




Optimal Repairs in the Description Logic \mathcal{EL} Revisited

Franz Baader^{1,2} , Patrick Koopmann¹ , and Francesco Kriegel¹ 

¹ Theoretical Computer Science, Technische Universität Dresden, Dresden, Germany
`firstname.lastname@tu-dresden.de`

² Center for Scalable Data Analytics and Artificial Intelligence (ScaDS.AI)
Dresden/Leipzig, Germany

Abstract. Ontologies based on Description Logics may contain errors, which are usually detected when reasoning produces consequences that follow from the ontology, but do not hold in the modelled application domain. In previous work, we have introduced repair approaches for \mathcal{EL} ontologies that are optimal in the sense that they preserve a maximal amount of consequences. In this paper, we will, on the one hand, review these approaches, but with an emphasis on motivation rather than on technical details. On the other hand, we will describe new results that address the problems that optimal repairs may become very large or need not even exist unless strong restrictions on the terminological part of the ontology apply. We will show how one can deal with these problems by introducing concise representations of optimal repairs.

1 Introduction

Description Logics (DLs) [4, 5] are a prominent family of logic-based knowledge representation formalisms, which offer a good compromise between expressiveness and the complexity of reasoning and are the formal basis for the Web ontology language OWL.³ In a DL ontology, the important notions of the application domain are introduced as background knowledge in the *terminology* (*TBox*), and then these notions are used to represent a specific application situation in the *ABox*. The DLs of the \mathcal{EL} family have drawn considerable attention since their reasoning problems are tractable [3], but they are nevertheless expressive enough to represent ontologies in many application domains, such as biology and medicine.⁴ For instance, the medical ontology SNOMED CT employs \mathcal{EL} and contains the following *concept inclusion* (*CI*) in its TBox:

$$\begin{aligned} \text{Common_cold} \sqsubseteq & \text{Disease} \sqcap \exists \text{causative_agent. Virus} \\ & \sqcap \exists \text{finding_site. Upper_respiratory_tract_structure} \\ & \sqcap \exists \text{pathological_process. Infectious_process,} \end{aligned}$$

³ <https://www.w3.org/TR/owl2-overview/>

⁴ see. e.g., <https://bioportal.bioontology.org> and <https://www.snomed.org/>

which says that a common cold is a disease that is caused by a virus, can be found in the upper respiratory tract, and has as pathological process an infectious process. A GP can then employ this concept to store in the ABox that patient Alice is diagnosed with common cold using the *concept assertion* $(\exists \text{has_diagnosis. Common_cold})(\text{alice})$. The GP’s ABox may also contain the information that Charles is Alice’s father, expressed as *role assertion* $\text{has_father}(\text{alice}, \text{charles})$, which might be of interest in the context of hereditary diseases.

Like all large human-made digital artefacts, the ontologies employed in such applications may contain errors, and this problem gets even worse if parts of the ontology (usually the ABox) are automatically generated by inexact methods based on information retrieval or machine learning. Errors in ontologies are often detected when the reasoner generates a consequence that formally follows from the knowledge base, but is incorrect in the sense that it does not hold in the application domain that is supposed to be modelled. For example, in a previous version of SNOMED CT, the concept “Amputation of finger” was classified as a subconcept of “Amputation of hand,” which is fortunately wrong in the real world. To correct such errors in large ontologies, the *knowledge engineer (KE)* should be supported by an appropriate *repair tool*. Such a tool receives as input one or more consequences of the given ontology that are unwanted, and it should return one or more repaired ontologies that no longer have these consequences (called *repairs*). The KE can then choose one of the computed repairs and either use it as the new ontology, or continue the repair process from it if other unwanted consequences are detected. Of course, it makes no sense to use as a repair an arbitrary ontology that does not have the unwanted consequences. The repaired ontology should (a) not introduce new information and (b) be as close as possible to the original ontology. There are different possibilities for how to formalize these conditions.

The *classical approaches* for ontology repair return maximal subsets of the ontology that do not have the unwanted consequence, and employ methods inspired by model-based diagnosis [33] to compute these sets [17, 32, 34]. Thus, these approaches interpret the above conditions in a syntactic way: (a) is read as “no new axioms” and (b) is realized by the maximality condition. In [15] we called classical repairs that satisfy this maximality condition *optimal classical repairs*. While these approaches preserve as many of the axioms in the ontology as possible, they need not preserve a maximal amount of consequences, and they are syntax-dependent. For example, consider the ABoxes $\mathcal{A} := \{(A \sqcap B)(a)\}$ and $\mathcal{B} := \{A(a), B(a)\}$, which both say that individual a belongs to the concepts A and B , and are thus equivalent. However, with respect to the unwanted consequence $A(a)$, the ABox \mathcal{A} has the empty ABox as only optimal classical repair, whereas \mathcal{B} has the optimal classical repair $\{B(a)\}$. Thus, the latter repair retains the consequence $B(a)$, whereas the former does not. To overcome this problem, more gentle repair approaches have been introduced, e.g., in [15, 21, 23, 26, 35]. The basic idea underlying these approaches is to replace some axioms of the ontology by weaker ones, rather than just removing them, as in the classical

approach. In our example, one can replace the axiom $(A \sqcap B)(a)$ in the ABox \mathcal{A} with the weaker axiom $B(a)$, and thus retain the consequence $B(a)$ even if one starts with \mathcal{A} rather than \mathcal{B} . However, these *gentle repairs* are still dependent on the syntactic structure of the axioms in the ontology, and how well they realize condition (b) depends on the employed weakening relation between axioms and the strategy used to apply it.

Providing the KE with syntax-dependent repair tools is not in line with the *functional approach* to knowledge representation [18,27] adopted by DLs. In this approach, the syntactic structure of the axioms in the ontology is supposed to be irrelevant. What counts is what queries are entailed by the ontology, which in DLs are usually *instance queries (IQ)* or *conjunctive queries (CQ)*. In this functional setting, (a) should be read as “no new consequences” (expressed in the adopted query formalism) and (b) as preserving a maximal set of such consequences. This leads us to the definition of an *optimal repair* [7,15], which is an ontology that does not have the unwanted consequences, is entailed by the original ontology (thus realizing property (a)), and preserves a maximal amount of consequences in the sense that there is no repair (i.e., no ontology satisfying the first two properties) that strictly entails it (property (b)). Entailment can be *IQ-entailment* or *CQ-entailment*, depending on whether we are interested only in instance queries, or also in conjunctive queries [28]. Maximizing the retained consequences is also motivated by the following observation. All the repair tool knows is the original ontology and the consequences that should be removed, which are specified in what we call a *repair request*. If it were to remove more consequences than are strictly needed to satisfy the repair request, then the decision which additional consequences to remove would be a random choice by the tool, not based on any application knowledge, which is held by the KE. In case the optimal repair retains consequences that should be removed, the KE needs to specify this in a subsequent repair request.

If a *repair problem* consisting of an ontology and a repair request does not have a repair, then it cannot have an optimal one. In general, however, optimal repairs of repair problems that have a repair need not exist either, even in the simple setting of \mathcal{EL} ABoxes without a TBox. This is illustrated in the following example, where the ABox $\mathcal{A} = \{V(n), \ell(n, n)\}$ says that Narcissus is a vain individual that loves itself, and the repair request $\mathcal{R} = \{V(n)\}$ wants us to remove the consequence that Narcissus is vain. Intuitively, to obtain a repair, we must remove $V(n)$. However, since all assertions of the form $\exists \ell.(V \sqcap (\exists \ell.)^k \top)(n)$, saying that Narcissus loves a vain individual that is the starting point of a loves-chain of length k , are consequences of \mathcal{A} and can be added to $\{\ell(n, n)\}$ without entailing $V(n)$, it is easy to see that there is no finite \mathcal{EL} ABox that is an optimal repair. In fact, since Narcissus is no longer vain, the retained cycle $\ell(n, n)$ cannot be used to generate the loves-chains of arbitrary length starting from a vain individual. Even if a given repair problem has optimal repairs, they may not *cover all repairs* in the sense that every repair is entailed by an optimal one. To see this, we can look at a modified version of the above example. Consider the ABox $\mathcal{B} = \{k(t, n), V(n), \ell(n, n)\}$, which contains the

additional information that Tiresias knows Narcissus, and the repair request $\mathcal{Q} = \{(\exists k.V)(t)\}$. Removing $k(t, n)$ from \mathcal{B} yields an optimal repair. However, there are also repairs that retain this assertion, but there is no optimal one among them for the same reason as in the previous example. Thus, if the KE is only offered the optimal repair $\{V(n), \ell(n, n)\}$ by the repair tool, the repair options that retain the assertion $k(t, n)$ are missed. This illustrates that the use of optimal repairs in a repair tool requires a setting where the optimal repairs always cover all repairs.

This can be achieved by using a more general notion of ABoxes, called *quantified ABoxes (qABoxes)* [16], where in addition to the usual named individuals we also have anonymous objects, which are represented as (existentially quantified) variables. In our Narcissus example, an optimal repair of \mathcal{A} for \mathcal{R} is obtained by removing $V(n)$ and introducing an anonymous vain and self-loving lover of Narcissus, which yields the qABox $\exists\{x\}. \{\ell(n, n), \ell(n, x), \ell(x, n), \ell(x, x), V(x)\}$. Note that we could not have used a named individual b instead of the variable x since then the resulting ABox would have entailed instance relationships for b , such as $V(b)$, that are not entailed by \mathcal{A} . One might think that retaining a consequence like $(\exists \ell.V)(n)$ is not justified since one of the reasons for this being a consequence of \mathcal{A} , namely $V(n)$, has been removed. However, with this argument, we would be back at the classical repair approach. As argued above, since the repair request only specifies that $V(n)$ should no longer be a consequence, other consequences like $(\exists \ell.V)(n)$ should not be lost unless this is needed to remove $V(n)$.

In [16] we consider a setting where ontologies are qABoxes and the repair requests consist of entailed \mathcal{EL} instance relationships.⁵ Given such a repair problem, we show how to construct a finite set of repairs, called the *canonical repairs*, which cover all repairs. The canonical repairs are of exponential size, and there may be exponentially many of them. Not every canonical repair is optimal, but due to the covering property, the set of them contains all optimal repairs up to equivalence. The set of optimal repairs can thus be obtained by removing non-optimal canonical repairs, i.e., ones that are strictly entailed by another canonical repair, and this set covers all repairs. The construction of the canonical repairs is actually the same for the CQ and the IQ case. The only difference is that, when removing non-optimal canonical repairs, the respective entailment relation must be used. Since CQ-entailment implies IQ-entailment, but not vice versa, more canonical repairs may be removed as non-optimal in the IQ setting. In addition, since CQ-entailment is NP-complete and IQ-entailment is tractable, the complexity of removing non-optimal repairs is higher in the CQ case.

The differences between the CQ and the IQ case get more pronounced if we add an \mathcal{EL} TBox. In [7], we assume that this *TBox is correct*, and thus should not be changed in the repair process. In order to adapt the approach and the results of [16] to this setting, the first step is to *saturate* the given qABox w.r.t.

⁵ The paper [16] actually calls repairs “compliant anonymisations” and repair requests “privacy policies” since it considers a situation where consequences are to be removed not because they are incorrect, but since this information should be hidden.

the TBox, to reduce entailment with TBox to entailment without TBox. For the IQ case, such a saturation always exists and can be computed in polynomial time. For the CQ case, a finite saturation need not exist in general. However, for *cycle-restricted TBoxes* [2], it always exists, but may be of exponential size. Continuing the repair process with the saturated qABox, we still need to take the TBox into account when defining canonical repairs, to ensure that consequences that have been removed from the qABox cannot be reintroduced by the TBox. With this adapted notion of canonical repairs, we obtain the same results as for the case without TBox. The canonical repairs cover all repairs and can be computed in exponential time. From them the set of all optimal repairs can be obtained by removing non-optimal ones using entailment test [7]. This works both for the IQ and the CQ case, but in the latter only if we can compute a finite saturation, which is always the case if the TBox is cycle-restricted. For TBoxes that are not cycle-restricted, optimal repairs need not exist in the CQ case. For example, with respect to the TBox $\{V \sqsubseteq \exists \ell. V, \exists \ell. V \sqsubseteq V\}$, which says that vain individuals are exactly the ones that love a vain individual, the qABox $\{V(n)\}$ does not have an optimal repair for the repair request $\mathcal{R} = \{V(n)\}$. Intuitively, the reason is that the qABox together with the TBox implies the existence of arbitrarily long loves-chains starting from n , which are no longer entailed by the TBox if $V(n)$ is removed (see Example 9 in [11] for a more detailed argument). One might think that the first GCI $V \sqsubseteq \exists \ell. V$ is enough to destroy existence of an optimal repair. This is, however, not the case. Without the second GCI one can introduce an anonymous vain individual x that is loved by n and loves itself to obtain an optimal repair.

In the *first part* of the paper (Section 2 and Section 3), we will describe the repair approaches developed in our previous work [7, 16], but with an emphasis on motivation rather than on technical details. The *second part* of the paper (Section 4 and Section 5) describes new result. We will consider more *concise representations* of optimal repairs, to deal both with the exponential size of canonical repairs in the IQ case and the non-existence problem w.r.t. cyclic TBoxes in the CQ case.

The former problem is due to the fact that the canonical repairs employed in our approach are by construction of exponential size. To alleviate this problem, we have, on the one hand, developed in [7] an optimized algorithm for computing repairs, which yields *optimized repairs* that are equivalent to the canonical ones, but in most cases considerably smaller, though in the worst case they may still be exponential. On the other hand, each canonical repair is induced by a so-called *repair seed*, whose size is polynomial in the size of the TBox and the repair request. We have seen in [13] that, for the IQ case, one can compute consequences of canonical repairs and check IQ-entailment between them by working only with the seed functions inducing them. This way, the exponential blow-up due to the construction of the canonical repair can be avoided. In Section 4, we report on experimental results that compare the performance on answering instance queries between the optimized repairs and the canonical ones represented by seed functions.

In Section 5, we show that, also in the CQ case, optimal repairs always exist and cover all repairs if we allow for certain infinite, but finitely represented qABoxes. To be more precise, we introduce the notion of a *shell unfolding* of a given qABox, which basically unravels parts of the qABox into (possibly infinite) trees. The shell unfoldings of IQ-saturations turn out to be CQ-saturations, and this also works for cyclic TBoxes. If we then consider the canonical IQ-repairs for a given repair problem, then we can prove that their shell unfoldings yields a set of (possibly infinite) CQ-repairs that cover all CQ-repairs. In addition, consequences from such shell unfolded repairs and entailment between them can be decided based on their finite representation without an increase in complexity. Thus, one can work with them as if they were finite.

2 Preliminaries

We recall the definition of the DL \mathcal{EL} and then introduce quantified ABoxes as well as the two entailment relations we employ for them.

The Description Logic \mathcal{EL} As usual in DL, knowledge about an application domain is represented in \mathcal{EL} using classes (called concepts), relationships (called roles), and objects (called individuals), which are collected in the signature Σ , consisting of pairwise disjoint sets of *concept names* Σ_C , *role names* Σ_R , and *individual names* Σ_I . *Concept descriptions* C of \mathcal{EL} are then constructed using the grammar rule $C ::= \top \mid A \mid C \sqcap C \mid \exists r.C$, where A ranges over concept names and r over role names. An *atom* is a concept name A or an *existential restriction* $\exists r.C$. Each concept description C is a conjunction of atoms, with \top corresponding to the empty conjunction. We denote the set of these atoms as $\text{Conj}(C)$.

An \mathcal{EL} TBox can be used to state subconcept-superconcept relationships between such concept descriptions, i.e., it is a finite set of *concept inclusions* (CIs) $C \sqsubseteq D$, where C, D are \mathcal{EL} concept descriptions. In the ABox one can then relate individuals with concepts and with other individuals, i.e., it is a finite set of *concept assertions* $C(a)$ and *role assertions* $r(a, b)$, where a, b are individual names, r is a role name, and C is an \mathcal{EL} concept description. An \mathcal{EL} ontology is a pair consisting of an \mathcal{EL} ABox and an \mathcal{EL} TBox.

The semantics of \mathcal{EL} is defined as usual [5] based on the notion of an *interpretation* $\mathcal{I} = (\text{Dom}(\mathcal{I}), \cdot^{\mathcal{I}})$, which assigns subsets $A^{\mathcal{I}}$ of the non-empty set $\text{Dom}(\mathcal{I})$ to concept names A , binary relations $r^{\mathcal{I}}$ on $\text{Dom}(\mathcal{I})$ to role names r , and elements $a^{\mathcal{I}}$ of $\text{Dom}(\mathcal{I})$ to individual names a . This mapping is extended to concept descriptions according to the semantics of the constructors. The interpretation \mathcal{I} is a model of the TBox \mathcal{T} if it satisfies all its CIs, i.e., $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ holds for all CIs $C \sqsubseteq D$ in \mathcal{T} . Similarly, \mathcal{I} is a model of the ABox \mathcal{A} if it satisfies its assertions, i.e., $a^{\mathcal{I}} \in C^{\mathcal{I}}$ and $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}$ holds for all concept assertions $C(a)$ and role assertion $r(a, b)$ in \mathcal{A} . It is a model of the ontology $(\mathcal{T}, \mathcal{A})$ if it is a model of both \mathcal{T} and \mathcal{A} .

Reasoning makes implicit consequences of an ontology explicit. For instance, we say that a concept assertion $C(a)$ is *entailed* by an ABox \mathcal{A} w.r.t. a TBox \mathcal{T} if $C(a)$ is satisfied in all models of \mathcal{A} and \mathcal{T} ; this is abbreviated as $\mathcal{A} \models^{\mathcal{T}} C(a)$ and we also say that a is an *instance* of C w.r.t. \mathcal{A} and \mathcal{T} . Similarly, a CI $C \sqsubseteq D$ is *entailed* by \mathcal{T} if $C \sqsubseteq D$ is satisfied in every model of \mathcal{T} ; we then write $C \sqsubseteq^{\mathcal{T}} D$ and also say that C is *subsumed* by D w.r.t. \mathcal{T} . In case $\mathcal{T} = \emptyset$, we may omit the superscript \emptyset and just write \models instead of \models^{\emptyset} . Both the instance and the subsumption problem are decidable in polynomial time in \mathcal{EL} [3].

Quantified ABoxes Quantified ABoxes were first introduced in [16], but they were also considered, as relational datasets with labelled nulls, in [20], and their existentially quantified variables correspond to the “anonymous individuals” in the OWL 2 standard [31]. Also, as explained in [16], quantified ABoxes are basically the same as Boolean conjunctive queries. Informally, a quantified ABox is an \mathcal{EL} ABox where concept assertions are restricted to concept names and in addition to individuals one can use variables in assertions. To indicate that the names of these variables are irrelevant, we quantify them existentially.

More formally, a *quantified ABox* (*qABox*) $\exists X.\mathcal{A}$ consists of a finite set X of *variables*, which is disjoint with the signature Σ , and of a *matrix* \mathcal{A} , which is a finite set of assertions $A(u)$ and $r(u, v)$, where A is a concept name, r a role name, and u, v individual names or variables. We call the individual names and variables occurring in $\exists X.\mathcal{A}$ its *objects*, and denote the set of them by $\text{Obj}(\exists X.\mathcal{A})$. Regarding the semantics of a qABox $\exists X.\mathcal{A}$, we can translate it in an obvious way into a first-order formula by taking the conjunction of the assertions in \mathcal{A} (viewed as atomic formulas) and prefacing it with an existential quantifier prefix containing exactly the variables in X . The models of $\exists X.\mathcal{A}$ are then the first-order models of this formula.

Based on this semantics, we can now define when a qABox entails another qABox or a concept assertion in the usual way. If α is an \mathcal{EL} concept assertion or a qABox, then $\exists X.\mathcal{A}$ entails α w.r.t. the \mathcal{EL} TBox \mathcal{T} (written $\exists X.\mathcal{A} \models^{\mathcal{T}} \alpha$) if every model of $\exists X.\mathcal{A}$ and \mathcal{T} is a model of α . Again, we may omit the superscript \emptyset if \mathcal{T} is empty. If α is a concept assertion, then entailment $\models^{\mathcal{T}}$ can be decided in polynomial time whereas it is NP-complete if α is a qABox [7, 16]. NP-hardness already holds without a TBox.

From a syntactic point of view, \mathcal{EL} ABoxes that use compound concept descriptions in concept assertions are not qABoxes, but it is easy to see that every \mathcal{EL} ABox can be transformed into an equivalent qABox (i.e., one having the same models) [16]. Conversely, not every qABox has an equivalent \mathcal{EL} ABox, the simplest example being $\exists \{y\}.\{r(y, y)\}$, which enforces an r -loop in every model, but without naming the element that has this loop. In contrast, \mathcal{EL} ABoxes can only enforce loops for named individuals, i.e., elements of Σ_1 . Also note that a qABox cannot entail $C(x)$ for a variable x since this is not a well-formed concept assertion. We can, however, view the matrix \mathcal{A} as a normal ABox (where the variables are treated as individuals), and then one can derive concept assertions for elements of X from \mathcal{A} . The following lemma, which gives a recursive charac-

terization of the instance relationship for the case of an empty TBox is relevant for our construction of canonical repairs.

Lemma 1 ([16]). *Let $\exists X.\mathcal{A}$ be a qABox, D an \mathcal{EL} concept description, and $u \in \text{Obj}(\exists X.\mathcal{A})$. Then $\mathcal{A} \models D(u)$ iff the following statements are satisfied for every $C \in \text{Conj}(D)$:*

1. *if $C = A$ is a concept name, then \mathcal{A} contains $A(u)$,*
2. *if $C = \exists r.E$ is an existential restriction, then \mathcal{A} contains a role assertion $r(u, v)$ such that $\mathcal{A} \models E(v)$.*

Two entailment relations between qABoxes As motivated in the introduction, it makes sense to compare qABoxes w.r.t. the queries they entail rather than w.r.t. the models they have. Instance queries (IQ) are just concept assertions whereas (Boolean) conjunctive queries (CQ) are just qABoxes. The qABox $\exists X.\mathcal{A}$ IQ-entails the qABox $\exists Y.\mathcal{B}$ w.r.t. \mathcal{T} (written $\exists X.\mathcal{A} \models_{\text{IQ}}^{\mathcal{T}} \exists Y.\mathcal{B}$) if $\exists Y.\mathcal{B} \models^{\mathcal{T}} C(a)$ implies $\exists X.\mathcal{A} \models^{\mathcal{T}} C(a)$ for every \mathcal{EL} concept assertion $C(a)$. The definition of CQ-entailment considers all qABoxes $\exists Z.\mathcal{C}$ in place of concept assertions $C(a)$. It is easy to see that the CQ-entailment relation $\models_{\text{CQ}}^{\mathcal{T}}$ actually coincides with the model-based entailment relation $\models^{\mathcal{T}}$ introduced above [7, 16]. Since every concept assertion can be translated into an equivalent qABox, CQ-entailment is a stronger requirement than IQ-entailment.

With respect to the empty TBox, these query-based entailment relations have structural characterizations by means of simulations and homomorphisms [16]. In the IQ case, $\exists X.\mathcal{A} \models_{\text{IQ}} \exists Y.\mathcal{B}$ iff there is a *simulation* from $\exists Y.\mathcal{B}$ to $\exists X.\mathcal{A}$, which is a relation $\mathfrak{S} \subseteq \text{Obj}(\exists Y.\mathcal{B}) \times \text{Obj}(\exists X.\mathcal{A})$ satisfying the following:

- (S1) If a is an individual name, then $(a, a) \in \mathfrak{S}$.
- (S2) If $(u, u') \in \mathfrak{S}$ and $A(u) \in \mathcal{B}$, then $A(u') \in \mathcal{A}$.
- (S3) If $(u, u') \in \mathfrak{S}$ and $r(u, v) \in \mathcal{B}$, then $(v, v') \in \mathfrak{S}$ and $r(u', v') \in \mathcal{A}$ for some v' .

A *homomorphism* from $\exists Y.\mathcal{B}$ to $\exists X.\mathcal{A}$ is a function $h: \text{Obj}(\exists Y.\mathcal{B}) \rightarrow \text{Obj}(\exists X.\mathcal{A})$ for which the relation $\{(u, h(u)) \mid u \in \text{Obj}(\exists Y.\mathcal{B})\}$ is a simulation. In the CQ case, entailment is characterized as follows: $\exists X.\mathcal{A} \models_{\text{CQ}} \exists Y.\mathcal{B}$ iff there is a homomorphism from $\exists Y.\mathcal{B}$ to $\exists X.\mathcal{A}$.

To extend these characterizations of the entailment relations to the case of non-empty TBoxes, we must first saturate the qABox on the left-hand side. We defer describing saturation to the second part of the next section, where we extend our repair approach from the setting without TBox to the one with a TBox.

3 Canonical and Optimal Repairs

We start with introducing (optimal) repairs in the general setting, but then concentrate first on the CQ case without a TBox for didactic reasons, before considering the IQ case and explaining how non-empty TBoxes can be tackled.

As unwanted consequences we consider \mathcal{EL} concept assertions. Whereas it would be useful to be able to specify unwanted consequences via CQs, this may cause non-existence of optimal repairs unless one considers a strongly restricted class of CQs [11]. For this reason, a *repair request* will in the following be a finite set of concept assertions, both in the IQ and in the CQ case.

Definition 2. Let \mathcal{T} be an \mathcal{EL} TBox, $\exists X.A$ a qABox, \mathcal{R} a repair request, and $QL \in \{IQ, CQ\}$.

- The qABox $\exists Y.B$ is a QL-repair of $\exists X.A$ for \mathcal{R} w.r.t. \mathcal{T} if $\exists X.A \models_{QL}^{\mathcal{T}} \exists Y.B$ and $\exists Y.B \not\models^{\mathcal{T}} C(a)$ for each $C(a) \in \mathcal{R}$.
- Such a repair $\exists Y.B$ is optimal if there is no QL-repair $\exists Z.C$ such that $\exists Z.C \models_{QL}^{\mathcal{T}} \exists Y.B$, but $\exists Y.B \not\models_{QL}^{\mathcal{T}} \exists Z.C$.
- We say that a set \mathfrak{R} of QL-repairs of $\exists X.A$ for \mathcal{R} w.r.t. \mathcal{T} covers all QL-repairs if every QL-repair of $\exists X.A$ for \mathcal{R} w.r.t. \mathcal{T} is QL-entailed by an element of \mathfrak{R} .

Since CQ-entailment implies IQ-entailment, every CQ-repair is also an IQ-repair, but the converse need not hold. The latter can be illustrated by the second version of our Narcissus example from the introduction. Consider the TBox $\mathcal{T} = \{V \sqsubseteq \exists \ell.V, \exists \ell.V \sqsubseteq V\}$, the qABox $\exists \emptyset.\{V(n)\}$ and the repair request $\mathcal{R} = \{V(n)\}$. Then $\exists \{x\}.\{\ell(n, x), \ell(x, x)\}$ is an IQ-repair, but not a CQ-repair. In fact, this qABox is not CQ-entailed w.r.t. \mathcal{T} by $\exists \emptyset.\{V(n)\}$ since there are models of $\exists \emptyset.\{V(n)\}$ and \mathcal{T} that do not contain an individual with a loop. It is IQ-entailed, basically since all \mathcal{EL} concept assertions of the form $(\exists \ell.)^k \top(n)$ are entailed by $\exists \emptyset.\{V(n)\}$ w.r.t. \mathcal{T} .

The question is now how one can actually compute all optimal repairs of a given repair problem, consisting of an \mathcal{EL} TBox, a qABox, and a query language $QL \in \{IQ, CQ\}$. We start with the case where the TBox is empty and $QL = CQ$.

Blind search A first idea could be to start with the input qABox and then generate a chain of qABoxes with entailment relationships between them, until a qABox that does not entail any element of \mathcal{R} has been found. Such a chain can be generated by applying the following rules successively to the current qABox $\exists X.A$:

Copy Rule. Choose an object u of $\exists X.A$ as well as a fresh variable $y \notin \text{Obj}(\exists X.A)$, and return the qABox $\exists (X \cup \{y\}). (\mathcal{A} \cup \{A(y) \mid A(u) \in \mathcal{A}\} \cup \{r(t, y) \mid r(t, u) \in \mathcal{A}\} \cup \{r(y, y) \mid r(u, u) \in \mathcal{A}\} \cup \{r(y, v) \mid r(u, v) \in \mathcal{A}\})$.

Delete Rule. Choose an assertion α in \mathcal{A} and return the qABox $\exists X. (\mathcal{A} \setminus \{\alpha\})$, or choose a variable $x \in X$ that does not occur in \mathcal{A} and return the qABox $\exists (X \setminus \{x\}). \mathcal{A}$.

It is easy to see that the qABox obtained from $\exists X.A$ by application of ones of these rules is CQ-entailed by $\exists X.A$. The following proposition shows that these rules indeed cover the whole search space of entailed qABoxes.

Proposition 3. *If $\exists X.\mathcal{A} \models_{\text{CQ}} \exists Y.\mathcal{B}$, then there is a finite chain of applications of the Copy and Delete Rules that starts with $\exists X.\mathcal{A}$ and ends with $\exists Y.\mathcal{B}$.*

Proof sketch. If $\exists X.\mathcal{A} \models_{\text{CQ}} \exists Y.\mathcal{B}$, then there is a homomorphism from $\exists Y.\mathcal{B}$ to $\exists X.\mathcal{A}$. If this homomorphism is not injective, then we can make it injective by adding copies of individuals that are images of several elements of $\text{Obj}(\exists Y.\mathcal{B})$ to $\exists X.\mathcal{A}$. After that, we can remove assertions that are in the image, but not in the pre-image. Finally, we can rename variables and remove variables that do not have a pre-image (see [6] for a more detailed proof). \square

If one starts with the input $\text{qABox } \exists X.\mathcal{A}$ and generates a search tree by applying the above rules, this process need not terminate since one can generate an arbitrary number of copies of objects. But now Proposition 11 in [11] comes to the rescue: if $\exists X.\mathcal{A}$ contains m objects and \mathcal{R} contains n atoms, then any repair of $\exists X.\mathcal{A}$ for \mathcal{R} is CQ-entailed by a repair that has at most $m \cdot 2^n$ objects. Thus, we can restrict the search to qABoxes that have at most this many objects, which makes the search tree finite. We can be sure that the repairs found this way cover all repairs. The optimal repair can be obtained from this covering set by removing non-optimal elements, i.e., elements that are strictly entailed by another element.

Canonical repairs Obviously, the blind search approach for computing optimal repairs sketched above is very inefficient. However, it provides us with several interesting ideas for how to construct, in a more direct way, a set of repairs that covers all repairs. First, we notice that we must generate copies of objects, and then may need to remove assertions for these copies. Second, the cited result from [11] tells us that creating at most exponentially many copies of each object is sufficient.

In our canonical repairs, each object u of the input $\text{qABox } \exists X.\mathcal{A}$ receives copies of the form $\langle\langle u, \mathcal{K} \rangle\rangle$, where the second component specifies which assertions $C(u)$ that are entailed by \mathcal{A} must *not* hold for this copy. More formally, \mathcal{K} is a *repair type* for u , i.e., a subset of the set of atoms occurring in \mathcal{R} that satisfies the following two properties:

- (RT1) $\mathcal{A} \models C(u)$ for each atom $C \in \mathcal{K}$,
- (RT2) $C \not\sqsubseteq^\emptyset D$ for each pair of distinct atoms C, D in \mathcal{K} .

The first condition is due to the fact that we only need to remove instance relationships that hold in \mathcal{A} . The second reduces the number of different repair types. It is justified by the fact that requiring to remove $D(u)$ ensures that also $C(u)$ must be removed if $C \sqsubseteq^\emptyset D$.

The canonical repairs have the same set of objects and the same matrix. They have all tuples $\langle\langle u, \mathcal{K} \rangle\rangle$ as their objects, where $u \in \text{Obj}(\exists X.\mathcal{A})$ and \mathcal{K} is a repair type for u . Using these objects, the matrix \mathcal{B} of the canonical repairs consists of the following assertions:

- (CR1) $A(\langle\langle u, \mathcal{K} \rangle\rangle) \in \mathcal{B}$ if $A(u) \in \mathcal{A}$ and $A \notin \mathcal{K}$,

(CR2) $r(\langle\langle u, \mathcal{K} \rangle\rangle, \langle\langle v, \mathcal{L} \rangle\rangle) \in \mathcal{B}$ if $r(u, v) \in \mathcal{A}$ and, for each $\exists r. C \in \mathcal{K}$ with $\mathcal{A} \models C(v)$, there is an atom $D \in \mathcal{L}$ such that $C \sqsubseteq^\emptyset D$.

To understand this definition, one needs to consider Lemma 1. Regarding concept names $A \in \mathcal{K}$, not adding the concept assertion $A(\langle\langle u, \mathcal{K} \rangle\rangle)$ to \mathcal{B} ensures that this assertion is not entailed by \mathcal{B} . For existential restrictions $\exists r. C \in \mathcal{K}$, we can only have the role assertion $r(\langle\langle u, \mathcal{K} \rangle\rangle, \langle\langle v, \mathcal{L} \rangle\rangle)$ in \mathcal{B} if \mathcal{B} does not entail $C(\langle\langle v, \mathcal{L} \rangle\rangle)$. This non-entailment is ensured by having an atom $D \in \mathcal{L}$ that satisfies $C \sqsubseteq^\emptyset D$. In fact, $\mathcal{B} \models C(\langle\langle v, \mathcal{L} \rangle\rangle)$ would otherwise imply $\mathcal{B} \models D(\langle\langle v, \mathcal{L} \rangle\rangle)$, which is forbidden due to $D \in \mathcal{L}$.

To determine a concrete canonical repair, we choose, for each individual a of $\exists X. \mathcal{A}$, one of its copies as representative of a in \mathcal{B} . Of course, this choice must be made such that the obtained qABox really is a repair, i.e., does not entail any of the unwanted consequences in \mathcal{R} . Formally, this is realized by fixing a *repair seed* \mathcal{S} , which maps each individual name a to a repair type \mathcal{S}_a for a such that the following condition is satisfied:

(RS) If $C(a) \in \mathcal{R}$ and $\mathcal{A} \models C(a)$, then there is an atom D in \mathcal{S}_a s.t. $C \sqsubseteq^\emptyset D$.

Given such a repair seed \mathcal{S} , the *canonical repair* $\text{rep}(\exists X. \mathcal{A}, \mathcal{S})$ induced by \mathcal{S} is the qABox $\exists Y. \mathcal{B}$, where individual names a and their copies $\langle a, \mathcal{S}_a \rangle$ are used as synonyms, and Y consists of the other objects of \mathcal{B} . This construction works both in the CQ and in the IQ case, and yields a set of repairs that covers all repairs.

Proposition 4 ([16]). *Consider a qABox $\exists X. \mathcal{A}$, an \mathcal{EL} repair request \mathcal{R} , and a query language $\text{QL} \in \{\text{IQ}, \text{CQ}\}$. For each repair seed \mathcal{S} , the induced canonical repair $\text{rep}(\exists X. \mathcal{A}, \mathcal{S})$ is a QL-repair of $\exists X. \mathcal{A}$ for \mathcal{R} . Conversely, if $\exists Z. C$ is a QL-repair of $\exists X. \mathcal{A}$ for \mathcal{R} , then there is a repair seed \mathcal{S} such that $\text{rep}(\exists X. \mathcal{A}, \mathcal{S}) \models_{\text{QL}} \exists Z. C$.*

The set of all canonical repairs can obviously be computed in exponential time. To obtain the optimal repairs, one needs to employ entailment tests to remove the non-optimal ones from it. Since IQ-entailment is in P and CQ-entailment is NP-complete, this yields the complexity results stated in the following theorem. Obviously, after removing redundant elements, the obtained set still covers all repairs.

Theorem 5 ([16]). *The set of optimal QL-repairs of $\exists X. \mathcal{A}$ for \mathcal{R} covers all QL-repairs. There is a (deterministic) algorithm that computes this set and runs in exponential time. If $\text{QL} = \text{CQ}$, then this algorithm needs access to an NP oracle, whereas no such oracle is required for $\text{QL} = \text{IQ}$.*

Let us come back to the first variant of the Narcissus example from the introduction, where the input qABox is $\exists \emptyset. \mathcal{A}$ for $\mathcal{A} = \{V(n), \ell(n, n)\}$ and the repair request is $\mathcal{R} = \{V(n)\}$. The only atom in \mathcal{R} is V , and both \emptyset and $\{V\}$ is a repair type for n . The only repair seed is \mathcal{S} with $\mathcal{S}_n = \{V\}$. If we denote $\langle n, \mathcal{S}_n \rangle$ with n and $\langle n, \emptyset \rangle$ with x , then the qABox $\exists \{x\}. \{\ell(n, n), \ell(n, x), \ell(x, n), \ell(x, x), V(x)\}$ is the only canonical repair, which thus is an optimal repair both in the CQ and in the IQ case.

Adding a static TBox As mentioned before, we restrict the attention to the case where the TBox is assumed to be correct, and thus is static in the sense that it must not be changed in the repair process. Our main idea for dealing with an \mathcal{EL} TBox \mathcal{T} is to extend the given qABox $\exists X.\mathcal{A}$ with consequences entailed by the CIs in \mathcal{T} . We call this extension process *saturation* [7].

Intuitively, if $C \sqsubseteq D \in \mathcal{T}$, then saturation adds the assertion $D(u)$ to the matrix \mathcal{A} if $\mathcal{A} \models C(u)$, but $\mathcal{A} \not\models D(u)$. However, if D is a compound concept description, then this does not generate a well-formed new qABox. For this reason, one must express $D(u)$ by atomic assertions. Obviously, for each concept name $A \in \text{Conj}(D)$, we must add the assertion $A(u)$ to \mathcal{A} . For each existential restriction $\exists r.E \in \text{Conj}(D)$, we add a new variable x to X and the assertions $r(u, x)$ and $E(x)$ to \mathcal{A} . In case E is still compound, we apply the process of expressing such an assertion by atomic ones recursively. To be more precise, the treatment of existential restrictions differs depending on whether we are in the CQ or the IQ case. In the former, we always need to use a new variable x . In the IQ case, for each concept description E occurring in an existential restriction $\exists r.E$ in \mathcal{T} , we introduce the variable x_E , and reuse this variable whenever we encounter an existential restriction with E in the second position. Let us call this process of expressing a concept assertion $D(u)$ for a compound concept description D the *QL-unfolding of $D(u)$* , for $\text{QL} \in \{\text{IQ}, \text{CQ}\}$. *QL-saturation* is the process of applying the following saturation rule exhaustively:

QL-Saturation Rule. Choose an object u of $\exists X.\mathcal{A}$ as well as a CI $C \sqsubseteq D$ in \mathcal{T} with $\mathcal{A} \models C(u)$, but $\mathcal{A} \not\models D(u)$, and add $D(u)$ to \mathcal{A} . Then apply QL-unfolding to $D(u)$.

Example 6. Consider again the TBox $\mathcal{T} = \{V \sqsubseteq \exists \ell.V, \exists \ell.V \sqsubseteq V\}$ and the qABox $\exists \emptyset.\{V(n)\}$. The first application of the IQ-saturation rule to n adds the assertion $(\exists \ell.V)(n)$ to the qABox. The IQ-unfolding of this assertion introduces one new variable x_V , adds the assertions $\ell(n, x_V)$ and $V(x_V)$, and removes the compound assertion. The IQ-saturation rule now applies to x_V , adding $(\exists \ell.V)(x_V)$. The IQ-unfolding of this assertion reuses the variable x_V , and adds the assertion $\ell(x_V, x_V)$. This completes the IQ-saturation process with the IQ-saturated qABox $\exists \{x_V\}.\{V(n), \ell(n, x_V), V(x_V), \ell(x_V, x_V)\}$.

This qABox is IQ-entailed by $\exists \emptyset.\{V(n)\}$ w.r.t. \mathcal{T} , but it is not CQ-entailed. The reason for the latter non-entailment is that there are models of $\exists \emptyset.\{V(n)\}$ and \mathcal{T} where no element has a loop. To avoid introducing a loop or a cycle, we must use a new variable in each CQ-unfolding of an assertion of the form $(\exists \ell.V)(x)$. But this clearly leads to non-termination of the CQ-saturation process. To ensure termination for the CQ case, we restrict the attention in [7] to cycle-restricted TBoxes, where an \mathcal{EL} TBox \mathcal{T} is *cycle-restricted* if there are no role names r_1, \dots, r_n and no \mathcal{EL} concept description C such that $C \sqsubseteq^{\mathcal{T}} \exists r_1 \dots \exists r_n.C$.

Proposition 7 ([7]). Let $\text{QL} \in \{\text{IQ}, \text{CQ}\}$, $\exists X.\mathcal{A}$ a qABox, and \mathcal{T} an \mathcal{EL} TBox, which is cycle-restricted if $\text{QL} = \text{CQ}$. Then QL-saturation always terminates

with a qABox $\text{sat}_{\text{QL}}^{\mathcal{T}}(\exists X.A)$ that satisfies $\exists X.A \models_{\text{QL}}^{\mathcal{T}} \exists Y.B$ iff $\text{sat}_{\text{QL}}^{\mathcal{T}}(\exists X.A) \models_{\text{QL}} \exists Y.B$ for all qABoxes $\exists Y.B$. The IQ-saturation $\text{sat}_{\text{IQ}}^{\mathcal{T}}(\exists X.A)$ can be computed in polynomial time, whereas the computation of $\text{sat}_{\text{CQ}}^{\mathcal{T}}(\exists X.A)$ may require exponential time in the worst case.

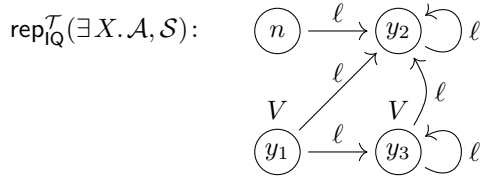
The idea is now to apply the repair process described above to the saturated qABox rather than the original one. This ensures that, in RT1, the entailment is then w.r.t. the TBox. However, without additional changes to our construction of canonical repairs, the obtained qABox would not be a repair. In our example, a canonical repair of $\exists\{x_V\}.\{V(n), \ell(n, x_V), V(x_V), \ell(x_V, x_V)\}$ for $\mathcal{R} = \{V(n)\}$ could choose as synonym for n the copy $\langle n, \{V\} \rangle$ that does not belong to V , but still has an ℓ -successor that belongs to V . Together with the CI $\exists \ell.V \sqsubseteq V$, this qABox would then still entail $V(n)$.

To avoid this problem, we amend the definition of repair types as follows. First, we now consider subsets of the atoms occurring in \mathcal{R} or \mathcal{T} as possible repair types. Second, we add an additional condition to the definition:

(RT3) If C is an atom in \mathcal{K} and $E \sqsubseteq F$ is a CI in \mathcal{T} with $\mathcal{A} \models E(u)$ and $F \sqsubseteq^{\emptyset} C$, then there is an atom D in \mathcal{K} such that $E \sqsubseteq^{\emptyset} D$.⁶

In our example, $\mathcal{K} = \{V\}$ does not satisfy RT3 since the saturated qABox entails $\exists \ell.V(n)$, there is a CI that has $\exists \ell.V$ as left-hand side and $V \in \mathcal{K}$ as right-hand side, but \mathcal{K} does not contain an atom that subsumes $\exists \ell.V(n)$. In fact, with the additional condition RT3, any repair type for n that contains V must also contain $\exists \ell.V$. The copy $\langle n, \{V, \exists \ell.V\} \rangle$ of n does not belong to V in the canonical repair, and also does not have an ℓ -successor that belongs to V .

Overall, for $\mathcal{T} = \{V \sqsubseteq \exists \ell.V, \exists \ell.V \sqsubseteq V\}$, the qABox $\exists X.A = \exists \emptyset.\{V(n)\}$, and the repair request $\mathcal{R} = \{V(n)\}$, we obtain the following canonical IQ-repair induced by the (unique) repair seed \mathcal{S} with $\mathcal{S}_n = \{V, \exists \ell.V\}$:



where y_1 stands for $\langle n, \emptyset \rangle$, y_2 for $\langle x_N, \{N, \exists \ell.N\} \rangle$, and y_3 for $\langle x_N, \emptyset \rangle$.

In general, let $\text{rep}_{\text{QL}}^{\mathcal{T}}(\exists X.A, \mathcal{S})$ be the canonical repairs obtained by first QL-saturating $\exists X.A$ w.r.t. \mathcal{T} and then applying the amended repair approach that takes RT3 into account. Then Proposition 4 and Theorem 5 hold accordingly in the presence of a static TBox \mathcal{T} if we replace $\text{rep}(\exists X.A, \mathcal{S})$ with $\text{rep}_{\text{QL}}^{\mathcal{T}}(\exists X.A, \mathcal{S})$ and in the CQ case add the assumption that \mathcal{T} is cycle-restricted (see [7]).

⁶ This condition differs from the one given in [7]. However, this third condition is only employed in Lemma XIII in [8] to show that the canonical repairs are saturated, for which the simpler condition given here suffices.

4 Concise Representations of Canonical IQ-Repairs

Canonical IQ-repairs are of exponential size, not only in the worst case, but also in the best case. In this section, we consider two approaches for alleviating this problem. One approach produces considerably smaller repairs in practice, which may, however, still be exponential in the worst case. The second approach uses the polynomial-sized repair seeds as representations for the exponentially large canonical repairs.

Optimized IQ-repairs To avoid generating exponential-sized repairs also in the best case, we have developed in [7] an optimized algorithm for computing repairs induced by repair seeds. Intuitively, these *optimized repairs* do not contain all the objects occurring in the canonical repair, but only those that are really needed. We have shown that the optimized IQ-repair induced by a repair seed \mathcal{S} is IQ-equivalent to the canonical one induced by \mathcal{S} , and thus the set of optimized IQ-repairs can be used in place of the set of canonical ones when computing the optimal repairs. The experiments described in [7] show that the optimized repairs are in most cases considerably smaller than the canonical ones. For example, in the canonical IQ-repair we have just computed for our Narcissus example, the objects y_1 and y_3 are not needed since they are not reachable from n . IQ-equivalence of the optimized repair $\exists\{y_2\}.\{\ell(n, y_2), \ell(y_2, y_2)\}$ with the canonical one can be seen by using the identity on the objects n and y_2 as simulation in both directions.

Note, however, that in general an exponential blow-up cannot be avoided, as already shown in [12] for a restricted class of qABoxes without a TBox. This blow-up is not only a problem when computing the repair, but also when using it later on to answer queries. While answering IQs is polynomial for the original (unrepaired) qABox, it may become exponential after the repair if we measure the complexity in the size the repair problem, consisting of the original qABox, the TBox, and the repair request.

Representing canonical IQ-repairs by repair seeds The size of a repair seed \mathcal{S} is polynomial in the size of the repair problem, and it uniquely determines the induced canonical repair $\text{rep}_{\text{IQ}}^{\mathcal{T}}(\exists X.\mathcal{A}, \mathcal{S})$. To take advantage of this more concise representation of canonical repairs, we must be able to work directly with this representation when comparing the repairs w.r.t. IQ-entailment and when answering IQs w.r.t. them. The following proposition shows how this can be realized.

Proposition 8 ([10,13]). *Let \mathcal{T} be an \mathcal{EL} TBox, $\exists X.\mathcal{A}$ a qABox, \mathcal{R} a repair request, $\mathcal{S}, \mathcal{S}'$ repair seeds, and $E(b)$ an \mathcal{EL} concept assertion. Then,*

1. $\text{rep}_{\text{IQ}}^{\mathcal{T}}(\exists X.\mathcal{A}, \mathcal{S}) \models_{\text{IQ}}^{\mathcal{T}} \text{rep}_{\text{IQ}}^{\mathcal{T}}(\exists X.\mathcal{A}, \mathcal{S}')$ iff for each individual name a and for each atom $C \in \mathcal{S}_a$, there is an atom $D \in \mathcal{S}'_a$ with $C \sqsubseteq^{\emptyset} D$.
2. $\text{rep}_{\text{IQ}}^{\mathcal{T}}(\exists X.\mathcal{A}, \mathcal{S}) \models^{\mathcal{T}} E(b)$ iff $\exists X.\mathcal{A} \models^{\mathcal{T}} E(b)$ and \mathcal{S}_b does not contain any atom D with $E \sqsubseteq^{\mathcal{T}} D$.

The conditions formulated in this proposition are clearly decidable in time polynomial in the size of the repair problem. Thus, from a theoretical point of view, representing canonical repairs using repair seeds is preferable to using optimized repairs since the worst-case complexity of the relevant inference problems is polynomial for the former, whereas it is exponential for the latter. Comparing the worst-case complexity of two algorithms does not always tell us which algorithm will perform better in practice. To investigate the advantages and disadvantages of our two concise representations of canonical IQ-repairs in practice, we performed experiments on real-world ontologies.

Experimental evaluation The goal of the experiments was to evaluate the performance of the two representations with respect to the time needed for answering instance queries. To this end, we created a benchmark consisting of \mathcal{EL} ontologies, instance queries, and repair requests. As in the experiments in [7], which mainly compared the sizes of the optimized repairs with that of the canonical ones, we took the ontologies from the OWL EL Materialization track of the OWL Ontology Reasoner Evaluation 2015 [30], filtering out axioms that cannot be expressed in \mathcal{EL} . To test the limits of both approaches, we this time included all 109 ontologies from this corpus, instead of considering only ontologies of up to 100,000 axioms as in [7]. Table 1 provides information on how large the employed ontologies were.

For each ontology, we randomly generated 100 IQs. To generate repair requests, we used the approach employed in [7], which generates requests where the concept assertions involve only concept names. In addition, we this time also generated repair requests containing assertions with compound concept descriptions. The repair requests generated in these two ways are respectively denoted RR1 and RR2 in the following. We attempted to compute 10 repair seeds per ontology based on the generated repair requests, which was, however, not always possible within a timeout of 10 minutes. For each tuple of ontology, repair request, and repair seed, we first computed the induced optimized IQ-repair, which was possible in most, but not all, cases within a timeout of 1 hour. Then we compared the performance of answering IQs from the optimized repairs and from the repair seeds. Any required \mathcal{EL} reasoning was performed using ELK [24]. More information on the experimental setup can be found in [6].

Figure 1 shows the results of this comparison, where each point corresponds to a tuple of ontology, repair request, and seed function, the x-axis to the runtime of evaluating all 100 IQs using the repair seed, and the y-axis of evaluating all IQs using the optimized repair, where the red color denotes that we also count the computation time of the optimized repair, and the blue color denotes that we do not. For RR1 with the simple repair requests, using the repair seed instead of the precomputed repair was faster in 98.7% of cases if we also count the time for computing the repair, and otherwise in 17.9% of cases. As we can see however in Figure 1, using the optimized repair was almost never significantly faster, and there were many cases in which using the repair seed instead of the repair was significantly faster even if we do not count the time for computing the

Size Ontology				Size ABox				Size TBox			
min.	max.	med.	avg.	min.	max.	med.	avg.	min.	max.	med.	avg.
154	891,452	6,751	77,761.5	103	747,998	2,089	46,625.7	61	473,254	2,706	31,135.8

Table 1. Statistics of the used corpus of \mathcal{EL} ontologies after filtering out non- \mathcal{EL} axioms.

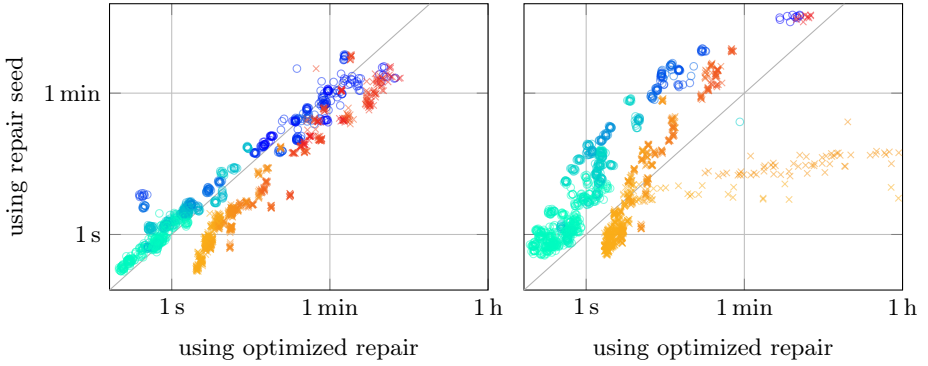


Fig. 1. Run times of evaluating 100 instance queries on repairs using the seed function (x-axis) vs. using the optimized repairs (y-axis). Color intensity corresponds to size of the input ontology. Orange-red crosses include times for computing the repair, whereas cyan-blue circles do not. Results of RR1 on the left, and for RR2 on the right.

repair. For RR2 with the complex repair requests, using the repair seed was faster in 64.6% of cases if we count the time for computing the repair, and otherwise almost never (0.13% of cases). The reason for this was that after obtaining the query answers from ELK, we still have to do a subsumption check for each individual in the answer when using the repair seed only (see the condition in Proposition 8). In RR2, each of these tests was more expensive, since we were comparing complex \mathcal{EL} concepts. When using the precomputed optimized repair, no additional subsumption tests are necessary.

The results show that computing the optimized repair explicitly rather than using the repair seed is only advisable if this repair is considered to be the final one, which is then used for many instance tests. This is not the case for intermediate repairs in a setting where the KE iteratively repairs the ontology by (a) choosing a repair seed, then (b) checking out the induced canonical repair by looking at some of its consequences, and based on this inspection deciding whether (c) to choose a different repair seed or (d) to use this repair seed, but maybe repair the obtained ontology further by formulating a new repair request. It then makes sense to compute the optimized repair only after the iterative repair process is finished.

If the repair is assumed to be the final one, a good indicator for when computing the optimized repair does not pay off is the size of the original ontology. If we consider RR1 and do not count the time for computing the repair, for on-

tologies with at most 404,509 axioms (85% of the corpus), using the repair seed was faster in only 6.8% of the cases, while for the larger ontologies, it was faster in 80.5% of the cases. The numbers are similar if we look at the size increase of the repair: if the repair contained at most 132,622 axioms more than the original ontology (85% of the corpus), then using the repair seed was faster in 5.5% of the cases, and otherwise in 87.5% of the cases.

5 Finite Representations of Optimal CQ-Repairs

The results concerning optimal CQ-repairs of [7] recalled in Section 3 assume that the TBox is cycle-restricted. We have seen an example (the version of our Narcissus example with a TBox) that for TBoxes not satisfying this restriction, optimal repairs need not exist. To overcome this problem, we allow for infinite qABoxes as repairs, but require that they have an appropriate finite representation. In our construction of optimal CQ-repairs, cycle-restrictedness of the TBox is needed to ensure that CQ-saturation terminates. For IQ-saturation, cycles in the TBox do not lead to non-termination since the saturation process can reuse variables. This is not possible for CQ-saturation since it may generate cycles in the saturated qABox that are not CQ-entailed by the original qABox. Whereas IQs cannot distinguish such cycles from their unfoldings, CQs obviously can. The idea is now to use appropriate unfoldings of IQ-saturations and canonical IQ-repairs in the CQ case.

Infinite qABoxes An *infinite qABox* is still of the form $\exists X.\mathcal{A}$, but now both the variable set X and the matrix \mathcal{A} may be infinite. The model-based semantics can straightforwardly be extended from finite qABoxes to infinite ones, and the correspondence between (model-based) entailment and the existence of a homomorphism is still easy to show. However, the equivalence between entailment and CQ-entailment no longer holds. While the existence of a homomorphism is still sufficient for CQ-entailment, it is no longer necessary, as illustrated by the following example.

Example 9. As left-hand side of the entailment, we consider the qABox representing the natural numbers with their usual order relation: $\exists X.\mathcal{A}$ with variables $X := \mathbb{N}$ and matrix $\mathcal{A} := \{r(m, n) \mid m < n\}$. As right-hand side, we take the real numbers: $\exists Y.\mathcal{B}$ with variables $Y := \mathbb{R}$ and matrix $\mathcal{B} := \{r(x, y) \mid x < y\}$. Each finite qABox entailed by $\exists Y.\mathcal{B}$ is also entailed by $\exists X.\mathcal{A}$, i.e., $\exists X.\mathcal{A} \models_{\text{CQ}} \exists Y.\mathcal{B}$. However, there is no homomorphism from $\exists Y.\mathcal{B}$ to $\exists X.\mathcal{A}$. In fact, no mapping from \mathbb{R} (the objects of $\exists Y.\mathcal{B}$) to \mathbb{N} (the objects of $\exists X.\mathcal{A}$) can be injective. Thus, if h was a homomorphism, then it would send two real numbers $x < y$ to the same natural number n , which would be a contradiction since \mathcal{B} contains the role assertion $r(x, y)$, whereas \mathcal{A} does not contain its image $r(n, n)$.

A slightly more complicated example can be used to show that this problem persists even if we consider only countable qABoxes [6]. The intuitive reason

for the difference between entailment and CQ-entailment is that CQs (which are finite) cannot capture differences of infinite qABoxes that manifest themselves only “in the infinite.” Fortunately, the problem goes away if we restrict the attention to shell unfoldings of finite ABoxes. Shell unfolding are similar to what is called unraveling in the DL literature [5], but it is applied to ABoxes rather than to interpretations.

Shell unfoldings and homomorphisms Consider a (finite) quantified ABox $\exists X.\mathcal{A}$, the objects of which are divided into *kernel objects* and *shell objects*, such that each individual name is a kernel object, each shell object is reachable from some kernel object, but no kernel object is reachable from any shell object. Later on, we will apply the shell unfolding operation to the IQ-saturation $\exists X.\mathcal{A}$ of a given finite qABox $\exists Y.\mathcal{B}$. In this setting, the kernel objects of $\exists X.\mathcal{A}$ are the objects of $\exists Y.\mathcal{B}$, and the shell objects are the additional objects introduced during the saturation process. It is easy to see that this division into kernel and shell objects satisfies the requirements we have just formulated.

A *shell path* is a sequence $u_0 \xrightarrow{r_1} u_1 \xrightarrow{r_2} \dots \xrightarrow{r_n} u_n$ that starts with a kernel object u_0 but otherwise only contains shell objects u_1, \dots, u_n such that \mathcal{A} contains $r_i(u_{i-1}, u_i)$ for all $i \in \{1, \dots, n\}$. We call $n \geq 0$ its *length*, u_0 its *source*, and u_n its *target*. Note that kernel objects, and thus also individuals, can be seen as shell paths of length 0. The target of such a shell path representing a kernel object is this object itself.

Definition 10. *The shell unfolding of $\exists X.\mathcal{A}$ is defined as the qABox $\exists X'.\mathcal{A}'$ with the following components:*

$$\begin{aligned} X' &:= \{ p \mid p \text{ is a shell path where } p \notin \Sigma_1 \}, \\ \mathcal{A}' &:= \{ A(p) \mid p \text{ is a shell path with target } u \text{ and } A(u) \in \mathcal{A} \} \cup \\ &\quad \{ r(u, v) \mid u, v \text{ are kernel objects and } r(u, v) \in \mathcal{A} \} \cup \\ &\quad \{ r(p, q) \mid p, q \text{ are shell paths such that } q = p \xrightarrow{r} u \text{ for a shell object } u \}. \end{aligned}$$

Note that a finite qABox can be seen as the shell unfolding of itself where all objects are assumed to be kernel objects. If the matrix \mathcal{A} contains cycles among shell objects, then the shell unfolding $\exists X'.\mathcal{A}'$ of $\exists X.\mathcal{A}$ is infinite. However, since $\exists X'.\mathcal{A}'$ is uniquely determined by the finite qABox $\exists X.\mathcal{A}$ and the division of its objects into kernel and shell objects, we can use this as a finite representation of the infinite qABox $\exists X'.\mathcal{A}'$.

We can show [6] that, for shell unfoldings, CQ-entailment can again be characterized by the existence of a homomorphism, and thus coincides with (model-based) entailment.

Proposition 11 ([6]). *If $\exists X'.\mathcal{A}'$ and $\exists Y'.\mathcal{B}'$ are shell unfoldings, then $\exists X'.\mathcal{A}' \models_{\text{CQ}} \exists Y'.\mathcal{B}'$ iff there is a homomorphism from $\exists Y'.\mathcal{B}'$ to $\exists X'.\mathcal{A}'$.*

If we want to work with (finitely represented) shell unfoldings in the context of CQ-repairs, we must be able to decide CQ-entailment, and thus the existence of

a homomorphism between shell unfoldings. This is possible in non-deterministic polynomial time in the size of the finite representation.

Theorem 12 ([6]). *Let $\exists X.\mathcal{A}$ and $\exists Y.\mathcal{B}$ be two finite qABoxes whose object sets are partitioned into kernel objects and shell objects as introduced above, and let $\exists X'.\mathcal{A}'$ and $\exists Y'.\mathcal{B}'$ be their shell unfoldings. Then the problem of deciding whether there is a homomorphism from $\exists X'.\mathcal{A}'$ to $\exists Y'.\mathcal{B}'$ is NP-complete in the size of the input $\exists X.\mathcal{A}$ and $\exists Y.\mathcal{B}$.*

Since a finite qABox can be seen as the shell unfolding of itself (with empty set of shell objects), this theorem also shows that answering CQs for shell unfoldings is NP-complete in the size of their finite representations.

Infinite CQ-saturation and CQ-repair The idea is now to extend the notion of a CQ-repair to a setting where qABoxes need not be finite, but must be finitely representable as the shell unfoldings of finite qABoxes. We call such qABoxes *rational qABoxes* since they consist of a finite part (the kernel) out of which grow (possibly) infinite trees, which are however rational [19]. We start with showing that, in this setting, finite qABoxes always have a CQ-saturation, even if the TBox is not cycle-restricted.

Given a finite qABox $\exists X.\mathcal{A}$ and an \mathcal{EL} TBox \mathcal{T} , we consider the shell unfolding of the IQ-saturation $\text{sat}_{\text{IQ}}^{\mathcal{T}}(\exists X.\mathcal{A})$, where all objects of the sub-qABox $\exists X.\mathcal{A}$ are kernel objects and all other objects (added by applications of the IQ-Saturation Rule) are shell objects. We can show that this rational qABox CQ-entails exactly those rational qABoxes that are CQ-entailed by $\exists X.\mathcal{A}$ and \mathcal{T} . It can thus replace the finite CQ-saturation from [7], but is not limited to cycle-restricted TBoxes. For this reason, we denote this shell unfolding by $\text{sat}_{\text{CQ}}^{\mathcal{T}}(\exists X.\mathcal{A})$ and call it the *CQ-saturation* of $\exists X.\mathcal{A}$ w.r.t. \mathcal{T} .

Proposition 13 ([6]). *Let $\exists X.\mathcal{A}$ be a finite qABox and \mathcal{T} an \mathcal{EL} TBox. Then $\exists X.\mathcal{A} \models_{\text{CQ}}^{\mathcal{T}} \exists Z.\mathcal{C}$ iff $\text{sat}_{\text{CQ}}^{\mathcal{T}}(\exists X.\mathcal{A}) \models_{\text{CQ}} \exists Z.\mathcal{C}$ for each rational qABox $\exists Z.\mathcal{C}$.*

Coming back to Example 6, where we constructed the IQ-saturation with kernel object n and shell object x_N , we now obtain as shell unfolding the CQ-saturation $\text{sat}_{\text{CQ}}^{\mathcal{T}}(\exists X.\mathcal{A}) = \exists \{x_1, x_2, \dots\}. \{N(n), \ell(n, x_1), N(x_1), \ell(x_1, x_2), N(x_2), \dots\}$, where $x_k := n \xrightarrow{\ell} x_N \xrightarrow{\ell} \dots \xrightarrow{\ell} x_N$.

$$\text{sat}_{\text{CQ}}^{\mathcal{T}}(\exists X.\mathcal{A}): \quad \begin{array}{c} N \\ \circlearrowleft \\ n \end{array} \xrightarrow{\ell} \begin{array}{c} N \\ \circlearrowleft \\ x_1 \end{array} \xrightarrow{\ell} \begin{array}{c} N \\ \circlearrowleft \\ x_2 \end{array} \xrightarrow{\ell} \begin{array}{c} N \\ \circlearrowleft \\ x_3 \end{array} \xrightarrow{\ell} \dots$$

Regarding repairs, we now allow them to be rational qABoxes, i.e., in Definition 2 the qABoxes $\exists Y.\mathcal{B}$ and $\exists Z.\mathcal{C}$ are allowed to be rational qABoxes rather than just finite one. We call such repairs *rational CQ-repairs*. But note that the input qABox is still assumed to be finite.

In this setting, the rôle of canonical CQ-repairs is now taken on by shell unfoldings of canonical IQ-repairs. In such an IQ-repair $\text{rep}_{\text{IQ}}^{\mathcal{T}}(\exists X.\mathcal{A}, \mathcal{S})$, an object

$\langle u, \mathcal{K} \rangle$ is a kernel object if u is a kernel object in the underlying IQ-saturation, and otherwise it is a shell object. We denote the shell unfolding of $\text{rep}_{\text{IQ}}^{\mathcal{T}}(\exists X. \mathcal{A}, \mathcal{S})$ as $\text{rep}_{\text{CQ}}^{\mathcal{T}}(\exists X. \mathcal{A}, \mathcal{S})$, and call it again the *canonical CQ-repair induced by \mathcal{S}* . The following proposition shows that using this notation is justified.

Proposition 14 ([6]). *Consider a finite qABox $\exists X. \mathcal{A}$, an \mathcal{EL} TBox \mathcal{T} , and an \mathcal{EL} repair request \mathcal{R} . For each repair seed \mathcal{S} , the induced canonical repair $\text{rep}_{\text{CQ}}^{\mathcal{T}}(\exists X. \mathcal{A}, \mathcal{S})$ is a rational CQ-repair of $\exists X. \mathcal{A}$ for \mathcal{R} . Conversely, if $\exists Z. \mathcal{C}$ is a rational CQ-repair of $\exists X. \mathcal{A}$ for \mathcal{R} , then there is a repair seed \mathcal{S} such that $\text{rep}_{\text{CQ}}^{\mathcal{T}}(\exists X. \mathcal{A}, \mathcal{S}) \models_{\text{CQ}} \exists Z. \mathcal{C}$.*

Note that the canonical CQ-repair must be constructed as shell unfolding of the *full* canonical IQ-repair, not from the optimized IQ-repair or any another qABox that is IQ-equivalent to it. In our Narcissus example with TBox, the canonical IQ-repair contains objects belonging to N , which are, however, not reachable from n . The optimized IQ-repair no longer contains such objects. Thus, the shell unfolding of the optimized repair does not entail $\exists \{x\}. \{N(x)\}$, but there are CQ-repairs that do, such as the shell unfolding of the canonical IQ-repair.

As an immediate consequence of the previous proposition, we obtain the main result of this section.

Theorem 15 ([6]). *Let $\exists X. \mathcal{A}$ be a finite qABox, \mathcal{T} an \mathcal{EL} TBox, and \mathcal{R} an \mathcal{EL} repair request. Then we can compute, in (deterministic) exponential time using an NP-oracle, a finite set of repair seeds $\{\mathcal{S}_1, \dots, \mathcal{S}_m\}$ such that the set $\{\text{rep}_{\text{CQ}}^{\mathcal{T}}(\exists X. \mathcal{A}, \mathcal{S}_1), \dots, \text{rep}_{\text{CQ}}^{\mathcal{T}}(\exists X. \mathcal{A}, \mathcal{S}_m)\}$ consists of all optimal rational CQ-repairs of $\exists X. \mathcal{A}$ for \mathcal{R} w.r.t. \mathcal{T} (up to CQ-equivalence). This set covers all rational CQ-repairs of $\exists X. \mathcal{A}$ for \mathcal{R} w.r.t. \mathcal{T} .*

Also note that the optimal repairs $\text{rep}_{\text{CQ}}^{\mathcal{T}}(\exists X. \mathcal{A}, \mathcal{S}_i)$ are saturated w.r.t. \mathcal{T} in the sense that they CQ-entail a rational qABox w.r.t. \mathcal{T} if they already entail it without \mathcal{T} . By Theorem 12, this implies that conjunctive queries can be answered for $\text{rep}_{\text{CQ}}^{\mathcal{T}}(\exists X. \mathcal{A}, \mathcal{S}_i)$ in non-deterministic polynomial time in the size of $\text{rep}_{\text{IQ}}^{\mathcal{T}}(\exists X. \mathcal{A}, \mathcal{S}_i)$ and the query.

6 Conclusion

In the first part of this paper we have mainly recalled the approaches and results from [7, 16]. In other work, we have extended these results in several directions. The paper [11] extends the expressivity of the underlying DL considerably, by adding nominals, inverse roles, regular role inclusions and the bottom concept to \mathcal{EL} , which yields a fragment of the well-known DL Horn-*SROIQ* [29]. In [9], we investigate whether and how one can obtain optimal repairs if one restricts the output of the repair process to being ABoxes rather than qABoxes. In general, such optimal ABox repairs need not exist. The main contribution of the paper is an approach that can decide the existence of optimal ABox repairs in exponential time, and can compute all such repairs in case they exist. The papers [13, 14] consider error-tolerant reasoning based on optimal repairs and [1] compares optimal

repairs with contractions from the area of belief change. Moreover, an approach to computing optimal repairs of \mathcal{EL} TBoxes is developed in [25].

In the second part of this paper we have presented new results on how to represent exponentially large repairs in a polynomial way and infinite repairs in a finite way. It would be interesting to see whether such approaches can also be extended to other settings. We conjecture that non-cycle-restricted TBoxes can still be tackled by using shell-unfoldings for the DLs considered in [11]. However, in [11] we also show that optimal repairs need not exist if the role inclusions are not regular. It is unclear whether this problem can be overcome by an appropriate finite representation of infinite repairs. Another interesting topic for future research is to investigate whether finitely represented rational repairs can be used in practice.

Authors' Contributions. FB and FK contributed equally to the paper. PK ran the experiments and wrote the description of them. He also wrote a first version of the proof of the last proposition in Section 5 of [6].

Acknowledgements. This work has been supported by Deutsche Forschungsgemeinschaft (DFG) in projects 430150274 (Repairing Description Logic Ontologies) and 389792660 (TRR 248: Foundations of Perspicuous Software Systems).

References

1. Baader, F.: Optimal repairs in ontology engineering as pseudo-contractions in belief change. In: Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing (SAC '23), March 27–31, 2023, Tallinn, Estonia. pp. 983–990. Association for Computing Machinery (2023), <https://doi.org/10.1145/3555776.3577719>
2. Baader, F., Borgwardt, S., Morawska, B.: SAT Encoding of Unification in \mathcal{ELH}_{R^+} w.r.t. Cycle-Restricted Ontologies. In: Gramlich, B., Miller, D., Sattler, U. (eds.) Automated Reasoning - 6th International Joint Conference, IJCAR 2012, Manchester, UK, June 26–29, 2012. Proceedings. Lecture Notes in Computer Science, vol. 7364, pp. 30–44. Springer (2012), https://doi.org/10.1007/978-3-642-31365-3_5
3. Baader, F., Brandt, S., Lutz, C.: Pushing the \mathcal{EL} envelope. In: Kaelbling, L.P., Saffiotti, A. (eds.) IJCAI-05, Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence, Edinburgh, Scotland, UK, July 30 - August 5, 2005. pp. 364–369. Professional Book Center (2005), <http://ijcai.org/Proceedings/05/Papers/0372.pdf>
4. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P.F. (eds.): The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press (2003)
5. Baader, F., Horrocks, I., Lutz, C., Sattler, U.: An Introduction to Description Logic. Cambridge University Press (2017), <https://doi.org/10.1017/9781139025355>
6. Baader, F., Koopmann, P., Kriegel, F.: Optimal repairs in the description logic \mathcal{EL} revisited (extended version). LTCS-Report 23-03, Chair of Automata Theory, Institute of Theoretical Computer Science, Technische Universität Dresden, Dresden, Germany (2023), <https://doi.org/10.25368/2023.121>

7. Baader, F., Koopmann, P., Kriegel, F., Nuradiansyah, A.: Computing optimal repairs of quantified ABoxes w.r.t. static \mathcal{EL} TBoxes. In: Platzer, A., Sutcliffe, G. (eds.) Automated Deduction - CADE 28 - 28th International Conference on Automated Deduction, Virtual Event, July 12-15, 2021, Proceedings. Lecture Notes in Computer Science, vol. 12699, pp. 309–326. Springer (2021), https://doi.org/10.1007/978-3-030-79876-5_18
8. Baader, F., Koopmann, P., Kriegel, F., Nuradiansyah, A.: Computing optimal repairs of quantified ABoxes w.r.t. static \mathcal{EL} TBoxes (extended version). LTCS-Report 21-01, Chair of Automata Theory, Institute of Theoretical Computer Science, Technische Universität Dresden, Dresden, Germany (2021), <https://doi.org/10.25368/2022.64>
9. Baader, F., Koopmann, P., Kriegel, F., Nuradiansyah, A.: Optimal ABox repair w.r.t. static \mathcal{EL} TBoxes: from quantified ABoxes back to ABoxes. In: 19th Extended Semantic Web Conference, ESWC 2022, Hersonissos, Greece, May 29 – June 2, 2022, Proceedings. Lecture Notes in Computer Science, vol. 13261, pp. 130–146. Springer (2022), https://doi.org/10.1007/978-3-031-06981-9_8
10. Baader, F., Koopmann, P., Kriegel, F., Nuradiansyah, A.: Optimal ABox repair w.r.t. static \mathcal{EL} TBoxes: from quantified ABoxes back to ABoxes (extended version). LTCS-Report 22-01, Chair of Automata Theory, Institute of Theoretical Computer Science, Technische Universität Dresden, Dresden, Germany (2022), <https://doi.org/10.25368/2022.65>
11. Baader, F., Kriegel, F.: Pushing optimal ABox repair from \mathcal{EL} towards more expressive Horn-DLs. In: Kern-Isberner, G., Lakemeyer, G., Meyer, T. (eds.) Proceedings of the 19th International Conference on Principles of Knowledge Representation and Reasoning, KR 2022, Haifa, Israel, July 31 – August 5, 2022. pp. 22–32 (2022), <https://doi.org/10.24963/kr.2022/3>
12. Baader, F., Kriegel, F., Nuradiansyah, A.: Privacy-preserving ontology publishing for \mathcal{EL} instance stores. In: Calimeri, F., Leone, N., Manna, M. (eds.) Logics in Artificial Intelligence - 16th European Conference, JELIA 2019, Rende, Italy, May 7-11, 2019, Proceedings. Lecture Notes in Computer Science, vol. 11468, pp. 323–338. Springer (2019), https://doi.org/10.1007/978-3-030-19570-0_21
13. Baader, F., Kriegel, F., Nuradiansyah, A.: Error-tolerant reasoning in the description logic \mathcal{EL} based on optimal repairs. In: Governatori, G., Turhan, A. (eds.) Rules and Reasoning - 6th International Joint Conference, RuleML+RR 2022, Virtual, September 26-28, 2022, Proceedings. Lecture Notes in Computer Science, vol. 13752, pp. 227–243. Springer (2022), https://doi.org/10.1007/978-3-031-21541-4_15
14. Baader, F., Kriegel, F., Nuradiansyah, A.: Treating role assertions as first-class citizens in repair and error-tolerant reasoning. In: Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing (SAC '23), March 27–31, 2023, Tallinn, Estonia. pp. 974–982. Association for Computing Machinery (2023), <https://doi.org/10.1145/3555776.3577630>
15. Baader, F., Kriegel, F., Nuradiansyah, A., Peñaloza, R.: Making repairs in description logics more gentle. In: Thielscher, M., Toni, F., Wolter, F. (eds.) Principles of Knowledge Representation and Reasoning: Proceedings of the Sixteenth International Conference, KR 2018, Tempe, Arizona, 30 October - 2 November 2018. pp. 319–328. AAAI Press (2018), <https://aaai.org/ocs/index.php/KR/KR18/paper/view/18056>
16. Baader, F., Kriegel, F., Nuradiansyah, A., Peñaloza, R.: Computing compliant anonymisations of quantified ABoxes w.r.t. \mathcal{EL} policies. In: Pan, J.Z., Tamma,

- V.A.M., d'Amato, C., Janowicz, K., Fu, B., Polleres, A., Seneviratne, O., Kagal, L. (eds.) The Semantic Web - ISWC 2020 - 19th International Semantic Web Conference, Athens, Greece, November 2-6, 2020, Proceedings, Part I. Lecture Notes in Computer Science, vol. 12506, pp. 3–20. Springer (2020), https://doi.org/10.1007/978-3-030-62419-4_1
17. Baader, F., Suntisrivaraporn, B.: Debugging SNOMED CT using axiom pinpointing in the description logic \mathcal{EL}^+ . In: Cornet, R., Spackman, K.A. (eds.) Proceedings of the Third International Conference on Knowledge Representation in Medicine, Phoenix, Arizona, USA, May 31st - June 2nd, 2008. CEUR Workshop Proceedings, vol. 410. CEUR-WS.org (2008), <http://ceur-ws.org/Vol-410/Paper01.pdf>
 18. Brachman, R.J., Fikes, R., Levesque, H.J.: Krypton: A functional approach to knowledge representation. *Computer* **16**(10), 67–73 (1983), <https://doi.org/10.1109/MC.1983.1654200>
 19. Colmerauer, A.: Prolog and infinite trees. In: Clark, K., Tarnlund, S.A. (eds.) *Logic Programming*, pp. 231–251. Academic Press, New York (1982)
 20. Cuenca Grau, B., Kostylev, E.V.: Logical foundations of linked data anonymisation. *J. Artif. Intell. Res.* **64**, 253–314 (2019), <https://doi.org/10.1613/jair.1.11355>
 21. Du, J., Qi, G., Fu, X.: A practical fine-grained approach to resolving incoherent OWL 2 DL terminologies. In: Proc. of the 23rd ACM Int. Conf. on Information and Knowledge Management, (CIKM'14). pp. 919–928 (2014), <http://doi.acm.org/10.1145/2661829.2662046>
 22. Greiner, R., Smith, B.A., Wilkerson, R.W.: A correction to the algorithm in Reiter's theory of diagnosis. *Artif. Intell.* **41**(1), 79–88 (1989), [https://doi.org/10.1016/0004-3702\(89\)90079-9](https://doi.org/10.1016/0004-3702(89)90079-9)
 23. Horridge, M., Parsia, B., Sattler, U.: Laconic and precise justifications in OWL. In: Sheth, A.P., Staab, S., Dean, M., Paolucci, M., Maynard, D., Finin, T.W., Thirunarayan, K. (eds.) The Semantic Web - ISWC 2008, 7th International Semantic Web Conference, ISWC 2008, Karlsruhe, Germany, October 26-30, 2008. Proceedings. Lecture Notes in Computer Science, vol. 5318, pp. 323–338. Springer (2008), https://doi.org/10.1007/978-3-540-88564-1_21
 24. Kazakov, Y., Krötzsch, M., Simancik, F.: The incredible ELK - from polynomial procedures to efficient reasoning with \mathcal{EL} ontologies. *Journal of Automated Reasoning* **53**(1), 1–61 (2014), <https://doi.org/10.1007/s10817-013-9296-3>
 25. Kriegel, F.: Optimal fixed-premise repairs of \mathcal{EL} TBoxes. In: Bergmann, R., Malburg, L., Rodermund, S.C., Timm, I.J. (eds.) Proceedings of the 45th German Conference on Artificial Intelligence (KI 2022), Virtual in Trier, Germany, September 19–23, 2022. Lecture Notes in Computer Science, vol. 13404, pp. 115–130. Springer (2022), https://doi.org/10.1007/978-3-031-15791-2_11
 26. Lam, J.S.C., Sleeman, D.H., Pan, J.Z., Vasconcelos, W.W.: A fine-grained approach to resolving unsatisfiable ontologies. *J. Data Semant.* **10**, 62–95 (2008), https://doi.org/10.1007/978-3-540-77688-8_3
 27. Levesque, H.J.: Foundations of a functional approach to knowledge representation. *Artif. Intell.* **23**(2), 155–212 (1984), [https://doi.org/10.1016/0004-3702\(84\)90009-2](https://doi.org/10.1016/0004-3702(84)90009-2)
 28. Lutz, C., Wolter, F.: Deciding inseparability and conservative extensions in the description logic \mathcal{EL} . *J. Symb. Comput.* **45**(2), 194–228 (2010), <https://doi.org/10.1016/j.jsc.2008.10.007>
 29. Ortiz, M., Rudolph, S., Šimkus, M.: Worst-case optimal reasoning for the Horn-DL fragments of OWL 1 and 2. In: Lin, F., Sattler, U., Truszczyński, M. (eds.)

- Principles of Knowledge Representation and Reasoning: Proceedings of the Twelfth International Conference, KR 2010 (2010), <http://aaai.org/ocs/index.php/KR/KR2010/paper/view/1296>
30. Parsia, B., Matentzoglou, N., Gonçalves, R.S., Glimm, B., Steigmiller, A.: The OWL reasoner evaluation (ORE) 2015 competition report. *J. Autom. Reason.* **59**(4), 455–482 (2017), <https://doi.org/10.1007/s10817-017-9406-8>, test ontology corpus: <https://doi.org/10.5281/zenodo.18578>
 31. Parsia, B., Rudolph, S., Hitzler, P., Krötzsch, M., Patel-Schneider, P.: OWL 2 web ontology language primer (second edition). W3C recommendation (2012), <http://www.w3.org/TR/2012/REC-owl2-primer-20121211/>
 32. Parsia, B., Sirin, E., Kalyanpur, A.: Debugging OWL ontologies. In: Ellis, A., Hagino, T. (eds.) Proceedings of the 14th international conference on World Wide Web, WWW 2005, Chiba, Japan, May 10-14, 2005. pp. 633–640. ACM (2005), <https://doi.org/10.1145/1060745.1060837>
 33. Reiter, R.: A theory of diagnosis from first principles. *Artif. Intell.* **32**(1), 57–95 (1987), [https://doi.org/10.1016/0004-3702\(87\)90062-2](https://doi.org/10.1016/0004-3702(87)90062-2), see the erratum [22].
 34. Schlobach, S., Huang, Z., Cornet, R., van Harmelen, F.: Debugging incoherent terminologies. *J. Autom. Reason.* **39**(3), 317–349 (2007), <https://doi.org/10.1007/s10817-007-9076-z>
 35. Troquard, N., Confalonieri, R., Galliani, P., Peñaloza, R., Porello, D., Kutz, O.: Repairing ontologies via axiom weakening. In: McIlraith, S.A., Weinberger, K.Q. (eds.) 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. pp. 1981–1988. AAAI Press (2018), <https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17189>