the asserted axioms, weaken all wrong , remove all wrong, complete/add completed axiom set one at a time
C15	Generate the justifications of each wrong axiom and validate one hitting set for each wrong axiom, weaken all wrong, remove all wrong, complete/add completed axiom set one at a time
C16	Generate the justifications of each wrong axiom and validate all the asserted axioms, weaken one at a time, complete all, add all completed axioms and remove all wrong at the end
C17	Generate the justifications of each wrong axiom and validate all the asserted axioms, remove all wrong, weaken all wrong, add all completed axioms together in the end
C18	Generate the justifications of each wrong axiom and validate all the asserted axioms, remove all wrong, weaken all wrong, complete/add completed axiom set one at a time
C19	Generate the justifications of each wrong axiom and validate all the asserted axioms, remove all wrong, weaken/complete/add completed axiom sets one at a time
C20	Generate the justifications of each wrong axiom and validate all the asserted axioms, weaken/complete/add completed axiom sets one at a time, and remove all wrong in the end

Algorithm C1 Weaken one at a time, add weakened axiom sets and remove all wrong at end

Input: TBox $\mathcal{T}$, Oracle Or, set of unwanted axioms W
Output: A repaired TBox

1: **for each** $\alpha \sqsubseteq \beta \in W$ **do**
2: $\quad \mathcal{T}_r \leftarrow$ Remove-axioms($\mathcal{T}$, $\{\alpha \sqsubseteq \beta\}$)
3: $\quad w_{\alpha \sqsubseteq \beta} \leftarrow$ weakened-axiom-set($\alpha \sqsubseteq \beta$, $\mathcal{T}_r$, Or)
4: **end for**
5: $\mathcal{T}_r \leftarrow$ Add-axioms($\mathcal{T}$,$\bigcup_{\alpha \sqsubseteq \beta} w_{\alpha \sqsubseteq \beta}$)
6: **return** Remove-axioms($\mathcal{T}_r$,W)

Algorithm C2 Remove/weaken/add weakened axiom sets one at a time

Input: TBox $\mathcal{T}$, Oracle Or, set of unwanted axioms W
Output: A repaired TBox

1: $\mathcal{T}_r \leftarrow \mathcal{T}$
2: **for each** $\alpha \sqsubseteq \beta \in W$ **do**
3: $\quad \mathcal{T}_r \leftarrow$ Remove-axioms($\mathcal{T}_r$, $\{\alpha \sqsubseteq \beta\}$)
4: $\quad w_{\alpha \sqsubseteq \beta} \leftarrow$ weakened-axiom-set($\alpha \sqsubseteq \beta$, $\mathcal{T}_r$, Or)
5: $\quad \mathcal{T}_r \leftarrow$ Add-axioms($\mathcal{T}_r$,$w_{\alpha \sqsubseteq \beta}$)
6: **end for**
7: **return** $\mathcal{T}_r$

Algorithm C3 Remove all wrong, weaken all and add weakened axiom sets at end

Input: TBox $\mathcal{T}$, Oracle Or, set of unwanted axioms W
Output: A repaired TBox

1: $\mathcal{T}_r \leftarrow$ Remove-axioms($\mathcal{T}$, W)
2: **for each** $\alpha \sqsubseteq \beta \in W$ **do**
3: $\quad w_{\alpha \sqsubseteq \beta} \leftarrow$ weakened-axiom-set($\alpha \sqsubseteq \beta$, $\mathcal{T}_r$, Or)
4: **end for**
5: **return** Add-axioms($\mathcal{T}_r$,$\bigcup_{\alpha \sqsubseteq \beta} w_{\alpha \sqsubseteq \beta}$)

Algorithm C4 Remove all wrong, weaken/add weakened axiom sets one at a time

Input: TBox $\mathcal{T}$, Oracle Or, set of unwanted axioms W
Output: A repaired TBox

1: $\mathcal{T}_r \leftarrow$ Remove-axioms($\mathcal{T}$, W)
2: **for each** $\alpha \sqsubseteq \beta \in W$ **do**
3: $\quad w_{\alpha \sqsubseteq \beta} \leftarrow$ weakened-axiom-set($\alpha \sqsubseteq \beta$, $\mathcal{T}_r$, Or)
4: $\quad \mathcal{T}_r \leftarrow$ Add-axioms($\mathcal{T}_r$,$w_{\alpha \sqsubseteq \beta}$)
5: **end for**
6: **return** $\mathcal{T}_r$

Algorithm C5 Weaken all, complete one at a time, add completed axiom set and remove all wrong at end

Input: TBox $\mathcal{T}$, Oracle Or, set of unwanted axioms W
Output: A repaired TBox

for each $\alpha \sqsubseteq \beta \in W$ **do**
  $\mathcal{T}_r \leftarrow$ Remove-axioms($\mathcal{T}$, $\{\alpha \sqsubseteq \beta\}$)
  $w_{\alpha \sqsubseteq \beta} \leftarrow$ weakened-axiom-set($\alpha \sqsubseteq \beta$, $\mathcal{T}_r$, Or)
end for
for each $\alpha \sqsubseteq \beta \in W$ **do**
  $c_{\alpha \sqsubseteq \beta} \leftarrow \emptyset$
  for each $sb \sqsubseteq sp \in w_{\alpha \sqsubseteq \beta}$ **do**
    $c_{sb \sqsubseteq sp} \leftarrow$ completed-axiom-set($sb \sqsubseteq sp$, $\mathcal{T}$, Or)
    $c_{\alpha \sqsubseteq \beta} \leftarrow c_{\alpha \sqsubseteq \beta} \cup c_{sb \sqsubseteq sp}$
  end for
end for
$\mathcal{T}_r \leftarrow$ Add-axioms($\mathcal{T}$,$\bigcup_{\alpha \sqsubseteq \beta} c_{\alpha \sqsubseteq \beta}$)
return Remove-axioms($\mathcal{T}_r$,W)

Algorithm C6 Weaken/complete/add completed axiom sets one at a time, remove all wrong at end

Input: TBox $\mathcal{T}$, Oracle Or, set of unwanted axioms W
Output: A repaired TBox

for each $\alpha \sqsubseteq \beta \in W$ **do**
  $\mathcal{T}_r \leftarrow$ Remove-axioms($\mathcal{T}$, $\{\alpha \sqsubseteq \beta\}$)
  $w_{\alpha \sqsubseteq \beta} \leftarrow$ weakened-axiom-set($\alpha \sqsubseteq \beta$, $\mathcal{T}_r$, Or)
  $c_{\alpha \sqsubseteq \beta} \leftarrow \emptyset$
  for each $sb \sqsubseteq sp \in w_{\alpha \sqsubseteq \beta}$ **do**
    $c_{sb \sqsubseteq sp} \leftarrow$ completed-axiom-set($sb \sqsubseteq sp$, $\mathcal{T}_r$, Or)
    $\mathcal{T} \leftarrow$ Add-axioms($\mathcal{T}$, $c_{sb \sqsubseteq sp}$)
  end for
end for
return Remove-axioms($\mathcal{T}$,W)

Algorithm C7 Remove/weaken/complete/add completed axiom sets one at a time

Input: TBox $\mathcal{T}$, Oracle Or, set of unwanted axioms W
Output: A repaired TBox

$\mathcal{T}_r \leftarrow \mathcal{T}$
for each $\alpha \sqsubseteq \beta \in W$ **do**
  $\mathcal{T}_r \leftarrow$ Remove-axioms($\mathcal{T}_r$, $\{\alpha \sqsubseteq \beta\}$)
  $w_{\alpha \sqsubseteq \beta} \leftarrow$ weakened-axiom-set($\alpha \sqsubseteq \beta$, $\mathcal{T}_r$, Or)
  $c_{\alpha \sqsubseteq \beta} \leftarrow \emptyset$
  for each $sb \sqsubseteq sp \in w_{\alpha \sqsubseteq \beta}$ **do**
    $c_{sb \sqsubseteq sp} \leftarrow$ completed-axiom-set($sb \sqsubseteq sp$, $\mathcal{T}_r$, Or)
    $\mathcal{T}_r \leftarrow$ Add-axioms($\mathcal{T}_r$, $c_{sb \sqsubseteq sp}$)
  end for
end for
return $\mathcal{T}_r$

Algorithm C8 Weaken/complete one at a time, add completed axiom sets and remove all wrong axioms at end

Input: TBox $\mathcal{T}$, Oracle Or, set of unwanted axioms W
Output: A repaired TBox

for each $\alpha \sqsubseteq \beta \in W$ **do**
  $\mathcal{T}_r \leftarrow$ Remove-axioms($\mathcal{T}$, $\{\alpha \sqsubseteq \beta\}$)
  $w_{\alpha \sqsubseteq \beta} \leftarrow$ weakened-axiom-set($\alpha \sqsubseteq \beta$, $\mathcal{T}_r$, Or)
  $c_{\alpha \sqsubseteq \beta} \leftarrow \emptyset$
  for each $sb \sqsubseteq sp \in w_{\alpha \sqsubseteq \beta}$ **do**
    $c_{sb \sqsubseteq sp} \leftarrow$ completed-axiom-set($sb \sqsubseteq sp$, $\mathcal{T}$, Or)
    $c_{\alpha \sqsubseteq \beta} \leftarrow c_{\alpha \sqsubseteq \beta} \cup c_{sb \sqsubseteq sp}$
  end for
end for
$\mathcal{T}_r \leftarrow$ Add-axioms($\mathcal{T}$,$\bigcup_{\alpha \sqsubseteq \beta} c_{\alpha \sqsubseteq \beta}$)
return Remove-axioms($\mathcal{T}_r$,W)

Algorithm C9 Weaken all wrong, remove all wrong, complete one at a time, then add completed axiom sets at end

Input: TBox $\mathcal{T}$, Oracle Or, set of unwanted axioms W
Output: A repaired TBox

for each $\alpha \sqsubseteq \beta \in W$ **do**
  $\mathcal{T}_r \leftarrow$ Remove-axioms($\mathcal{T}$, $\{\alpha \sqsubseteq \beta\}$)
  $w_{\alpha \sqsubseteq \beta} \leftarrow$ weakened-axiom-set($\alpha \sqsubseteq \beta$, $\mathcal{T}_r$, Or)
end for
$\mathcal{T}_r \leftarrow$ Remove-axioms($\mathcal{T}_r$,W)
for each $\alpha \sqsubseteq \beta \in W$ **do**
  $c_{\alpha \sqsubseteq \beta} \leftarrow \emptyset$
  for each $sb \sqsubseteq sp \in w_{\alpha \sqsubseteq \beta}$ **do**
    $c_{sb \sqsubseteq sp} \leftarrow$ completed-axiom-set($sb \sqsubseteq sp$, $\mathcal{T}_r$, Or)
    $c_{\alpha \sqsubseteq \beta} \leftarrow c_{\alpha \sqsubseteq \beta} \cup c_{sb \sqsubseteq sp}$
  end for
end for
$\mathcal{T}_r \leftarrow$ Add-axioms($\mathcal{T}_r$,$\bigcup_{\alpha \sqsubseteq \beta} c_{\alpha \sqsubseteq \beta}$)
return $\mathcal{T}_r$

Algorithm C10 Weaken all wrong, remove all wrong, complete/add completed axiom sets one at a time

Input: TBox $\mathcal{T}$, Oracle Or, set of unwanted axioms W
Output: A repaired TBox

for each $\alpha \sqsubseteq \beta \in W$ **do**
  $\mathcal{T}_r \leftarrow$ Remove-axioms($\mathcal{T}$, $\{\alpha \sqsubseteq \beta\}$)
  $w_{\alpha \sqsubseteq \beta} \leftarrow$ weakened-axiom-set($\alpha \sqsubseteq \beta$, $\mathcal{T}_r$, Or)
end for
$\mathcal{T}_r \leftarrow$ Remove-axioms($\mathcal{T}$, W)
for each $\alpha \sqsubseteq \beta \in W$ **do**
  for each $sb \sqsubseteq sp \in w_{\alpha \sqsubseteq \beta}$ **do**
    $c_{sb \sqsubseteq sp} \leftarrow$ completed-axiom-set($sb \sqsubseteq sp$, $\mathcal{T}_r$, Or)
    $\mathcal{T}_r \leftarrow$ Add-axioms($\mathcal{T}_r$,$c_{sb \sqsubseteq sp}$)
  end for
end for
return $\mathcal{T}_r$

Algorithm C11 Weaken all wrong, remove/complete/add the wrong/completed axiom sets one at a time, remove all wrong at end

Input: TBox $\mathcal{T}$, Oracle Or, set of unwanted axioms W
Output: A repaired TBox

for each $\alpha \sqsubseteq \beta \in W$ **do**
  $\mathcal{T}_r \leftarrow$ Remove-axioms($\mathcal{T}$, $\{\alpha \sqsubseteq \beta\}$)
  $w_{\alpha \sqsubseteq \beta} \leftarrow$ weakened-axiom-set($\alpha \sqsubseteq \beta$, $\mathcal{T}_r$, Or)
end for
for each $\alpha \sqsubseteq \beta \in W$ **do**
  $\mathcal{T}_r \leftarrow$ Remove-axioms($\mathcal{T}$, $\{\alpha \sqsubseteq \beta\}$)
  for each $sb \sqsubseteq sp \in w_{\alpha \sqsubseteq \beta}$ **do**
    $c_{sb \sqsubseteq sp} \leftarrow$ completed-axiom-set($sb \sqsubseteq sp$, $\mathcal{T}_r$, Or)
    $\mathcal{T} \leftarrow$ Add-axioms($\mathcal{T}$,$c_{sb \sqsubseteq sp}$)
  end for
end for
return Remove-axioms($\mathcal{T}$,W)

Algorithm C12 Remove all wrong, weaken all, complete all, add completed axiom sets at end

Input: TBox $\mathcal{T}$, Oracle Or, set of unwanted axioms W
Output: A repaired TBox

$\mathcal{T}_r \leftarrow$ Remove-axioms($\mathcal{T}$, W)
for each $\alpha \sqsubseteq \beta \in W$ **do**
  $w_{\alpha \sqsubseteq \beta} \leftarrow$ weakened-axiom-set($\alpha \sqsubseteq \beta$, $\mathcal{T}_r$, Or)
end for
for each $\alpha \sqsubseteq \beta \in W$ **do**
  $c_{\alpha \sqsubseteq \beta} \leftarrow \emptyset$
  for each $sb \sqsubseteq sp \in w_{\alpha \sqsubseteq \beta}$ **do**
    $c_{sb \sqsubseteq sp} \leftarrow$ completed-axiom-set($sb \sqsubseteq sp$, $\mathcal{T}_r$, Or)
    $c_{\alpha \sqsubseteq \beta} \leftarrow c_{\alpha \sqsubseteq \beta} \cup c_{sb \sqsubseteq sp}$
  end for
end for
return Add-axioms($\mathcal{T}_r$,$\bigcup_{\alpha \sqsubseteq \beta} c_{\alpha \sqsubseteq \beta}$)

Algorithm C13 Remove all wrong, weaken/complete/add completed axiom sets one at a time

Input: TBox $\mathcal{T}$, Oracle Or, set of unwanted axioms W
Output: A repaired TBox

$\mathcal{T}_r \leftarrow$ Remove-axioms($\mathcal{T}$, W)
for each $\alpha \sqsubseteq \beta \in W$ **do**
  $w_{\alpha \sqsubseteq \beta} \leftarrow$ weakened-axiom-set($\alpha \sqsubseteq \beta$, $\mathcal{T}_r$, Or)
  for each $sb \sqsubseteq sp \in w_{\alpha \sqsubseteq \beta}$ **do**
    $c_{sb \sqsubseteq sp} \leftarrow$ completed-axiom-set($sb \sqsubseteq sp$, $\mathcal{T}_r$, Or)
    $\mathcal{T}_r \leftarrow$ Add-axioms($\mathcal{T}_r$,$c_{sb \sqsubseteq sp}$)
  end for
end for
return $\mathcal{T}_r$

Algorithm C14 Generate the justifications of each wrong axiom and validate all the asserted axioms from the generated justifications, weaken all wrong asserted axioms, remove all wrong asserted axioms, complete/add completed axiom set one at a time

Input: TBox $\mathcal{T}$, Oracle Or, set of unwanted axioms W
Output: A repaired TBox

$D \leftarrow \emptyset$
for each $\alpha \sqsubseteq \beta \in W$ **do**
  for each $J \in$ Justifications$(\alpha \sqsubseteq \beta)$ **do**
    for each $axiom \in J$ **do**
      if $Validate\_axiom(axiom, Or) = false$ **then**
        $D \leftarrow D \cup \{axiom\}$
      end if
    end for
  end for
end for
for each $\alpha \sqsubseteq \beta \in D$ **do**
  $\mathcal{T}_r \leftarrow$ Remove-axioms($\mathcal{T}$, $\{\alpha \sqsubseteq \beta\}$)
  $w_{\alpha \sqsubseteq \beta} \leftarrow$ weakened-axiom-set($\alpha \sqsubseteq \beta$, $\mathcal{T}_r$, Or)
end for
$\mathcal{T}_r \leftarrow$ Remove-axioms($\mathcal{T}_r$, D)
for each $\alpha \sqsubseteq \beta \in D$ **do**
  for each $sb \sqsubseteq sp \in w_{\alpha \sqsubseteq \beta}$ **do**
    $c_{sb \sqsubseteq sp} \leftarrow$ completed-axiom-set($sb \sqsubseteq sp$,$\mathcal{T}_r$, Or)
    $\mathcal{T}_r \leftarrow$ Add-axioms($\mathcal{T}_r$,$c_{sb \sqsubseteq sp}$)
  end for
end for
return $\mathcal{T}_r$

Algorithm C15 Generate the justifications of each wrong axiom and validate one hitting set from the generated justifications, weaken all wrong asserted axioms, remove all wrong asserted axioms, complete/add completed axiom set one at a time

Input: TBox $\mathcal{T}$, Oracle Or, set of unwanted axioms W
Output: A repaired TBox

1: $D \leftarrow \emptyset$
2: **for each** $\alpha \sqsubseteq \beta \in W$ **do**
3: **for each** $J \in$ Justifications$(\alpha \sqsubseteq \beta)$ **do**
4: **for each** $axiom \in J$ **do**
5: **if** $Validate\_axiom(axiom, Or) = false$ **then**
6: $D \leftarrow D \cup \{axiom\}$
7: **break**
8: **end if**
9: **end for**
10: **end for**
11: **end for**
12: **for each** $\alpha \sqsubseteq \beta \in D$ **do**
13: $\mathcal{T}_r \leftarrow$ Remove-axioms($\mathcal{T}$, $\{\alpha \sqsubseteq \beta\}$)
14: $w_{\alpha \sqsubseteq \beta} \leftarrow$ weakened-axiom-set($\alpha \sqsubseteq \beta$, $\mathcal{T}_r$, Or)
15: **end for**
16: $\mathcal{T}_r \leftarrow$ Remove-axioms($\mathcal{T}_r$,D)
17: **for each** $\alpha \sqsubseteq \beta \in D$ **do**
18: **for each** $sb \sqsubseteq sp \in w_{\alpha \sqsubseteq \beta}$ **do**
19: $c_{sb \sqsubseteq sp} \leftarrow$ completed-axiom-set($sb \sqsubseteq sp$,$\mathcal{T}_r$, Or)
20: $\mathcal{T}_r \leftarrow$ Add-axioms($\mathcal{T}_r$,$c_{sb \sqsubseteq sp}$)
21: **end for**
22: **end for**
23: **return** $\mathcal{T}_r$

Algorithm C16 Generate the justifications of each wrong axiom and validate one hitting set from the generated justifications, weaken all wrong asserted axioms, complete all weakened axioms, add completed axiom sets and remove all wrong asserted axioms at end

Input: TBox $\mathcal{T}$, Oracle Or, set of unwanted axioms W
Output: A repaired TBox

$D \leftarrow \emptyset$
for each $\alpha \sqsubseteq \beta \in W$ **do**
  for each $J \in$ Justifications$(\alpha \sqsubseteq \beta)$ **do**
    for each $axiom \in J$ **do**
      if $Validate\_axiom(axiom, Or) = false$ **then**
        $D \leftarrow D \cup \{axiom\}$
        break
      end if
    end for
  end for
end for
for each $\alpha \sqsubseteq \beta \in D$ **do**
  $w_{\alpha \sqsubseteq \beta} \leftarrow$ weakened-axiom-set$(\alpha \sqsubseteq \beta, \mathcal{T}, Or)$
end for
$c_{\alpha \sqsubseteq \beta} \leftarrow \emptyset$
for each $\alpha \sqsubseteq \beta \in D$ **do**
  for each $sb \sqsubseteq sp \in w_{\alpha \sqsubseteq \beta}$ **do**
    $c_{sb \sqsubseteq sp} \leftarrow$ completed-axiom-set$(sb \sqsubseteq sp, \mathcal{T}, Or)$
    $c_{\alpha \sqsubseteq \beta} \leftarrow c_{\alpha \sqsubseteq \beta} \cup c_{sb \sqsubseteq sp}$
  end for
end for
$\mathcal{T}_r \leftarrow$ Add-axioms$(\mathcal{T}, \bigcup_{\alpha \sqsubseteq \beta} c_{\alpha \sqsubseteq \beta})$
$\mathcal{T}_r \leftarrow$ Remove-axioms$(\mathcal{T}_r, D)$
return $\mathcal{T}_r$

Algorithm C17 Generate the justifications of each wrong axiom and validate all the asserted axioms from the generated justifications, remove all wrong asserted axioms, weaken all wrong asserted axioms, complete all the weakened axioms, then add completed axiom sets at end

Input: TBox $\mathcal{T}$, Oracle Or, set of unwanted axioms W
Output: A repaired TBox

$D \leftarrow \emptyset$
for each $\alpha \sqsubseteq \beta \in W$ **do**
    for each $J \in$ Justifications$(\alpha \sqsubseteq \beta)$ **do**
        for each $axiom \in J$ **do**
            if $Validate\_axiom(axiom, Or) = false$ **then**
                $D \leftarrow D \cup \{axiom\}$
            end if
        end for
    end for
end for
$\mathcal{T}_r \leftarrow$ Remove-axioms($\mathcal{T}_r$, D)
for each $\alpha \sqsubseteq \beta \in D$ **do**
    $w_{\alpha \sqsubseteq \beta} \leftarrow$ weakened-axiom-set($\alpha \sqsubseteq \beta$, $\mathcal{T}_r, Or$)
end for
$c_{\alpha \sqsubseteq \beta} \leftarrow \emptyset$
for each $\alpha \sqsubseteq \beta \in D$ **do**
    for each $sb \sqsubseteq sp \in w_{\alpha \sqsubseteq \beta}$ **do**
        $c_{sb \sqsubseteq sp} \leftarrow$ completed-axiom-set($sb \sqsubseteq sp$,$\mathcal{T}_r, Or$)
        $c_{\alpha \sqsubseteq \beta} \leftarrow c_{\alpha \sqsubseteq \beta} \cup c_{sb \sqsubseteq sp}$
    end for
end for
$\mathcal{T}_r \leftarrow$ Add-axioms($\mathcal{T}_r$,$\bigcup_{\alpha \sqsubseteq \beta} c_{\alpha \sqsubseteq \beta}$)
return $\mathcal{T}_r$

Algorithm C18 Generate the justifications of each wrong axiom and validate all the asserted axioms from the generated justifications, remove all wrong asserted axioms, weaken all wrong asserted axioms, complete/add completed axiom sets one at a time

Input: TBox $\mathcal{T}$, Oracle Or, set of unwanted axioms W
Output: A repaired TBox

$D \leftarrow \emptyset$
for each $\alpha \sqsubseteq \beta \in W$ **do**
  for each $J \in$ Justifications$(\alpha \sqsubseteq \beta)$ **do**
    for each $axiom \in J$ **do**
      if $Validate\_axiom(axiom, Or) = false$ **then**
        $D \leftarrow D \cup \{axiom\}$
      end if
    end for
  end for
end for
$\mathcal{T}_r \leftarrow$ Remove-axioms($\mathcal{T}_r$, D)
for each $\alpha \sqsubseteq \beta \in D$ **do**
  $w_{\alpha \sqsubseteq \beta} \leftarrow$ weakened-axiom-set($\alpha \sqsubseteq \beta$, $\mathcal{T}_r$, Or)
end for
for each $\alpha \sqsubseteq \beta \in D$ **do**
  $c_{\alpha \sqsubseteq \beta} \leftarrow \emptyset$
  for each $sb \sqsubseteq sp \in w_{\alpha \sqsubseteq \beta}$ **do**
    $c_{sb \sqsubseteq sp} \leftarrow$ completed-axiom-set($sb \sqsubseteq sp$,$\mathcal{T}_r$, Or)
    $\mathcal{T}_r \leftarrow$ Add-axioms($\mathcal{T}_r$,$c_{sb \sqsubseteq sp}$)
  end for
end for
return $\mathcal{T}_r$

Algorithm C19 Generate the justifications of each wrong axiom and validate all the asserted axioms from the generated justifications, remove all wrong, weaken/complete/add completed axiom sets one at a time

Input: TBox $\mathcal{T}$, Oracle Or, set of unwanted axioms W
Output: A repaired TBox

$D \leftarrow \emptyset$
for each $\alpha \sqsubseteq \beta \in W$ **do**
  for each $J \in$ Justifications$(\alpha \sqsubseteq \beta)$ **do**
    for each $axiom \in J$ **do**
      if $Validate\_axiom(axiom, Or) = false$ **then**
        $D \leftarrow D \cup \{axiom\}$
      end if
    end for
  end for
end for
$\mathcal{T}_r \leftarrow$ Remove-axioms$(\mathcal{T}, D)$
for each $\alpha \sqsubseteq \beta \in D$ **do**
  $w_{\alpha \sqsubseteq \beta} \leftarrow$ weakened-axiom-set$(\alpha \sqsubseteq \beta, \mathcal{T}_r, Or)$
  for each $sb \sqsubseteq sp \in w_{\alpha \sqsubseteq \beta}$ **do**
    $c_{sb \sqsubseteq sp} \leftarrow$ completed-axiom-set$(sb \sqsubseteq sp, \mathcal{T}_r, Or)$
    $\mathcal{T}_r \leftarrow$ Add-axioms$(\mathcal{T}_r, c_{sb \sqsubseteq sp})$
  end for
end for
return $\mathcal{T}_r$

Algorithm C20 Generate the justifications of each wrong axiom and validate all the asserted axioms from the generated justifications, weaken/complete/add completed axiom sets one at a time, remove all wrong at end

Input: TBox $\mathcal{T}$, Oracle Or, set of unwanted axioms W
Output: A repaired TBox

$D \leftarrow \emptyset$
for each $\alpha \sqsubseteq \beta \in W$ **do**
  for each $J \in$ Justifications$(\alpha \sqsubseteq \beta)$ **do**
    for each $axiom \in J$ **do**
      if $Validate\_axiom(axiom, Or) = false$ **then**
        $D \leftarrow D \cup \{axiom\}$
      end if
    end for
  end for
end for
for each $\alpha \sqsubseteq \beta \in D$ **do**
  $\mathcal{T}_r \leftarrow$ Remove-axioms$(\mathcal{T}, \{\alpha \sqsubseteq \beta\})$
  $w_{\alpha \sqsubseteq \beta} \leftarrow$ weakened-axiom-set$(\alpha \sqsubseteq \beta, \mathcal{T}_r, Or)$
  for each $sb \sqsubseteq sp \in w_{\alpha \sqsubseteq \beta}$ **do**
    $c_{sb \sqsubseteq sp} \leftarrow$ completed-axiom-set$(sb \sqsubseteq sp, \mathcal{T}_r, Or)$
    $\mathcal{T}_r \leftarrow$ Add-axioms$(\mathcal{T}_r, c_{sb \sqsubseteq sp})$
  end for
end for
$\mathcal{T}_r \leftarrow$ Remove-axioms$(\mathcal{T}, D)$
return $\mathcal{T}_r$

C. Experiments - data

In order to compare the use of the different combinations of strategies, we run experiments on several ontologies: Mini-GALEN, PACO, NCI, EKAW, OFSMR and Pizza ontology. Mini-GALEN is an example inspired by the GALEN (from https://bioportal.bioontology.org/) ontology and is presented in the paper; PACO, NCI, OFSMR are available at https://bioportal.bioontology.org/, EKAW from the conference track of http://oaei.ontologymatching.org/ and the Pizza ontology is available at https://github.com/owlcs/pizza-ontology. We made the versions of the ontologies that we used available at https://figshare.com/s/f3b9472a7e5dd69237dc. We have used the parts of these ontologies that are expressible in $\mathcal{EL}_\bot$ in the sense that we removed the parts of axioms that used constructors not in $\mathcal{EL}_\bot$.

An overview of the numbers of concepts, roles and axioms in these ontologies is given in Table 9.

Table 9
Ontologies

	Mini-GALEN	Pizza	EKAW	OFSMR	PACO	NCI
Concepts	9	74	100	159	224	3304
Roles	1	33	8	2	23	1
Axioms	20	341	801	1517	1153	30364

D. Experiments - results

In this part, we give the full results of the comparative experiments run in different ontologies (Tables 11-38). Table 10 lists the wrong asserted axioms we introduced in each test ontology for the experiments. These wrong axioms were generated by replacing existing axioms with axioms where their left/right-hand side concepts were changed.

Table 11 shows the influence of using different combinations in the debugging step for the EKAW ontology.

Tables 12-14 show the influence of using different combinations on the weakening results for Mini-GALEN using Algorithms C1-C13.

Tables 15-18 show the influence of removing wrong axioms one at a time in different orders on the sizes of the sub-and super-concepts sets for weakening. We used Mini-GALEN and Algorithm C2.

Tables 19-23 show the sizes of the super- and sub-concepts sets for weakening the PACO, EKAW, NCI, Pizza and OFSMR ontologies using Algorithms C1-C4.

Tables 24-29 show the differences between using source/target sets (not introducing equivalences in the ontologies) and using sup/sub sets for completing. We present the sizes of these sets as well as the results of completing for Mini-GALEN and Algorithms C5-C7 in Tables 24-25. Further, the sizes of these sets are listed in Tables 26-29 for the NCI ontology and Algorithms C5-C13. For the remaining ontologies, we list the sizes of the source and target sets to generate the completed axioms using Algorithms C5-C13 in Tables 30-37.

Table 38 shows the results of using debugging, removing, weakening and completing when repairing Mini-GALEN using Algorithms C14-C20.

Table 10

Wrong axioms in each ontology.

Ontology	Wrong axioms(W)	Wrong asserted axioms used in the examples
Mini-GALEN	PathologicalProcess⊑GranulomaProcess, Endocarditis⊑GranulomaProcess	PathologicalProcess⊑InflammationProcess, InflammationProcess⊑GranulomaProcess, PathologicalProcess⊑GranulomaProcess, Endocarditis⊑PathologicalProcess
PACO	clearing walking⊑⊥, Polish_car⊑Home_improvement_maintenance, Washing_windows⊑ Home_improvement_maintenance, Moderate⊑Speed Per_week⊑By_duration, Washing_car⊑ Home_improvement_maintenance	Polish_car⊑Home_improvement_maintenance, Washing_windows ⊑ Home_improvement_maintenance, Moderate⊑Speed, Per_week⊑By_duration, Washing_car⊑ Home_improvement_maintenance, Walking⊑Daily_living_activity
EKAW	Tutorial ⊑⊥, Camera_Ready_Paper⊑∃writtenBy.Student, Invited_Talk_Abstract⊑Paper, Programme_Brochure⊑⊥	Camera_Ready_Paper⊑∃writtenBy.Student, Tutorial⊑Conference, Tutorial⊑Conference_session, Invited_Talk_Abstract⊑Paper, Programme_Brochure⊑Flyer
NCI	Inner_Enamel_Epithelium ⊑⊥, Foramen_Apicis_Dentis ⊑⊥, Stratum_Intermedium ⊑⊥, Enamel ⊑⊥, Secretory-stage_Ameloblast ⊑⊥, Canalis_Radicis_Dentis ⊑⊥, Stellate_Reticulum ⊑⊥, Ameloblast ⊑⊥, Cementum ⊑⊥, Cementocyte ⊑⊥, Odontoblast ⊑⊥, Dental_Pulp ⊑⊥, Outer_Enamel_Epithelium ⊑⊥, Tooth_Cell ⊑⊥, Dentin ⊑⊥, Tooth_Tissue ⊑⊥, Maturation-stage_Ameloblast ⊑⊥, Cementoblast ⊑⊥, Red_fiber⊑Connective_tissue_fiber, Eye_lid⊑Cheek	Tooth_tissue⊑Tooth, Red_fiber⊑Connective_tissue_fiber, Eye_lid⊑Cheek
Pizza	Ice cream⊑⊥, PineKernels⊑VegetableTopping, PeperoniSausageTopping⊑ PeperonataTopping, RosemaryTopping⊑VegetableTopping	PineKernels⊑VegetableTopping, PeperoniSausageTopping⊑PeperonataTopping, RosemaryTopping⊑VegetableTopping IceCream⊑ ∃hasTopping.FruitTopping, RosemaryTopping⊑VegetableTopping
OFSMR	Beverage ⊑⊥, Bread⊑Procesed_fruit_and_vegetables, Pasta⊑Procesed_fruit_and_vegetables	Beverage⊑Food, Bread⊑Procesed_fruit_and_vegetables, Pasta⊑Procesed_fruit_and_vegetables

Table 11

Debugging for EKAW, W = {Tutorial $\sqsubseteq \bot$, Camera_Ready_Paper $\sqsubseteq \exists$writtenBy.Student, Invited_Talk_Abstract $\sqsubseteq$ Paper, Programme_Brochure $\sqsubseteq \bot$ }. For D-one-v there are different solutions based on the generated Hitting set (but one may not always be able to find a repair after validation).

	D-v-all	D-one-v(1)	D-one-v(2)
Asserted Wrong Axioms	Tutorial$\sqsubseteq$Conference, Tutorial$\sqsubseteq$Conference_Session, Programme_Brochure$\sqsubseteq$Flyer, Camera_Ready_Paper$\sqsubseteq$ $\exists$writtenBy.Student, Invited_Talk_Abstract$\sqsubseteq$Paper	Tutorial$\sqsubseteq$Conference_Session, Programme_Brochure$\sqsubseteq$Flyer, Camera_Ready_Paper$\sqsubseteq$ $\exists$writtenBy.Student, Invited_Talk_Abstract$\sqsubseteq$Paper	Programme_Brochure$\sqsubseteq$Flyer, Camera_Ready_Paper$\sqsubseteq$ $\exists$writtenBy.Student, Invited_Talk_Abstract$\sqsubseteq$Paper, Tutorial$\sqsubseteq$Conference

Table 12

Weakening for Mini-GALEN using Algorithms C1-C4. Four wrong asserted axioms give 4 sup/sub-sets respectively per algorithm. The wrong axioms set W={PPr $\sqsubseteq$ GPr, E $\sqsubseteq$ GPr}.

	C1	C2	C3	C4
Wrong asserted axioms	PPr $\sqsubseteq$ IPr, IPr $\sqsubseteq$ GPr, E $\sqsubseteq$PPr, PPr $\sqsubseteq$ GPr	PPr $\sqsubseteq$ IPr, IPr $\sqsubseteq$ GPr, E $\sqsubseteq$PPr, PPr $\sqsubseteq$ GPr	PPr $\sqsubseteq$ IPr, IPr $\sqsubseteq$ GPr, E $\sqsubseteq$PPr, PPr $\sqsubseteq$ GPr	PPr $\sqsubseteq$ IPr, IPr $\sqsubseteq$ GPr, E $\sqsubseteq$PPr, PPr $\sqsubseteq$ GPr
Sup(β,$\mathcal{T}$) Sub(α,$\mathcal{T}$)	3 2 4 2 2 3 1 2	2 2 4 2 1 2 1 1	1 2 1 2 1 1 1 1	2 2 1 2 1 1 1 1
Weakened	IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr	IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr	IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr	IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr

Table 13

Weakening for Mini-GALEN using Algorithms C5-C9. Four wrong asserted axioms give 4 sup/sub-sets respectively per algorithm. The wrong axioms set W={PPr $\sqsubseteq$ GPr, E $\sqsubseteq$ GPr}.

	C5	C6	C7	C8	C9
Wrong asserted axioms	PPr $\sqsubseteq$ GPr, IPr $\sqsubseteq$ GPr, PPr $\sqsubseteq$ IPr, E $\sqsubseteq$ PPr	PPr $\sqsubseteq$ GPr, IPr $\sqsubseteq$ GPr, PPr $\sqsubseteq$ IPr, E $\sqsubseteq$ PPr	PPr $\sqsubseteq$ GPr, IPr $\sqsubseteq$ GPr, PPr $\sqsubseteq$ IPr, E $\sqsubseteq$ PPr	PPr $\sqsubseteq$ GPr, IPr $\sqsubseteq$ GPr, PPr $\sqsubseteq$ IPr, E $\sqsubseteq$ PPr	PPr $\sqsubseteq$ GPr, IPr $\sqsubseteq$ GPr, PPr $\sqsubseteq$ IPr, E $\sqsubseteq$ PPr
Sub(α,$\mathcal{T}$) Sup(β,$\mathcal{T}$)	2 3 2 1 2 2 3 4	2 4 4 1 2 3 4 4	2 4 4 1 2 3 3 2	2 3 2 1 2 2 3 4	2 3 2 1 2 2 3 4
Weakened	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr	IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr	IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr

Table 14

Weakening for Mini-GALEN using Algorithms C10-C13. Four wrong asserted axioms give 4 sup/sub-sets respectively per algorithm. The wrong axioms set W={PPr $\sqsubseteq$ GPr, E $\sqsubseteq$ GPr}.

	C10	C11	C12	C13
Wrong asserted axioms	PPr $\sqsubseteq$ GPr, IPr $\sqsubseteq$ GPr, PPr $\sqsubseteq$ IPr, E $\sqsubseteq$ PPr	PPr $\sqsubseteq$ GPr, IPr $\sqsubseteq$ GPr, PPr $\sqsubseteq$ IPr, E $\sqsubseteq$ PPr	PPr $\sqsubseteq$ GPr, IPr $\sqsubseteq$ GPr, PPr $\sqsubseteq$ IPr, E $\sqsubseteq$ PPr	PPr $\sqsubseteq$ GPr, IPr $\sqsubseteq$ GPr, PPr $\sqsubseteq$ IPr, E $\sqsubseteq$ PPr
Sub(α,$\mathcal{T}$) Sup(β,$\mathcal{T}$)	2 3 2 1 2 2 3 4	2 3 2 1 2 2 3 4	1 1 1 1 2 2 1 1	1 1 2 1 2 2 3 2
Weakened	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr	IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr

Table 15

Removing wrong axioms: ①PPr⊆IPr ②IPr⊆GPr ③PPr⊆GPr ④E⊆PPr in different order for Mini-GALEN by Algorithm C2 (Removing ④ first).

Wrong Axiom	①→②→③	②→①→③	②→③→①	①→③→②	③→①→②	③→②→①
Sub(α,$\mathcal{T}$)	1 1 1 1	1 2 1 1	1 2 1 1	1 1 1 1	1 1 1 1	1 1 1 1
Sup(β,$\mathcal{T}$)	4 3 2 2	4 2 1 2	4 2 2 2	4 3 2 2	4 2 3 2	4 2 2 2

Table 16

Removing wrong axioms: ①PPr⊆IPr ②IPr⊆GPr ③PPr⊆GPr ④E⊆PPr in different order for Mini-GALEN by Algorithm C2 (Removing ③ first).

Wrong Axiom	④→①→②	④→②→①	②→④→①	①→④→②	②→①→④	①→②→④
Sub(α,$\mathcal{T}$)	2 1 1 1	2 1 2 1	2 3 1 1	2 2 1 1	2 3 2 1	2 2 1 1
Sup(β,$\mathcal{T}$)	2 4 3 2	2 4 2 2	2 2 2 2	2 3 2 2	2 2 2 2	2 3 2 2

Table 17

Removing wrong axioms: ①PPr⊆IPr ②IPr⊆GPr ③PPr⊆GPr ④E⊆PPr in different order for Mini-GALEN by Algorithm C2 (Removing ② first).

Wrong Axiom	④→③→①	④→①→③	③→④→①	①→④→③	③→①→④	①→③→④
Sub(α,$\mathcal{T}$)	3 1 1 1	3 1 1 1	3 2 1 1	3 2 1 1	3 2 1 1	3 2 1 1
Sup(β,$\mathcal{T}$)	2 3 2 2	2 3 2 2	2 2 2 2	2 2 3 2	2 2 2 2	2 2 2 2

Table 18

Removing wrong axioms: ①PPr⊆IPr ②IPr⊆GPr ③PPr⊆GPr ④E⊆PPr in different order for Mini-GALEN by Algorithm C2 (Removing ① first).

Wrong Axiom	④→③→②	④→②→③	②→④→③	③→④→②	②→③→④	③→②→④
Sub(α,$\mathcal{T}$)	2 1 1 1	2 1 1 1	2 1 1 1	2 2 1 1	2 1 2 1	2 2 1 1
Sup(β,$\mathcal{T}$)	3 3 2 2	3 3 2 2	3 2 3 2	3 2 2 2	3 2 2 2	3 2 2 2

Table 19

Weakening the PACO ontology using Algorithms C1-C4. Six wrong axioms give 6 sup/sub-sets per algorithm.

	C1	C2	C3	C4
Sup(β,$\mathcal{T}$)	4 4 4 3 4 3	4 4 4 3 4 3	4 4 4 3 4 3	4 4 4 3 4 3
Sub(α,$\mathcal{T}$)	1 1 1 6 1 1	1 1 1 6 1 1	1 1 1 6 1 1	1 1 1 6 1 1

Table 20

Weakening the EKAW ontology using Algorithms C1-C4. Four wrong axioms give 4 sup/sub-sets per algorithm.

	C1	C2	C3	C4
Sup(β,$\mathcal{T}$)	3 4 3 3	3 4 3 3	3 4 3 3	3 4 3 3
Sub(α,$\mathcal{T}$)	1 1 1 1	1 1 1 1	1 1 1 1	1 1 1 1

Table 21

Weakening the NCI ontology using Algorithms C1-C4. Three wrong axioms give 3 sup/sub-sets per algorithm.

	C1	C2	C3	C4
Sup(β,$\mathcal{T}$)	13 15 8	13 15 8	13 15 8	13 15 8
Sub(α,$\mathcal{T}$)	7 1 3	7 1 3	7 1 3	7 1 3

Table 22

Weakening the Pizza ontology using Algorithms C1-C4. Four wrong axioms give 4 sup/sub-sets per algorithm.

	C1	C2	C3	C4
Sup(β,$\mathcal{T}$)	4 8 4 8	4 8 4 8	4 8 4 8	4 8 4 8
Sub(α,$\mathcal{T}$)	1 1 1 1	1 1 1 1	1 1 1 1	1 1 1 1

Table 23

Weakening the OFSMR ontology using Algorithms C1-C4. Three wrong axioms give 3 sup/sub-sets per algorithm.

	C1	C2	C3	C4
Sup(β,$\mathcal{T}$)	2 4 4	2 4 4	2 4 4	2 4 4
Sub(α,$\mathcal{T}$)	2 1 1	2 1 1	2 1 1	2 1 1

Table 24

Completing the Mini-GALEN ontology using Algorithms C5-C7 by excluding concepts that would introduce equivalence relations in the ontology

	C5	C6	C7
Weakened	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr
Source(α,$\mathcal{T}$) Target(β,$\mathcal{T}$)	3 2 3 3 2 3	3 1 1 3 1 3	3 1 1 3 1 3
Completed	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr, GPr $\sqsubseteq$ IPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ PPr, GPr $\sqsubseteq$ IPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ PPr, GPr $\sqsubseteq$ IPr

Table 25

Completing the Mini-GALEN ontology using Algorithms C5-C7.

	C5	C6	C7
Weakened	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr
Sup(α,$\mathcal{T}$) Sub(β,$\mathcal{T}$)	4 3 4 5 5 5	4 1 4 5 4 5	4 1 4 5 4 5
Completed	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ PPr, GPr $\sqsubseteq$ IPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ PPr, GPr $\sqsubseteq$ IPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ PPr, GPr $\sqsubseteq$ IPr

Table 26

Completing the NCI ontology using Algorithms C5-C9.

	C5	C6	C7	C8	C9
Sup(α,$\mathcal{T}$)	3 1 1	3 1 1	3 1 1	15 16 9	3 1 1
Sub(β,$\mathcal{T}$)	41 2143 83	41 2143 83	41 2136 76	66 2144 86	41 2133 76

Table 27

Completing the NCI ontology using Algorithms C10-C13.

	C10	C11	C12	C13
Sup(α,$\mathcal{T}$)	3 1 1	3 1 1	3 1 1	3 1 1
Sub(β,$\mathcal{T}$)	41 2136 76	41 2143 83	41 2133 76	41 2133 76

Table 28

Completing the NCI ontology using Algorithms C5-C9 by excluding concepts that would introduce equivalence relations in the ontology.

	C5	C6	C7	C8	C9
Source	3 1 1	3 1 1	3 1 1	6 14 6	3 1 1
Target	40 2143 83	40 2143 83	40 2136 76	59 2143 83	40 2133 76

Table 29

Completing the NCI ontology using Algorithms C10-C13 by excluding concepts that would introduce equivalence relations in the ontology.

	C10	C11	C12	C13
Source	3 1 1	3 1 1	3 1 1	3 1 1
Target	40 2136 76	40 2143 83	40 2133 76	40 2133 76

Table 30

Completing the PACO ontology using Algorithms C5-C9 by excluding concepts that would introduce equivalence relations in the ontology.

	C5	C6	C7	C8	C9
Source	1 1 1 1 1 1	1 1 1 1 1 1	1 1 1 1 1 1	2 2 2 2 2 3	1 1 1 1 1 1
Target	59 59 59 171 40 40	59 59 59 171 40 40	59 59 59 171 40 40	59 59 59 171 40 40	51 51 51 168 39 39

Table 31

Completing the PACO ontology using Algorithms C10-C13 by excluding concepts that would introduce equivalence relations in the ontology.

	C10	C11	C12	C13
Source	1 1 1 1 1 1	1 1 1 1 1 1	1 1 1 1 1 1	1 1 1 1 1 1
Target	51 52 53 170 39 40	59 59 59 171 40 40	51 51 51 168 39 39	51 52 53 170 39 40

Table 32

Completing of the EKAW ontology using Algorithms C5-C9 by excluding concepts that would introduce equivalence relations in the ontology.

	C5	C6	C7	C8	C9
Source	9 1 1 1	9 1 1 1	9 1 1 1	10 2 2 2	9 1 1 1
Target	23 17 34 34	23 17 34 34	23 17 34 34	23 17 34 34	23 17 33 33

Table 33

Completing of the EKAW ontology using Algorithms C10-C13 by excluding concepts that would introduce equivalence relations in the ontology.

	C10	C11	C12	C13
Source	9 1 1 1	9 1 1 1	9 1 1 1	9 1 1 1
Target	23 17 33 34	23 17 34 34	23 17 33 33	23 17 33 34

Table 34

Completing by the Pizza ontology using Algorithms C5-C9 by excluding concepts that would introduce equivalence relations in the ontology.

	C5	C6	C7	C8	C9
Source	1 1 3 3	1 1 3 3	1 1 3 3	2 7 4 6	1 1 3 3
Target	50 147 50 50	50 147 50 50	50 147 50 50	50 147 50 50	50 144 48 48

Table 35

Completing the Pizza ontology using Algorithms C10-C13 by excluding concepts that would introduce equivalence relations in the ontology.

	C10	C11	C12	C13
Source	1 1 3 3	1 1 3 3	1 1 3 3	1 1 3 3
Target	49 147 48 50	50 147 50 50	18 144 48 48	48 145 49 50

Table 36

Completing the OFSMR ontology using Algorithms C5-C9 by excluding concepts that would introduce equivalence relations in the ontology.

	C5	C6	C7	C8	C9
Source	1 1 1	1 1 1	1 1 1	2 3 3	1 1 1
Target	125 125 125	125 125 125	125 125 125	125 125 125	123 122 122

Table 37

Completing the OFSMR ontology using Algorithms C10-C13 by excluding concepts that would introduce equivalence relations in the ontology.

	C10	C11	C12	C13
Source	1 1 1	1 1 1	1 1 1	1 1 1
Target	123 122 123	125 125 125	123 122 122	123 122 123

Table 38

The result when applying Algorithms C14-C20 on Mini-GALEN. The wrong axioms set W={PPr $\sqsubseteq$ GPr, E $\sqsubseteq$ GPr}.

	C14	C15	C16	C17	C18	C19	C20
Wrong asserted axioms	PPr $\sqsubseteq$ GPr, IPr $\sqsubseteq$ GPr, PPr $\sqsubseteq$ IPr, E $\sqsubseteq$ PPr	PPr $\sqsubseteq$ GPr, IPr $\sqsubseteq$ GPr, E $\sqsubseteq$ GPr	PPr $\sqsubseteq$ GPr, IPr $\sqsubseteq$ GPr, PPr $\sqsubseteq$ IPr, E $\sqsubseteq$ PPr	PPr $\sqsubseteq$ GPr, IPr $\sqsubseteq$ GPr, PPr $\sqsubseteq$ IPr, E $\sqsubseteq$ PPr	PPr $\sqsubseteq$ GPr, IPr $\sqsubseteq$ GPr, PPr $\sqsubseteq$ IPr, E $\sqsubseteq$ PPr	PPr $\sqsubseteq$ GPr, IPr $\sqsubseteq$ GPr, PPr $\sqsubseteq$ IPr, E $\sqsubseteq$ PPr	PPr $\sqsubseteq$ GPr, IPr $\sqsubseteq$ GPr, PPr $\sqsubseteq$ IPr, E $\sqsubseteq$ PPr
Sub(α,$\mathcal{T}$) Sup(β,$\mathcal{T}$)	2 3 2 1 2 2 3 4	2 3 1 2 2 4	2 3 2 1 2 2 3 4	1 1 1 1 2 2 1 1	1 1 1 1 2 2 1 1	1 1 2 1 2 2 3 2	2 4 4 1 2 3 4 4
Weakened	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr	IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr	IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr
Sup(α,$\mathcal{T}$) Sub(β,$\mathcal{T}$)	1 1 2 2 3 4	1 1 2 3	4 3 4 5 5 5	1 1 2 2	1 1 2 3	1 1 2 2 3 4	4 1 4 5 4 5
Completed	IPr $\sqsubseteq$ PPr, PPr $\sqsubseteq$ NPr	IPr $\sqsubseteq$ PPr, PPr $\sqsubseteq$ NPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ PPr, GPr $\sqsubseteq$ IPr	IPr $\sqsubseteq$ NPr, PPr $\sqsubseteq$ NPr	IPr $\sqsubseteq$ PPr, PPr $\sqsubseteq$ NPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ PPr	PPr $\sqsubseteq$ NPr, IPr $\sqsubseteq$ PPr, GPr $\sqsubseteq$ IPr